GAMIFICATION IN INTRODUCTORY COMPUTER SCIENCE

by

KARA ALEXANDRA BEHNKE

B.A., University of Colorado Boulder, 2010

M.A., Michigan State University, 2014

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado Boulder in partial fulfillment

of the requirement for the degree of

Doctor of Philosophy

ATLAS Institute

2015

This thesis entitled:

Gamification in Introductory Computer Science

written by Kara Alexandra Behnke
has been approved for the ATLAS Institute

_____
John K. Bennett

_____
Dirk Grunwald

_____
Clayton Lewis

_____
Jane Prey

_____
Doug C. Sicker

_____
Diane E. Sieber

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

IRB protocol # 15-0018
IRB protocol # 15-0119

Behnke, Kara Alexandra (Ph.D., ATLAS Institute)

Gamification in Introductory Computer Science

Thesis directed by Archuleta Professor of Computer Science and Professor of Electrical and Computer Engineering Dr. John K. Bennett

This thesis investigates the impact of gamification on student motivation and learning in several introductory computer science educational activities. The use of game design techniques in education offers the potential to make learning more motivating and more enjoyable for students. However, the design, implementation, and evaluation of game elements that actually realize this promise remains a largely unmet challenge. This research examines whether the introduction of game elements into curriculum positively impacts student motivation and intended learning outcomes for entry-level computer science education in four settings that apply similar game design techniques in different introductory computer science educational settings. The results of these studies are evaluated using mixed methods to compare the effects of game elements on student motivation and learning in both formal and non-formal learning environments.

The first chapter of this thesis introduces the focus of the research. The second chapter discusses the related work on introductory computer science education and gaming culture. The third chapter introduces the conceptual and theoretical framework that forms the basis of this work. Chapter four introduces the research design. Chapter five presents the first case study, which investigated how a "gamified" class impacts student motivation and learning outcomes in an undergraduate computer science course. Chapter six discusses the second case study, which examined how the design techniques informed by the first case study support high school students learning introductory computer science in an extracurricular program. Chapter seven details the third case study, which examined how game elements influence children learning in a robotics workshop. Chapter eight discusses the final case study, which briefly examined how online tutorial websites facilitate introductory programming through gaming. The final chapter summarizes the key findings and discusses opportunities for future work in gamifying computer science education.

ACKNOWLEDGEMENTS

I am grateful for my advisor, Dr. John K. Bennett, who first believed in my ability. I am profoundly thankful for all my committee members: Dr. Diane Sieber for her insight in gaming and passion for the humanities in engineering; Dr. Clayton Lewis, whose work in both teaching and research continues to inspire me in all my academic endeavors; Dr. Dirk Grunwald, who helped connect me to the computer science education committee both in Boulder, CO and abroad; Dr. Doug Sicker, for the invigorating talks about policy and technology; and Dr. Jane Prey, your work at both the National Science Foundation and Microsoft Research is something I aspire for in my professional ambitions. Thanks to each of you for helping me accomplish the most challenging task I have ever sought out to do.

I would not be who I am today, nor made it through this doctoral journey, without my Chosen Family. Zander, my husband and life partner, whose love, encouragement, and grounded reassurance helped me pull myself through many tough times. Dan, who has helped me grow in more ways than I thought possible, and for helping me manage my data. Julia, experiencing this academic journey alongside your own has been a privilege in its own right. Thank you to my Consigli and Behnke families for many happy memories and much needed home-cooked meals.

My gratitude for the ATLAS Institute and its community are beyond words. I never felt that I fit in anywhere academically or professionally until I found this inspiring community of creative, talented, fantastic misfits. I want to especially thank Brittany Ann Kos, the best TA and second coder I could have hoped for. Special thanks to Dr. Meg Ambrose, Dr. Katherine Goodman, Dr. Joanne White, Dr. Leslie Dodson, and Dr. Heather Underwood, who each in turn inspired and encouraged me throughout this process.

I would also like to thank the National Science Foundation GK-12 ECSITE Program, who offered me the graduate funding and professional support to help me finish my doctorate. Special thanks to Jessica Feld, Dr. Debra Goldberg, the other ECISTE Fellows, and the magnificent high school teachers I worked with during my time as a Fellow—I learned more about teaching students and teaching computer science than I could have imagined.

Thank you to everyone who helped make this journey become an “Epic Win!”

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I
INTRODUCTION

I. BACKGROUND AND MOTIVATION

Information and communication technologies have profoundly altered interaction in human society, productivity in enterprise, and the creation and dissemination of information (NRC, 2003; NRC 2005; NRC, 2007). The use of modeling and simulation, visualization, and the management of massive data sets have fostered computer science as a bridge between science, technology, engineering, and mathematics (STEM) disciplines. Computer science (CS) has enabled significant advances in innovation and imagination, and continues to facilitate efforts to address many social problems.

Although the United States has become increasingly dependent upon information and communication technologies, many of its citizens are ill-equipped to make informed decisions about technology (NRC, 2005), and also lack the knowledge and capability to take advantage of the many opportunities emerging from technological and social change (NRC, 2005; Owens & Stephenson, 2011). Professionals in every discipline—art and entertainment, communications and health care, government infrastructure and economic exchange—need to understand and be able to employ computing in order to function and thrive in a globally competitive environment (NRC, 2007). Without meaningful engagement and facility with computer science by the general public, the U.S. economy risks losing competency and self-efficacy in the global networked age (NRC, 2007; NRC, 2005).

Many jobs that today's students will have during their careers have not yet been invented, and the majority of careers in the 21st century will require some background in computer science

(Owens & Stephenson, 2011; NRC, 2007). The U.S. Bureau of Labor Statistics (2013) estimates that between 2010 and 2020 approximately 1.4 million computing-related jobs will be created in the U.S. alone. Even though the computing industry has more projected career growth than all other jobs combined, including all STEM careers (Bureau of Labor Statistics, 2013), less than 2.4% of college students graduate with a degree in computer science. Even though undergraduate student enrollment in computer science has increased slightly over the last few years, CS as a field of study historically has high attrition rates, and in some cases, negative growth for non-majority groups (ACM & IEEE, 2013). Diversity is critical to innovation and excellence (NRC, 2005; NRC, 2007); thus increasing diversity is key to long-term economic growth and global competitiveness (NRC, 2003; NRC 2005; NRC, 2007). However, only 26% of the computing workforce are women, and less than 4% of the computing workforce are Latina or African American women (NCWIT, 2015). Therefore, a basic understanding of and facility with computational thinking (Wing, 2009), computing literacy, and computer science is essential to being part of a well-educated and informed citizenry in the twenty-first century (NRC, 2004; NRC, 2007). As the Information Age continues to transform virtually every aspect of life, computer science must become part of a general liberal education. The required understanding of computing is deeper than the ability to use a computer or a mobile phone for day-to-day personal productivity or social interaction.

How should we teach students computer science skills? Seymour Papert (1971; 1980) argued that learning computer science is a way of learning about learning. Computer science can help individuals learn how to think critically and creatively (Owens & Stephenson, 2011; NRC, 2003); how to approach ambiguous and challenging problems (ACM & IEEE, 2013); how to analyze and synthesize vast amounts of information (NRC, 2003); how to use computation as an

analysis tool (NRC, 2007; Wing, 2009); how to understand relationships between seemingly disparate data (NRC, 2007); and how to collaborate (NRC, 2005). Learning computer science thus affords individuals the ability to navigate more effectively through a society in which they frequently encounter technology in their personal lives (NRC, 2010). Why then, do so few students pursue studies in this critically important field?

II. PRIOR RESEARCH

College and high school curricula have struggled to provide students with engaging methods for learning both basic and advanced computational concepts, and efforts to incorporate computer science into K-12 curricula have met with limited success. Only twenty-seven states in the U.S. allow students to count computer science courses towards high school graduation (Association of Computing Machinery, 2011). Moreover, the majority of schools in the U.S. do not even offer computer programming classes (Association of Computing Machinery, 2011; Gallup, 2014). When courses are offered, many students, particularly female students and those from underrepresented minority groups, are choosing not enroll in CS classes (ACM & IEEE, 2013). Although the number of students taking Advanced Placement (AP) CS courses has increased in recent years (Ericson & Guzdial, 2014), the number of students taking this examination is much smaller than for other AP STEM Examinations. For example, of the 1,379,585 AP math and science exams taken by U.S. high school students in 2013, only 29,555 were computer science exams—a 2:98 ratio of students in computer science relative to the rest of STEM. In addition, the participation rate of underrepresented groups in AP computer science, including female, African American, and Latina students, is also lower than other STEM AP

examinations[1]. Ericson and Guzdial (2014) argue, "If you want more AP CS exam-takers, the focus should be on the underrepresented groups" (p. 222). Issues of privileged access may be another reason why non-majority populations continue to have low representations in the computing workforce (NCWIT, 2015). The low participation of students, particularly non-majority groups, in the Advanced Placement Computer Science curriculum is emblematic of this problem.

Many students, particularly girls and women, find traditional approaches for learning computer science off-putting, asocial, and boring (AAUW, 2000; Margolis & Fisher, 2002; Ashcraft et al., 2012). How best to teach introductory computer science on a broad level, particularly for those with limited exposure or no previous experience in CS, remains a significant challenge. Since many young people are actively engaged in digital gaming[2] spaces (Van Eck, 2006; Kafai et al., 2008; ESA, 2014) and video games[3] themselves are computational artifacts, digital gaming offers a potentially promising approach to increase student interest and engagement for learning computer science.

There have been numerous efforts to translate the engagement found in games into education (Ritzhaupt, 2009; Linhoff & Settle, 2009; Barnes, Richter & Powell, 2007; Eagle & Barnes, 2009; Cooper, Dann, & Pausch, 2003; Malan & Leitner, 2007; Basawapatna, Koh & Repenning, 2010). Many argue that games embody beneficial learning principles (Malone, 1980a, 1980b; Shaffer, 2008; Collins & Halverson, 2009; Gee, 2003; Squire, 2011, 2003) by promoting

[1] In 2012, AP Calculus had a higher percentage of underrepresented groups take the exam than CS: for females 48.3% (Calculus) versus 18.7% (CS), for African American students 5.4% versus 4.0%, and Latina students 11.5% versus 7.7% (Arpaci-Dusseau et al., 2013).
[2] Digital gaming refers to video games, computer games, mobile hand-held games, social-media games, etc.
[3] I use *video games*, *gaming*, and *games* interchangeably to denote the broader pop-cultural applications of all digital gaming spaces, since the vernacular of "video games" is often used to describe this phenomenon.

extrinsic and intrinsic motivations (Schmitz et al., 2011), allowing the freedom to explore and fail (Mitgutsch, 2012; Malaby, 2007), and fostering effective problem-solving skills (Squire et al., 2008; Gee, 2003; Bogost, 2007; Khaled et al., 2009).

Despite the numerous attempts to use games in education, these efforts have yielded only mixed results (Charsky, 2010; Van Eck, 2006). Many game designers and educators have attempted to harness the power of games for learning, but these efforts have often resulted in boring games with ineffective learning (Papert, 1980; Mitgutsch & Alvarado, 2012). As Bogost (2011) succinctly states: "Making games is hard. Making good games is even harder. Making good games that hope to serve some external purpose is even harder" (p. 3). Some studies have made progress in developing methodologies for designing games that obtain or retain the player's engagement, while also achieving some explicit learning purpose (Mitgutsch & Alvarado, 2012; Wong et al., 2007; Raybourn & Bos, 2005; Von Ahn & Dabbish, 2008; Raybourn et al., 2005; Wideman et al., 2007). One approach, called gamification, seeks to combine the motivational power of video games to serve an explicit purpose.

Gamification is the process of incorporating game elements into non-game contexts in an effort to increase user-engagement (Deterding et al., 2011; Deterding & Dixon, 2011; Nicholson, 2012; Huotari & Hamari, 2012). Common implementations of gamification include incorporating the scoring elements of games, such as points, levels, and achievements, and applying them to a work or educational environment (Deterding & Dixon, 2011; Deterding et al., 2011). Effective gamification allows participants to use game elements as a way to reflect on their completion of a learned activity, as noted by Deterding (2011):

> "Indeed, games are full of points, scores, tokens, and so on… the 'fun', the pleasure of these elements does not come from some extrinsic reward value of those elements, but chiefly from the experience of competence they give rise to."

Gamification has the potential to increase user engagement in learning, while providing feedback on that learning. Since students often report a lack of engagement when learning introductory computer science, gamification has the potential to provide a way to promote student motivation and engagement while also providing feedback on the student's level of competency of the learned material.

Thus, there appears to be a good fit between introductory computer science education and gamification. One of gamification's core design principles is to "provide feedback so that players can achieve a sense of mastery" (Richards et al., 2014). Mastering an activity, and learning through failure, are often represented in gamification in the form of achievements, badges, high leaderboard rank, reaching the maximum level, etc. (Richards et al., 2014). Games maintain this positive relationship with failure by making feedback cycles rapid and keeping the stakes low—players can keep trying until they succeed and they risk very little by trying different alternatives. In schools, on the other hand, the stakes of failure are high and the feedback cycles tend to be long. Students have few opportunities to experiment, and when they do, the results of failure are lasting, causing anxiety, rather than anticipation, when offered the chance to fail (Pope, 2003). In contrast, gamification can shorten feedback cycles, give learners low-stakes ways to assess their own skills and capabilities, and create an environment in which effort in learning is rewarded. Students, in turn, can learn to see failure as a learning opportunity. This may be particularly useful in teaching introductory computer science to students with limited experience in CS. Gamification, applied to introductory CS pedagogy, may provide students an engaging, effective, novel way for students

to learn introductory computer science skills and thereby increase attraction and retention for students into the field.

III. RESEARCH GOALS

The central objective of this research is to explore how the design and implementation of game elements impact student motivation and learning in introductory computer science education. When appropriate, I employed the Computer Science Teacher Association (CSTA) K-12 Computer Science Standards to model the curriculum and learning outcomes of this work. In addition, I investigated the role of introductory CS education within formal and non-formal learning environments, since CS learning is happening both inside and outside of the classroom (ACM & CSTA, 2011; CSTA, 2011), and in learning environments for students at different ages, multicultural backgrounds, and with varying experience using and developing computing technology. The results of these investigations contribute insight and ideas for educators interested in using game design techniques to help make learning experiences more engaging for introductory CS students.

As part of this research, I investigated the impact of game elements in an undergraduate introductory computer science course, where one section of the course included game elements and one did not. This experiment sought to assess the motivational role and impact of game elements in otherwise identical courses. I also explored the impact of gaming on student motivation and learning in an extracurricular mobile "app" development program for high school Latinas. Here, I focused on the role of game elements for students who have historically had few opportunities to engage with or develop computer technology. Many of these students also face challenging socio-economic issues. The third area of exploration involved a learning workshop

for elementary and early middle-school children learning about computational thinking and engineering concepts for the first time, where I examined how game elements supported children's engagement in their learning. Finally, I examined the learning affordances of several popular online tutorials that teach introductory computer science for free, and considered whether gaming components in these tutorials have impact on the motivation or learning of individuals using the online tools.

IV. METHODS

To explore the impact of gamification on student motivation and learning, I investigated several ways in which game design techniques could be situated into "real world" learning scenarios, using the lens of Self-Determination Theory (Deci & Ryan, 1991). Four case studies (Stake, 1995) were conducted to explore impact of gamification within introductory CS learning settings. The Self-Determination Theory framework allowed the examination of the way students experience intrinsic and extrinsic motivation within the socio-cultural context of the learning environment in which they are embedded. As part of this work, I considered how students, particularly females and minority groups, are impacted and motivated by the gamification of CS curricula. The four case studies include (1) an introductory undergraduate CS course intended predominately for non-majors, (2) an extracurricular mobile app development program for high school Latinas, (3) a public library CS workshop series for elementary and early middle school children, and (4) the assessment of different gaming and non-gaming online tutorial websites that teach introductory programming. The table below summarizes the research methodology for each case study in more detail.

Case Study Name	Learning Environment	Gamification Prototype	Number of Participants (*n*)	Duration	Methods
1. Virtual Worlds	Formal	Website	*n* = 94	16 weeks	• Surveys • Course grades • Web analytics • Content analysis
2. Technovation	Informal	Website	*n* = 7	12 weeks	• Surveys • Focus group interviews • Web analytics • Content analysis
3. CoderDojo	Informal	Workbook	*n* = 36	4 sessions over 8 weeks	• Surveys • Ethnographic observation • Content analysis
4. Online Tutorials	Informal	Website	*n* = 11	8 weeks	• Pre / Post Surveys • Content Analysis

Table 1. Summary of Case Study Methodology.

This dissertation first presents a review of the relevant literature and the conceptual framework underlying this research. I then present and discuss the methods and findings for each of the four case studies. The first case study examines the feasibility of using gamification to support *Computer Science Principles* pedagogical outcomes within a formal academic environment, in this case, an undergraduate classroom. The second and third studies consider gamification solutions for "outside-of-school" learning settings. The fourth case study considers how game-based learning, gamification, and non-gaming website tutorials that teach computer programming can influence student motivation and knowledge retention.

Findings from the case studies reveal that students enjoy "playful design" if the game elements support either their intrinsic or extrinsic motivations to learn the material. However,

when the game elements do not align with a student's individual learning goals or style of preferred motivation, then gamification can have a potentially negative effect on student motivation to pursue the learning material. Therefore, the game design techniques must strategically align with both the student's intrinsic and/or extrinsic motivational engagements and the overall pedagogical goals of the curricula.

Chapter 1 introduces the problem. Chapter 2 discusses the related work on introductory computer science education, video games and learning, and human motivation research. Chapter 3 introduces the conceptual and theoretical framework that forms the basis of this work. Chapter 4 introduces the research design and methodology. Chapter 5 presents the first case study, which investigates how a "gamified" class impacts student motivation and learning outcomes in an undergraduate computer science course. Chapter 6 discusses the second case study, which examines how game design techniques informed by the first case study influence high school students in an extracurricular mobile app development program. Chapter 7 details the third case study, which examines how game elements influence children learning in a modular robotics workshop. Chapter 8 discusses the final case study, which explores how gaming influences students while learning programming through online tutorial websites. Chapter 9 summarizes the research findings and discusses opportunities for future work in gamifying computer science education.

CHAPTER II
RELATED WORK

This work builds upon prior work in two main areas: 1) advances in introductory computer science education in the United States, and 2) research at the intersection of gaming and learning. This section reviews the most relevant of this prior work.

I. INTRODUCTORY COMPUTER SCIENCE EDUCATION

A basic understanding and facility with computer science is becoming part of the basic education of an informed citizenry. However, even though the computing industry has more projected career growth than all other jobs combined (Bureau of Labor Statistics, 2013), student enrollment in computer science as a field of study typically exhibits high attrition rates, particularly among non-majority groups such as women and students of color (ACM & IEEE, 2013). Two thirds of American secondary-schools do not implement any computer science standards (Wilson et al., 2010), nor accept computer science as credit towards high school graduation (ACM & CSTA, 2011). Moreover, nine out of ten high schools in the U.S. do not even offer computer science courses to students (ACM & CSTA, 2011; CSTA, 2011). These trends indicate that CS concepts and courses in K-12 curriculum have not kept pace with other academic disciplines in the U.S. As a result, the general public is not as well educated about CS as it should be, to the point that the nation faces a serious shortage of computer scientists at all levels that is likely to continue into the foreseeable future (ACM & CSTA, 2011).

1. Defining "Computer Science Education" Terminology

Computer science is being shaped and defined as new technological innovations emerge, which has resulted in evolution in what constitutes computer science. The three most common

areas of computing education currently offered in schools include educational technology, information technology, and computer science.

Educational technology is the use of computing technology to learn about other disciplines (CSTA, 2011). The primary goal of educational technology is to integrate technology into teaching in order to advance student learning across academic disciplines (ACM & CSAT, 2011). *Information technology* (IT) is defined as "the proper use of technologies by which people manipulate and share information in its various forms" (CSTA, 2011, p. 6). IT classes often involve students learning how to use technology, such as keyboard-typing, word processing, inputting and manipulating data into spreadsheets, etc. While IT involves learning about computers, it emphasizes learning about the technology itself, such as integrating hardware and software products with organizational needs and infrastructure, rather than understanding how that technology works. Terminology such as *IT literacy* or *IT fluency* similarly suggest that students learn how to use and learn new forms of technology, but not learn how to create that technology.

Computer science, on the other hand, spans a wide range of computing endeavors, including the design and implementation of software, developing effective ways to solve computing problems, and devising new ways to use computers. The Association of Computing Machinery and the Computer Science Teachers Association (2011) define *computer science* education as "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society" (p. 1). This research uses this definition of CS education, which focuses not only on using technology but making technology as part of the learning process in the curriculum.

2. CSTA K-12 Standards

Even when CS courses and curriculum are available to students, the methods and practices for how best to teach introductory computer science represent a significant pedagogical challenge. In an effort to address this problem, the Computer Science Teacher Association (CSTA) Standards Task Force and the Association for Computing Machinery Inc. (ACM) provided a white-paper in 2011 with comprehensive standards for K-12 computer science education. The CSTA Standards (2011) are designed to strengthen computer science fluency and competency throughout primary and secondary schools in an effort to provide academic coherence between coursework, the rapid growth of computing and technology. These standards are also intended to foster an educated public who can employ that technology for the benefit of society. The core set of learning standards are designed to provide the foundation for a complete CS curriculum and its implementation at the K-12 level.

The CSTA Standards (2011) propose a three-level framework for computer science education. The framework focuses on fundamental concepts with four general goals:

1) The curriculum should prepare students to understand the nature of computer science and its place in the modern world.
2) Students should understand that computer science interweaves concepts and skills.
3) Students should be able to use computer science skills (especially computational thinking) in their problem-solving activities in other subjects.
4) The computer science standards should complement IT and AP computer science curricula in schools where they are currently being offered.

Level one is intended for grades K-6, where learning outcomes are expected to be addressed in the context of other academic subjects such as math, science, and social studies. Level two learning outcomes, intended for students in grades 6-9, can be addressed either through other subjects or in discrete computer science courses. Level three is intended for grade 9-12 high school students and divided into three separate courses: *Computer Science in the Modern World*, *Computer Science Principles*, and *Topics in Computer Science*.

CS as a field of study has been hampered by the perception that the field focuses exclusively on programming. This misconception has been particularly damaging in grades K-12, where courses' limited scope have been negatively perceived by students (Margolis & Fisher, 2002; Rich, Perry, & Guzdial, 2004). This problem is exacerbated by perceptions of CS as a solitary pursuit, disconnected from other intellectual endeavors, with little relevance to the interests and concerns of students (ACM & IEEE, 2013). To address these concerns, the CSTA Standards distinguish five complementary and essential content "strands" throughout all three levels of these standards. These strands include:

- Computational Thinking
- Collaboration
- Computing Practice and Programming
- Computers and Communication Devices
- Community, Global, and Ethical Impacts

Each content strand has different learning objectives for different grades or age-levels of students; however, these strands can be described generally independent of a student's level of expertise.

Computational thinking (Wing, 2009; Hu, 2011) is an approach to solving problems in a way that can be implemented with a computer. It involves the use of concepts, such as abstraction, recursion, and iteration, to process and analyze data and to create real and virtual artifacts. The *collaboration* strand refers to how CS is an intrinsically collaborative discipline; significant progress is rarely made in CS by one person working alone. Typically, computing projects involve large teams of professionals working together to design, code, test, debug, describe, and maintain software over time. Therefore, developing collaboration skills is considered an important part of these CS standards. *Computing practice and programming* refers to the use of computational tools as an important part of CS education at all levels. Although this topic is traditionally branded as "information technology," computing practice includes the ability to create digital artifacts and explore the use of programming to solve problems. The *computer and communication devices* strand teaches students how technologies like the Internet facilitate global communication and innovation. Students are introduced to devices and media that assist in their learning activities, and they learn the basic components of computers and how computer networks operate. Finally, the *community, global, and ethical impacts* strand examines the ethical use of computers and networks as a fundamental aspect of CS. Computers and networks represent a multicultural phenomenon that impacts society at all levels, raising issues of equity, access, and power. Therefore, it is important for students to understand the impact of computers on international and multicultural communication enabled through the use and distribution of technology (ACM & CSTA, 2011).

These five strands not only demonstrate the richness of computer science to students, but also help organize the subject matter for students so that they can begin to perceive CS as more engaging and relevant instead of a solitary pursuit (CSTA, 2011). Specific recommendations and

learning objectives for Level One (grades K-6), Level Two (grades 6-9), and Level Three (grades 9-12) for each of the five strands are listed in Appendix A of this thesis.

3. Computer Science Principles

Although the computer science community has labored to increase participation in K-12 education (Owens & Stephenson, 2011; Briggs & Snyder, 2012; Repenning, Webb & Ioannidou, 2010; Fincher et al., 2005; Barr & Stephenson, 2011; Buechley et al., 2008) and to create intellectually rich and engaging courses (Papert, 1980; Kafai & Resnick, 1996; Tew, Fowler & Guzdial, 2005; Ericson, Guzdial & Biggers, 2007; Malan & Leitner, 2007; Maloney et al., 2008; Furst, Isbell & Guzdial, 2007; Utting et al., 2010; Behrens et al., 2011), many students do not have the opportunity to take advantage of these courses (Arpaci-Dusseau et al., 2013; ACM & IEEE, 2013). The low participation of students, particularly non-majority groups, in the Advanced Placement Computer Science curriculum is emblematic of this problem.

The Advanced Placement (AP) Computer Science A Exam[4] is meant to be the equivalent of a first-semester college course in computer science, and its curriculum has undergone multiple revisions over the years[5]. Since 2003, the AP CS A examination has tested students on their knowledge of CS through Java[6]. Although the number of AP CS A exam-takers has increased in recent years (Ericson & Guzdial, 2014), the number of participating students for this examination is much smaller compared to other AP STEM examinations (e.g., AP Calculus had 256,163 exam-

[4] "Advanced Placement" is a protected trademark of the College Board. AP examinations are given across the United States for students to earn credit in post-secondary education.
[5] From 1984 to 1999, the AP CS exam tested students on their knowledge of Pascal; from 1999 to 2003 the exam tested students on C++. For the 2014-2015 year, AP CS A replaced the *GridWorld* case-study with a 20-hour lab requirement.
[6] From 1999 to 2009, the AP CS AB examination included all of the topics in the A exam, as well as more in-depth study of algorithms, data structures, and data abstraction. However, the AP Computer Science AB exam was discontinued in 2009 due to low numbers of students taking the AB exam.

takers in 2012, whereas CS A had 24,782 students). The participation rate of underrepresented groups in AP CS A (including female, African American, and Latina students) is also lower than other AP examinations[7]. Ericson and Guzdial (2014) argue, "If you want more AP CS A exam-takers, the focus should be on the underrepresented groups" (p. 222). Further, Mattern et al. (2011) demonstrate a strong relationship between AP exam participation and pursuing a major in a related field. Thus, AP computer science examinations have the potential to bring more students into CS as a major field of study, yet the way we teach introductory computer science, as represented by the AP CS A course, is particularly problematic for non-majority students. In response to this problem, computer science educators, supported by the National Science Foundation, have developed a new AP computer science course specifically designed to broaden participation and diversity in computer science (Astrachan & Briggs, 2012; Arpaci-Dusseau et al., 2013; Snyder et al., 2012). This effort includes motivating and educating substantially larger numbers of students (and teachers) to fulfill the demands of the market, while also reaching a broader, more diverse segment of the population (Astrachan & Briggs, 2012).

Computer Science Principles is a new AP computer science course designed to introduce students to the central ideas of computer science, to instill ideas and practices of computational thinking, promote programming literacy and creativity, and engage students in activities that show how computer science changes the world (Astrachan & Briggs, 2012; Arpaci-Dusseau et al., 2013). CS Principles is expected to compliment the AP CS A course, and is intended to help students gain competencies similar to those gained by students completing a university CS course for non-majors (Arpaci-Dusseau et al., 2013). AP CS Principles seeks to serve a more diverse population of

[7] In 2012, AP Calculus had a higher percentage of underrepresented groups take the exam than CS: for females 48.3% (Calculus) versus 18.7% (CS), for African American students 5.4% versus 4.0%, and Hispanic students 11.5% versus 7.7% (Arpaci-Dusseau et al., 2013).

students by introducing a broader set of computing principles and by putting a greater emphasis on creativity than the existing AP CS A course (Astrachan & Briggs, 2012). The College Board launched CS Principles in fall 2016; the first AP examination will be administered in May 2017.

The primary curriculum objective for the AP Computer Science Principles course is to help students learn the principles and practices of computing that are critical for the 21st century workforce. While programming is prominently featured in AP CS Principles, it is not the sole focus of the course, in contrast to the AP CS A course. CS Principles emphasizes computational thinking and algorithmic skills, engages students in the creative aspects of the field, and promotes programming literacy. CS Principles' secondary objective is to use the curriculum to serve as a catalyst for students with limited experience in computer science to pursue CS as a major field of study (Arpaci-Dusseau et al., 2013). Through both its content and pedagogy, the course aims to appeal to a broad audience. The following table summarizes differences between the AP Computer Science A and the AP Computer Science Principles courses:

AP Computer Science A	**AP Computer Science Principles**
Curriculum is focused on object-oriented programming and problem solving	Curriculum is built around fundamentals of computing including problem solving, working with data, understanding the Internet, abstraction, and programming.
Java is the designated programming language	Teachers choose the appropriate programming language(s)
Encourages skill development among students considering a career in computer science or other STEM fields	Encourages a broader participation in the study of computer science and other STEM fields, including AP Computer Science A
AP exam assessment experience: • Multiple-choice and free-response questions (written exam)	AP exam assessment experience: • Students submit digital artifacts demonstrating the skills they have developed • Multiple-choice questions (written exam)

Table 2. AP Computer Science A vs. AP Computer Science Principles Curricula

The Computer Science Principles course is organized into seven *Big Ideas* with six correlating *Computational Thinking Practices* taken from foundational computer science theory and practice. Computational thinking (Wing, 2009; Hu, 2011) captures many important aspects of the work in which computer scientists engage, including using algorithmic approaches to solve challenging problems. These practices are designed to help students coordinate and make sense of knowledge to accomplish a goal or task (Astrachan & Briggs, 2012). They enable students to engage with the course content by developing computational artifacts and analyzing data, information, or knowledge represented for computational use (Arpaci-Dusseau et al., 2013). In addition, the practices require students to learn to collaborate to build computational artifacts and communicate their purpose. Each learning objective of CS Principles directly correlates to a computational thinking practice.

AP Computer Science Principles: ***Computational Thinking Practices***
1) Connecting Computing
2) Developing Computational Artifacts
3) Abstracting
4) Analyzing Problems & Artifacts
5) Communicating
6) Collaborating

Table 3. The CS Principles Six *Computational Thinking Practices*

In contrast to the programming-focused current AP CS A course, the seven big ideas of CS Principles connect students to a curriculum that includes the art of programming, but is not

programming-centric (Astrachan & Briggs, 2012). The big ideas present the fundamental principles that are essential to succeed in future college courses, and in a variety of computing and STEM careers (Astrachan & Briggs, 2012). Emphasizing these key big ideas potentially helps students built a solid understanding and facility with computational thinking.

AP Computer Science Principles *Big Ideas*	
1) Creativity	Computing is a creative human activity that engenders innovation and exploration.
2) Abstraction	Abstraction reduces information and detail to focus on concepts relevant to understanding and solving problems.
3) Data	Data and information facilitate the creation of knowledge.
4) Algorithms	Algorithms are tools for developing and expressing solutions to computational problems.
5) Programming	Programming is a creative process that produces computational artifacts.
6) The Internet	Digital devices, systems, and the networks that interconnect them enable and foster computational approaches to solving problems.
7) Global Impact	Computing enables innovation in other fields including science, humanities, arts, medicine, engineering, law, and business.

Table 4. The CS Principles Seven *Big Ideas*

Detailed learning objectives, with evidence statements, are used to determine whether the objectives have been met (Arpaci-Dusseau et al., 2013). Thirty-five objectives are constructed by matching the seven big ideas of CS Principles with the six computational thinking practices (College Board, 2014). For example, consider Abstraction, one of the big ideas. Abstraction reduces information and detail to facilitate focus on the most relevant information. This big idea is divided into three key concepts, each of which is divided into three-to-four supporting concepts. Using the big idea and the computational thinking practice of Developing Computational Artifacts

leads to the following learning objective: The student can develop an abstraction. The following evidence statements are used to determine if a student has met this learning objective:

- Creation of an abstraction for a hardware, software, or conceptual purpose
- Use of appropriate abstractions in the creation of the artifact
- Selection of appropriate algorithmic and information- management abstractions

More details on learning objectives and evidence statements can be found by examining all thirty-five learning objectives and their associated evidence statements (College Board, 2014).

Pilot implementations of Computer Science Principles courses (e.g., Snyder et al., 2012; Arpaci-Dusseau et al., 2013), taught at both high school and collegiate levels[8] appear to have positively impacted females and underrepresented students (Arpaci-Dusseau et al., 2013), but these pilots differed in structure, breadth, and focus (Snyder et al., 2012). One possible reason for this is that the CS Principles content, represented by the framework's list of topics and objectives and its broad assertions of desired outcomes, resulted in multiple interpretations by instructors to achieve the learning outcomes (Snyder et al., 2012). The vision for CS Principles is not a packaged curriculum ready to be taught and administered, but rather a curriculum framework specified by the seven loosely defined big ideas and the six computational thinking practices. The pilot implementations reflected a broad interpretation of these big ideas and computational thinking practices, resulting in substantially different courses (Snyder et al., 2012). However, only limited data regarding the impact of the curriculum on students is available.

Pilot efforts also faced problems in reaching consensus for what the college-equivalent CS Principles target outcomes should be (Snyder et al., 2012). Introductory computer science courses,

[8] By definition, an AP course teaches content taught in college courses.

particularly for non-majors, are not often a standardized part of college curricula. In addition, CS courses (for both majors and non-majors) are taught in several variations with different content emphases, including courses taught for Electrical and Computer Engineering, Information Technology, Information Science, Human-Computer Interaction, etc. Even though most of the pilot sites reported that students found the curriculum engaging (Snyder et al., 2012), more evaluation is needed to determine whether students achieve the desired learning outcomes intended for CS Principles. More work is also needed to determine if there is a meaningful impact for non-majority populations, since broadening participation and diversity in computer science helped motivate the creation of the CS Principles course. Finally, few pilot studies have reported on whether students pursued CS as a major field of study or as part of other disciplines after their exposure to computer science through the CS Principles curriculum. Thus, prior work leaves considerable room to explore other approaches to teaching introductory computer science to diverse groups of students, and to assess how these approaches may potentially impact their motivation and facility with computer science.

4. Relevance to the Research

Although efforts have increased to develop national and state content standards for computer science (CSTA, 2011), many obstacles lie in the way of arriving at a model of K-12 CS education appropriate for all students. CSTA Standards form a basis by which educators can begin to implement a coherent CS curriculum that is available to all students, but more work needs to be done in designing, implementing, and evaluating these standards in real-world classrooms. For example, although pilot projects have tested how CS Principles could support different teaching and learning approaches, and portfolio-based assessments for the AP examination (Arpaci-Dusseau, 2013), there is limited research on whether students who have taken CS Principles

continue to pursue computing by taking additional CS courses, or if they pursue computing in other venues after course completion. In addition, we have limited understanding how different populations undertake and perform in CS Principles, despite its primary goal to broaden participation and diversity in computing.

II. GAMING: BEYOND ENTERTAINMENT

There is significant potential for digital games[9] to support education if designers and teachers take advantage of what makes games different than other media: they are interactive, engaging, and fun (NRC, 2010). Video games[10] are increasingly seen as effective tools for achieving non-entertainment purposes, because of their immersive and simulation capabilities (NRC, 2010), but also for their potential to engage and inspire students (Van Eck, 2006; Salen, 2008; Tapscott, 1998). Previous work has shown that games are most suitable in learning contexts when they exhibit the characteristics of *intrinsic motivation* (Schmitz et al., 2011) and *flow* (Csikszentmihalyi, 1990; Shernoff, 2013), which are significant for both learning and for player engagement (Gee, 2003; Squire, 2003; Squire et al., 2008; Sheldon, 2012). In short, games support learning by using play to facilitate concentration, interest, and enjoyment.

1. Play & Learning

Games in any form are an expression of *play*. Play (in the context of this research) is a form of activity with three explicit characteristics:

- it is separable from everyday life in that exists within a "magic circle" (Huizinga, 1944),

[9] Digital gaming refers to video games, computer games, mobile hand-held games, social-media games, etc.
[10] I use *video games*, *gaming*, and *games* interchangeably to denote the broader pop-cultural applications of all digital gaming spaces, since the vernacular of "video games" is often used to describe this phenomenon.

- it is safe in that it is "consequence free" and allows individuals to learn by trail-and-error without significant risk (Jewell & Loizos, 1966), and
- it is pleasurable, normatively positive, or an act of "fun" (Csikszentmihalyi, 1990).

Vygotsky (1978) analogized play to the focus of a magnifying glass, explaining that "play contains all developmental tendencies in a condensed form and is itself a major source of development" (p. 102). The natural world makes use of the principle of play as an instructional strategy. For example, lions do not learn to hunt through direct instruction but through modeling and play (Jewell & Loizos, 1966). However, play is more than a response to environmental pressures; it is an important component of the human condition because it is enjoyable in and of itself (Csikszentmihalyi, 1990).

There is a common experiential state that is present in various forms of play, and also under certain conditions in other activities that are not normally considered as play. Csikszentmihalyi (1990) refers to this experiential sate as "flow." *Flow* is a conscientious state of deep absorption in an activity that is "so gratifying that people are willing to do it for its own sake, with little concern for what they will get out of it, even when it is difficult or dangerous" (Csikszentmihalyi, 1990). Csikszentmihalyi's research found that optimal experience, or flow, is experienced the same way around the world—seemingly dissimilar activities are described in similar ways when they are being enjoyed, and that enjoyment is the same, regardless of culture, social class, age, or gender (Cowley et al., 2008).

Play inherently has purpose; yet, play has often been thought of as being frivolous, similar to how digital gaming is often considered to be superficial and meaningless (Malaby, 2007). As Vygotsky (1978) stated, "It is incorrect to conceive of play as activity without purpose…creating an imaginary situation can be regarded as a means of development abstract thought" (p. 103).

Most flow experiences occur with activities that are goal-directed, bounded by rules, and require mental energy and appropriate skills (Cowley et al., 2008). This has led to work on how games promote and facilitate flow for players (Salen, 2008; Cowley et al., 2008).

There is a measurable correlation between gameplay and flow (Cowley et al., 2008). In the spirit of computational thinking, a game is a subclass of play and inherits from *play* the characteristic of *flow*. An individual experiencing flow perceives her performance or activity to be pleasurable and is perceived as worth doing for its own sake, even if no further goal is reached. Individuals function at their fullest capacity, and the experience itself becomes its own reward. During flow, people become so intensely involved in an activity that nothing else seems to matter and they often lose a sense of time. They are in a state of enjoyment because they have situated themselves in an optimal learning environment (Csikszentmihalyi, 2003). Similarly, game players voluntarily invest countless hours in developing their problem-solving skills within the context of games (Gee, 2008). They recognize the value of extended practice, and develop personal qualities such as persistence, creativity, and resilience through extended play (McGonigal, 2011).

Effective game design often seeks to create the prerequisites of flow: games must have some type of goal for the player to pursue (whether the goal is self-invoked or invoked via the design black-box); that concentration is possible because the task provides immediate feedback; the ability to exercise a sense of control over actions; a deep but effortless involvement that removes awareness of the frustrations of everyday life; concern for self disappears, sense of self emerges stronger afterwards; and the sense of the duration of time is altered (Cowley et al., 2008; Koster, 2005; Dignan, 2011). Flow is experienced through the process of users pursuing a goal, provides users with immediate feedback, is often involved within the play of a magic circle, and

helps retain interest and concentration for users. In short, flow is a critical component for effective gameplay.

Flow has also been empirically related to learning, talent-development, academic achievement, and creative accomplishment (Csikszentmihalyi, 1996; Shernoff et al., 2003). Flow promotes a simultaneous occurrence of elevated *concentration*, *interest*, and *enjoyment*; experiences that are directly related to effective learning (Shernoff, 2013). Concentration, which is central to flow (Csikszentmihalyi, 1990), is related to meaningful learning (Montessori, 1967) and academic performance (Corno & Mandinach, 1983). Interest directs attention, reflects intrinsic motivation, stimulates the desire to continue engagement in an activity, and is related to school achievement (Schiefele, Krapp, & Winteler, 1992). Enjoyment is a positive feeling related to the demonstration of competencies, creative accomplishment, and school performance (Csikszentmihalyi, Rathunde, & Whalen, 1993). Flow is thus a critical component for effective and motivated learning—whether in a game or in a classroom.

2. Intrinsic and Extrinsic Motivation in Gaming

To be motivated means *to be moved* by something (Ryan & Deci, 2000). A person who feels inspired to act is thus characterized as "motivated," and is energized toward an action or an end. Self-Determination Theory, discussed in more detail in Chapter 3, distinguishes between types of motivation based on the different reasons or goals that give rise to an action. The most basic distinction is between *intrinsic motivation*, which refers to doing something because it is inherently enjoyable or interesting, and *extrinsic motivation*, which refers to doing something because it leads to a desired outcome (Ryan & Deci, 2000). In order for a video game to be successful in motivating the player, the game must use a combination of intrinsic and extrinsic motivations (Dignan, 2011; Bartle, 2004).

Even though games are generally considered to be overall enjoyable experiences, games do not only evoke positive emotions. For example, players may experience frustration from difficulty, disappointment from failure, resentment against opponents, and so on. Although there are cases of masochistic motivations[11] for playing games, or a pathological reason such as addiction[12], games are enjoyable because of intrinsic and extrinsic motivations. As game developer Bartle (2004) points out, in order for a game to be successful, the players must expect to get something out of their experience" (p. 128). Gaming motivations can extend outside of the system black-box into other socially and culturally discursive experiences because of both intrinsic and extrinsic motivations (Behnke, 2011).

3. Serious Games in Education

Gaming technologies increasingly transcend the traditional boundaries of their medium (Barab et al., 2010), as evidenced by the growth of *serious games* as an industry and research field (NRC, 2010; Sanchez & Smith, 2007; Raybourn et al., 2005; Castronova, 2005; Michael & Chen, 2005; WTEC, 2009; Kafai et al., 2008;). Serious games combine entertaining "game" elements with "serious objectives" other than pure entertainment for purposes of education, corporate and military training, policy making, political and social activism, medicine and mental health, etc. (Kafai et al., 2008; Michael & Chen, 2005). The intention of a serious game is to create "purpose-driven playful environments intended to impact the players beyond the self-contained aim of the game" (Mitgutsch & Alvarado, 2012, p. 122). Bogost (2011) explains how the term "serious" has a specific rhetorical purpose to earn the support of high-level governmental and corporate officials, individuals for whom "game" implies trivial entertainment, even if that trivialization is precisely

[11] One of the most popular video games franchises is *Dark Souls* (FromSoftware), which became popular in large part because of its overwhelming difficulty and unforgiving "hard-core" style of gameplay.
[12] Video game addiction treatment and rehabilitation clinics exist in the United States, China, and Europe.

part of games' power. Using the word "serious" has given gaming advocates a way to frame the uses of games in governmental and industrial contexts by making the claim that games can tackle consequential topics and provide profound results (Bogost, 2011).

There have been numerous efforts to translate the engagement found in games into education (Ritzhaupt, 2009; Linhoff & Settle, 2009; Barnes, Richter & Powell, 2007; Eagle & Barnes, 2009; Cooper, Dann, & Pausch, 2003; Malan & Leitner, 2007; Basawapatna, Koh & Repenning, 2010). Many argue that games embody beneficial learning principles (Malone, 1980a, 1980b; Shaffer, 2008; Collins & Halverson, 2009; Gee, 2003; Squire, 2011, 2003) by promoting extrinsic and intrinsic motivations (Schmitz et al., 2011), allowing the freedom to explore and fail (Mitgutsch, 2012; Malaby, 2007), and fostering effective problem-solving skills (Squire et al., 2008; Gee, 2003; Bogost, 2007; Khaled et al., 2009).

4. Gamification

Gamification attempts to harness the motivational power (both intrinsic and extrinsic) of game mechanics and meaningfully apply them into real-world situations such as education, health, and public policy (Lee & Hammer, 2011). Gamification has been increasingly the subject of debate by game scholars and game designers alike. John Ferrara (2012) argues that user-experience designers, if they want to improve user-engagement through game mechanics, should start with a game *first*, not add superficial gaming mechanics to an existing interface. Some argue that by putting the term "game" first, it implies that the entire activity will become an engaging experience through extrinsic motivations such as points and badges (Nicholson, 2012), when, in application, scoring systems in games are the least interesting and provoking aspect to a game (Bogost, 2011; Ferrara, 2012). Bogost (2011) labels this extraction of gaming elements to non-game contexts as "exploitationware," in that these systems mistake games' secondary properties—

points, badges, leveling-up, etc., —as the primary properties, leaving the truly transformative and meaningful components of gaming negligible. As Bogost (2011) argues:

> "[Gamification] confuses the magical magnetism of games for simplistic compulsion meted out toward extrinsic incentives. It fails to embrace the complex responsiveness of 'real' games, games that make solutions seem interestingly hard rather than tediously so." (p. 2)

Bogost (2011) asserts that gamification as "exploitationware" is a cursory and crass marketing scheme employed by corporations and advertisers in an effort to improve productivity and sales, losing sight of the larger and broader impact of persuasive games and their meaningful functions within society.

The difference of perspectives regarding gamification as exploitationware verses gamification as meaningful engagement arises in large part from the motivation for the use of game design elements in a given experiential context (Decker & Lawley, 2013). Motivational theory research has shown that extrinsic rewards can have a reversed and damaging effect on individual motivation to participate in activities (Lepper, Greene & Nisbett, 1973). These rewards can undermine the intrinsic sense of satisfaction that engaging in creative and productive work can generate. However, when these techniques are used appropriately, game elements can reinforce intrinsic satisfaction and reward rather than replacing them (Decker & Lawley, 2013). In summary, gamification remains an unclear, contentious area of research and game design (Bartle, 1996; Chen & Jang, 2010).

5. Gamification Techniques

There is an apparently good fit between education and gamification—when it is designed effectively and implemented appropriately. One of gamification's core design principles is to

"provide feedback so that players can achieve a sense of mastery" (Richards et al., 2014). Mastering an activity, and learning through failure, are often represented in gamification in the form of achievements, badges, high leaderboard rank, reaching the maximum level, etc. (Richards et al., 2014). Games maintain this positive relationship with failure by making feedback cycles rapid and keeping the stakes low—players can keep trying until they succeed and they risk very little by doing so. Gamification can shorten feedback cycles, give learners low-stakes ways to assess their own skills and capabilities, and create an environment in which effort in learning is rewarded. Students, in turn, can learn to see failure as a learning opportunity. This may be particularly useful in teaching introductory computer science to students with limited exposure or no previous experience in CS. Gamification, when coupled with well-designed CS pedagogy, may provide students an engaging and effective way to learn introductory computer science.

Previous work had identified four underlying dynamics of gamification that have been shown to be successful in learning applications: the freedom to fail, rapid feedback, player progression, and player-defined goals (Miller et al., 1999; Hamari et al., 2011; Gee, 2003; Squire et al., 2008; Sheldon, 2012; Nicholson, 2012; Decker & Lawley, 2013; Huotari & Hamari, 2012; Deterding & Dixon, 2011; Deterding et al., 2011). Kapp, Blair, and Mesch (2014) distinguish between two types of gamification that support the freedom to fail, feedback, progression, and goals: structural gamification and content gamification.

Structural gamification is the "application of game elements to propel a learner through content with no alteration or changes to the content" (p. 55). The learning content does not become game-like, but the structure around the content incorporates gaming elements. The primary focus for this type of gamification is to motivate learners to go through the content and to engage them in the process of learning through rewards. The most common elements in this type of

gamification are points, badges, achievements, levels, a leaderboard, methods for tracking learning process, as well as social components where learners can share accomplishments with others (Kapp, Blair & Mesch, 2014). An example of applied structural gamification would be a learner gaining points within a course for watching a video or completing an assignment when the assignment or video had no game elements associated with it, other than the fact that the learner received points. The use of structural gamification provides a number of affordances to help learners to gain knowledge, skills, and abilities they need while simultaneously allowing them to have control over when they learn and how the decide to approach the learning process (Nicholson, 2012; Kapp, Blair & Mesch, 2014; Huotari & Hamari, 2012; Deterding & Dixon, 2011). Kapp, Blair, and Mesch (2014) argue that establishing clear goals to the user, adding incremental goals and rewards, a sense of learner progression, and transparent rules promotes effective structural gamification.

Content gamification is the "application of game elements and game thinking to alter content to make it more game-like" (p. 55). The most common content gamification elements are story, challenge, curiosity, character, interactivity, feedback, freedom to fail (Kapp, Blair & Mesch, 2014). An example of content gamification in use would be adding story elements to a course or starting a course with a challenge instead of a list of objectives. Adding these elements makes the learning content more game-like but does not necessarily turn the content into a game; it provides context or activities that are used within games and adds them to the content being taught (Kapp, Blair & Mesch, 2014). In content gamification, challenge plays a big role in engaging leaners. Previous work indicates that challenge can be a strong motivator in learning (Shernoff, 2013) and is also a core prerequisite for flow (Csikszentmihalyi, 1990). Two of the advantages video games have over traditional learning environments is the frequency of feedback

and the freedom to fail. Providing regular feedback to learners in the form of self-paced exercises, visual cues, and progress bars can be effective (Kapp, Blair & Mesch, 2014). In addition, content gamification allows learners the freedom to fail, which encourages them to explore content, take chances with their decision-making, and to be exposed to consequences in the "safe" environment of a "magic circle" (Nicholson, 2012). The risk of failure without punishment can be engaging because learners can explore and examine causes and effects to content if it is safe to fail and learn from their mistakes (Deterding et al., 2011; Kapp, Blair & Mesch, 2014).

6. Relevance to the Research

This thesis examines how gamification can foster meaningful learning and engagement in the context of into introductory computer science education. Students learning introductory computer science often report a lack of engagement with the course content (AAUW, 2010; Margolis & Fisher, 2002). Since many young people are actively engaged in gaming spaces (Van Eck, 2006; Kafai et al., 2008; ESA, 2014) and video games themselves are computational artifacts, gaming offers a potentially promising approach to increase student interest and engagement for learning computer science. Specifically, gamification may provide a way to promote student motivation and engagement while also providing feedback on the student's level of competency of the learned material.

One of the reasons that gaming frameworks are potentially effective for CS education is that successful learning techniques are already conventional practice in commercial game design (Schmitz et al., 2011; Sheldon, 2012; Gee, 2003; Squire, 2003; Squire et al., 2008). Games embody powerful principles of learning that educators can emulate (Malone, 1980a, 1980b; Shaffer, 2008; Collins & Halverson, 2009; Gee, 2003; Squire, 2011) because games "teach" concepts by modeling objectives and requiring problem-solving, calling attention to key features

of problems through gameplay cues, and structuring problems so that players build upon previous understandings (Malone & Lepper, 1987; Gee, 2003; Prensky, 2003; Squire, 2011; Dickey, 2005). Modeling for problem-solving, and abstracting information to build upon previous knowledge, map well with computational thinking practices.

CHAPTER III
THEORETICAL FRAMEWORK

The primary goal of this research is to examine the value of gamification in introductory computer science education. In order to conduct this work, a theoretical lens is needed that brings into focus the disparate ideas of effective computer science pedagogy and motivational game design techniques, including flow, the role of failure in learning, and intrinsic and extrinsic motivations. The conceptual framework needs to (1) frame the pedagogical outcomes that are desirable by the CS education community, and (2) use a theory that is appropriate for the design and implementation of effective game design techniques. As explained below, the conceptual and theoretical framework chosen for this work is Self-Determination Theory (Ryan & Deci, 1991). Self-Determination Theory (SDT) has been used extensively by game designers and educators to represent the psychological needs of users. Below, I discuss SDT and the relevance of motivation research to game design and instructional design techniques.

A. SELF-DETERMINATION THEORY

Self-Determination Theory is an empirically derived theory of human motivation that postulates there are three universal psychological needs that are essential for optimal human development and functioning—competency, autonomy, and relatedness (Ryan & Deci, 2000). SDT argues that students learn better and are more creative and innovative when they feel competent, autonomous, and related to something bigger than themselves (Harlow, Harlow & Meyer, 1950; Deci, 1971; Ryan & Powelson, 1991; Ryan & Deci, 2000). Similarly, games motivate users and support effective learning engagement because they employ SDT functions. As Ryan, Rigby, and Przybylski (2006) showed, "games are primarily motivating to the extent that

players experience autonomy, competence and relatedness while playing" (p. 345). There is a rich body of prior work that underlies SDT, making it an appropriate conceptual framework for using game elements to facilitate introductory computer science education. This relationship is discussed further in the remainder of this chapter.

1. Brief Summary of Prior Work

Daniel Pink (2009) summarizes decades of SDT scholarship by stating that the core of all human endeavor is "our innate need to direct or own lives, to learn and create new things, and to do better by ourselves and our world" (p. 10). Wagner (2012) and Brown (2012) argue that personal motivation, self-engagement, and computational thinking foster innovation that is needed for the twenty-first century workforce. SDT can thus inform the structure and design of gamified educational content to make it more motivating, effective, and meaningful for students. Previous work in SDT has shown that people learn more effectively and creatively when they feel competent, autonomous, and related to something bigger than themselves (Ryan & Deci, 2000)., providing a convenient way to use autonomy, competency, and relatedness to build upon other learning theories: Self-Directed Learning (Knowles, 1975), Constructionism (Papert & Harel, 1991), and Connectivism (Siemens, 2005).

1.1. Self-Directed Learning—Autonomy

Autonomy is our innate desire to be self-directed. Autonomy within SDT concerns a sense of volition or willingness when doing a task (Ryan & Deci, 2000; Ryan & Powelson, 1991; Locke & Latham, 2002). When activities are done for interest or personal value, perceived autonomy is high. Ryan, Rigby, and Przybylski (2006) argue that, "Provisions for choice, use of rewards as informational feedback (rather than to control behavior), and non-controlling instructions [in games] have all been shown to enhance autonomy and, in turn, intrinsic motivation" (p 346). SDT

thus shows how games can enhance autonomy when game elements provide player flexibility, choices in how to select tasks and complete goals, and use reward structures to provide meaningful feedback to the player (Ryan, Rigby & Przybylski, 2006). Self-Directed Learning (Knowles, 1975; Brockett & Hiemstra, 1991) may also inform ways to design autonomous gamification into pedagogy. According to Knowles (1975) and Hiemstra (1994), self-directed learning requires that:

1) students take initiative and responsibility for their own learning;
2) students select, manage, and assess their own learning activities;
3) motivation and volition are critical for success;
4) independence in setting goals and defining what is worthwhile to learn;
5) teachers provide scaffolding, mentoring, and advising;
6) peers provide collaboration to facilitate the group's learning.

In this research, self-directed learning is uniquely instantiated within each learning context as described in more detail in Chapters Five through Eight. In brief, the self-directed components attempt to implement these six principles into the curriculum and content design. Students are encouraged to (1) *take initiative* regarding the specific content on which to focus their efforts so that they can (2) take responsibility for their learning by *selecting and assessing* their own learning activities, which can potentially (3) sustain their *motivation* and *volition* because of the emphasis on individual creative interests, and (4) by *setting their own goals*, which are framed carefully in the learning environment in order to (5) help *scaffold* these objectives and skills for students, so that they can (6) create peer-to-peer *collaboration* within their learning communities.

Miller et al. (1999) found that goal-oriented tasks were less effective than non-goal learning objectives for teaching physics through a game. Players perceived a loss of autonomy when given

goal-oriented tasks, rather than developing their own goals through a self-directing process. Thus, gamified instructional design must carefully consider the tasks that frame the game if the gaming framework is to lead to the desired learning outcomes. These learning frameworks need to be mindful of not "teaching the test" but rather teaching the process through specific game mechanics incorporating self-directed tasks and goals (Miller et al., 1999). In this thesis, curriculum is built around these theories in order to teach the process of computational thinking by encouraging and facilitating "self-directed autonomy" through the gamification process.

1.2. Constructionism—Mastery

In SDT, competency refers to our innate desire to gain *mastery*[13] of an activity over time, i.e., to get better and better at what we do (Ryan & Deci, 2000; Csikszentmihalyi, 1990; Ryan & Powelson, 1991; Pink, 2009). SDT further contends that mastery, or the process of gaining competency in self-ability, is essential for education and motivation (Ryan & Deci, 2000; Ryan & Powelson, 1991). Opportunities to acquire new skills or abilities, to be optimally challenged, and to receiving positive feedback enhances perceived competence, and, in turn, intrinsic motivation (Ryan, Rigby & Przybylski, 2006).

Flow is a necessary but insufficient component of mastery (Pink, 2009). Flow happens in the moment (Csikszentmihalyi, 1990), whereas mastery unfolds over time (Duckworth et al., 2007). Csikszentmihalyi (1975; 1990) found that the most satisfying experiences in people's lives occur when goals are clear (but selected autonomously), the relationship between a person's ability and the task at hand remains just beyond current abilities, and feedback on one's ability is adequately accessed and communicated. In moments of flow, people are autonomously engaged

[13] I use Pink's (2009) nomenclature of "mastery" over Ryan & Deci's (2000) "competency"; mastery underscores an ongoing process of learning, whereas competency suggests a completed state of experience or being. *Mastery* also underscores play discourse within gaming culture.

while challenged in ability (Pink, 2009). The reciprocal balance between flow and challenge is often considered a prerequisite for effective game design (Cowley et al., 2008; Gee, 2006; Squire, 2011). Other work demonstrates that gaming enhances player's perceived competency when game controls are intuitive and readily mastered, when tasks within the game provide ongoing optimal challenges, and provide quick opportunities for positive feedback (Ryan, Rigby & Przybylski, 2006; Cowly et al., 2008). Mastery, the process of becoming more competent in self-ability, is essential to well-being and optimal development. As Ryan and Powelson (1991) summarize, "The pleasure in mastery, in effectance, in assimilating, in experiencing action merely for its own sake is, as Piaget (1952) once called it, a basic fact of psychic life" (p. 52).

Prior work in SDT research has shown that optimal learning environments combine academic challenge with positive emotional response, which makes learning both playfully intense, spontaneous and important (Anderson, 2005; Rathunde & Csikszentmihalyi, 2005; Shernoff, 2013; Turner & Meyer, 2004). Recent work observes that students appear to be meaningfully engaged in learning activities that are structured more like non-academic classes (Shernoff et al., 2003) and after-school enrichment activities (Shernoff & Csikszentmihalyi, 2009). This work leads educators to employ "learning by doing" pedagogy, particularly through the lens of constructionism. *Constructionism* (Papert & Harel, 1991), instantiated from constructivist theory (Papert, 1971; Papert, 1980), posits that learners construct mental models to understand the world around them. In contrast to constructivism, constructionism argues that learning happens most effectively when students are also active in making tangible artifacts in the real world to reflect their understanding of the learned material. Seymour Papert (1991) discusses the theoretical difference between constructionism versus constructivism and why this difference remains significant. He argues that,

> "Constructionism—the N word as opposed to the V word— shares constructivism's view of learning as 'building knowledge structures' through progressive internalization of actions… It then adds the idea that this happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe." (Papert, 1991, p.1)

Constructionism has been used to facilitate STEM learning (Papert & Harel, 1991). Game design has also been used as a constructionist methodology to teach computer science (Barnes et al., 2007; Bayliss & Strout, 2006; Treanor et al., 2012; Linhoff & Settle, 2009). Thus, game design can incorporate computer science, and the construction of digital games can teach foundational CS topics. As Gee (2008) argues, "Game design is not accidently related to learning, but rather that learning is integral to it" (p. 24). By actively constructing tangible artifacts, students learn the material through mastery and persistence. Brown (2012) argues that the recursive process of ideation, experimentalism, and implementation fosters innovation. Constructionism facilitates computational thinking (Brown, 2012) and computational artifacts through the recursive learning (Mitgutsch, 2012) of mastering a challenge.

CSTA Standards incorporate constructionism when students actively construct models, computational artifacts, and computer programs. I use constructionism in this model to facilitate "learning-by-making" (Kafai & Resnick, 1996) in an effort to make computer science more personally engaging, motivating, and meaningful for introductory students (Malan & Leitner, 2007; Maloney et al., 2010; Arpaci-Dusseau, 2013). This appears to be an effective approach, since Brown (2012) and Wagner (2012) argue that innovation-creation and computational thinking can only be effectively learned within iterative constructionism. Having the desire to do something

because one finds it deeply satisfying and personally challenging inspires the highest levels of creativity, whether it is in the arts, sciences, or business (Amabile, 1996).

Mastery of an activity is challenging, hard work (Duckworth et al., 2007; Dweck, 2006), but gratifying through induced flow (Csikszentmihalyi, 1990). Mastery creates the sense of accomplishment and efficacy, which is facilitated by the exercise of one's capacities under conditions of optimal challenge (Ryan & Powelson, 1991). As Mitgutsch (2012) succinctly states:

> "…Learning [in games] is understood as an experience-based recursive form of learning through irritations, confrontations, failure, disappointment, and disillusionment… Playing is the voluntary attempt to confront ourselves with unnecessary challenges in a satisfying way… The players learn recursively through failing and dealing with confrontations in the game." (p. 578-580)

Game elements frequently use challenge as a means to teach players new abilities or knowledge. These elements promote flow by allowing players to overcome failures and master the material, resulting in both satisfying play and effective learning (Mitgutsch, 2012).

1.3. Connectivism—Purpose

Relatedness to community, or *purpose*[14], is essential for optimal human development and well-being (Ryan & Deci, 2000; Dunn, Ankin & Norton, 2008; Pink, 2009). Relatedness and purpose concern the emotional and personal bonds between individuals. It reflects our strivings for contact, support, and community with others (Ryan & Powelson, 1991). However, purpose implies more than mere connection; it refers to the experience of connecting with others in ways that conduce toward well-being and self-cohesion for all individuals involved (Ryan & Powelson,

[14] I use "purpose" as a term for Ryan and Deci's (2000) "relatedness"—*purpose* can refer to multifaceted forms of relatedness, rather than just interpersonal connection.

1991). Purposeful needs are not antithetical to autonomy or mastery; in fact, one often feels most related and purposefully driven to those who are responsive to one's autonomous expressions (Ryan & Deci, 2000; Ryan & Powelson, 1991). As Csikszentmihalyi (1997) lamented, "One cannot lead a life that is truly excellent without feeling that one belongs to something greater and more permanent than oneself" (p. 131). Thus, purpose needs to play an increasingly important role within educational contexts (Ryan, Connell & Grolnick, 1993).

Ryan and Powelson (1991) argue that a learner is most likely to be interested, engaged, and volitional in contexts of learning characterized by autonomy and purpose. The kinds of needs that are expressed by students (and that can energize their involvement in learning) predominately consist of an interpersonal nature (Ryan & Deci, 2000; Ryan & Powelson, 1991; Vygotsky, 1978; Papert, 1971). Where students experience support of their autonomy, and when they feel connected to and supported by others, students are more likely to be highly motivated in the learning process (Ryan & Powelson, 1991). By contrast, in contexts that are controlling, and when students feel disconnected from others, alienation and disengagement are likely outcomes (Ryan & Deci, 2000). Students are engaged and motivated in domains where they feel purposefully connected (Ryan & Powelson, 1991), which gives rise to learning theories that enable learning experiences to be conducive toward interrelation and purpose.

Connectivism (Siemens, 2005) guides the development of learning materials for the networked world. Connectivist theory argues that individuals learn and work in a networked environment (Siemens, 2005), and as a result, learners do not have control over what they learn since others in the network continually and actively change information, requiring new learning, the unlearning of old information, and relearning current information (Ally, 2008). However, this "loss of control" does not indicate a loss of autonomy for the learner—rather, connectivists

advocate for enabling learners to become autonomous and independent so that they can retroactively acquire information to build a valid, accurate, and current knowledge base (Ally, 2008). Some of the key intellectual ideas behind innovative digital learning includes connectivism (learning is the connection of people to knowledge), distributed representation (knowledge exists in the network of learners) and negotiated meaning (the meaning of knowledge is negotiated in the network) (Downes, 2006; Downes, 2012). Connectivism is fundamentally driven by individuals creating purpose and communal meaning within a network.

Social interaction and online communities are an important component of meaningful gaming. These interpersonal connections provide additional value to the game space in many ways: fans can purchase additional game-related items, find game updates and modifications, use tutorials and walkthroughs to aid their play experience, post requests to the designers for game fixes and extensions, and interact and communicate with other players (Williams, 2006; Ducheneaut, Yee & Nickell, 2006; Yee, 2006). These connectivist phenomena in gaming promote and sustain autonomy; many games use planned, online communities distributed by the company or game designers. In other cases online communities have emerged organically, without the designer's intention (Sweetser & Johnson, 2004). Regardless whether online communities are explicitly or implicitly designed, games facilitate connectivism through collaborative and confrontational experiences (Yee, 2006). Therefore, gaming communities increasingly play an important role for connectivist motivations of gameplay and connectivism represents an essential game element and learning affordance.

In summary, under conditions conducive to autonomy, mastery, and purpose, people will be likely to express their inherent tendency to learn, to do, and to grow. People are engaged and motivated in domains where their basic psychological needs can be fulfilled (Ryan & Powelson,

1991). These interpersonal processes are fundamental to learning and personal motivation. Self-directed autonomy, constructing mastery, and connectivist purpose thus inform the design and application of gamification into introductory computer science education.

2. Relevance to the Research

My research uses gamification to facilitate Self-Directed Learning, Constructionism, and Connectivism into introductory computer science. In these educational frameworks, learners set their own goals (implicitly or explicitly) in order to achieve the desired outcome of the facilitated gamification environment. Creating one's own goals underscores the components of self-directed learning thereby sustaining autonomy. As Nicholson (2012) stated:

> "Gamification systems need to either allow different ways for users to achieve goals so that users can be involved in the ways most meaningful to them or to allow users to set their own goals and achievements." (p. 3)

If students are provided ways to demonstrate mastery in only one way, those students who can demonstrate the desired learning outcome in a different way than what is measured will be excluded. Meaningful gamification for computer science education therefore requires the implementation of autonomous self-directed learning through user-centered design practices.

Another core affordance of gamification is to "provide feedback so that players can achieve a sense of mastery" (Richards et al., 2014). Gamification supports student mastery by providing feedback on competency through badges, points, leaderboard rank, etc. Mastering an activity or skill requires "gritty growth" (Dweck, 2006; Duckworth et al., 2007) i.e., learning repeatedly through failure. Gamification provides feedback on the triumph over failure through earned achievements and badges, high leaderboard rank, maxing a "level-cap", etc. Although these

mechanics are often considered as solely extrinsic motivations (Bogost, 2011), such game elements represent a far more meaningful type of engagement—overcoming failure and gaining mastery. For example, in many games, the only way to learn how to play the game is to fail at it repeatedly and learn from each failure (Gee, 2008). Gamification can provide quick feedback cycles, give learners low-stakes ways to assess their own capabilities, and create an environment in which effort and flow, the path toward mastery, is rewarded. Students, in turn, can learn to see failure as an opportunity, rather than becoming helpless, fearful or overwhelmed. Therefore, meaningful gamification may support Constructionist learning by recognizing effort and failure of the making-activity as feedback for the path toward mastery.

Gamification also incorporates Connectivist approaches, in that many gamification frameworks create affordances of competition and collaboration with other members in that community of practice (Nicholson, 2012). Leaderboards, guilds, groups, and other collaboration strategies in games can bring together large, like-minded communities to promote a sense of interrelated purpose (Richards et al., 2014; Ducheneaut et al., 2007). This is one reason why many gamification systems frequently incorporate leaderboards and user ranking mechanisms (Huotari & Hamari, 2012). Gamification's inherent design supports and facilitates connectivist networks, which may facilitate purpose into the implemented learning environment.

This research employs autonomy through self-directed learning, mastery through constructionism, and purposeful connectivism. Together, these frameworks provide a theoretical perspective for the incorporation of meaningful gamification into introductory computer science pedagogy.

CHAPTER IV
RESEARCH DESIGN & METHODOLOGY

This chapter presents the research design and methodology of my thesis research, and discusses the methodological choices made to conduct and evaluate this work.

I. RESEARCH STRATEGY

The research seeks to investigate whether gamification can contribute to student motivation and success in introductory computer science in both formal and informal educational settings.

1. Research Questions

Previous work suggests that gamification can potentially influence student engagement for learning when designed effectively (Kim & Lee, 2013). This research extends prior work by examining how gamification in both traditional and alternative educational environments impacts student learning and motivation in different introductory computer science venues. Underlying this assessment is the following research questions:

RQ1: *Does the introduction of game elements into an entry-level computer science course improve student motivation?*

RQ2: *Does the introduction of game elements into an entry-level computer science course improve student learning?*

RQ3: *Do these results differ across differing student populations?*

To answer these research questions, I introduced gamification techniques (e.g., the use of badges, points, leaderboards, etc.) into four introductory computer learning environments with varying formality. I evaluated the results of these experiments using a mixed methods approach that

examined both quantitative and qualitative data. I then examined whether results differed across student demographics, gender, etc., and if so, the way in which the results differed.

Each research setting implemented a variant of the CSTA student learning objectives and included gamification using situated-design (Greenbaum & Kyng, 1992) techniques for each site's instructional design objectives. I employed gamification within an introductory CS undergraduate course, in an after-school program for high school Latinas, in a public library workshop for elementary and middle school children. I also evaluated three online CS tutorial websites that used gaming and non-gaming elements as part of that site's training program.

2. Research Hypothesis

I hypothesize that the introduction of game elements will increase student motivation and improve student completion of learning objectives. Formally, the null and alternative hypothesis are as follows:

Null Hypothesis H_0**:** Gamification has no effect on student motivation and intended learning outcomes.
Alternative Hypothesis H_1**:** Gamification has an effect on student motivation and intended learning outcomes.

Table 5. Null and Alternative Hypothesis

I tested this hypothesis under the rubric of Self-Determination Theory (SDT). SDT asserts there are three universal psychological needs that are essential for effective learning: competency, autonomy, and relatedness (Ryan & Deci, 2000). SDT is an appropriate rubric because prior work has shown that "games are primarily motivating to the extent that players experience autonomy, competence and relatedness while playing" (Ryan, Rigby, and Przbylski, 2006, p. 345). Thus,

meaningful gamification (Nicholson, 2012), or the integration of user-centered game design elements into non-game contexts, appears to match well with the tenants of SDT. In particular, I have selected for implementation game elements intended to emphasize autonomy, competency, and relatedness. As described in further detail in Chapter Three, the game elements selected for implementation included the use of quests, avatars, badges, experience points, leaderboards, guilds, and progress bars.

3. The Role of Case Study

The strategy of inquiry most appropriate to the central research questions is the *case study*. As Stake (1995) observes:

> "Case studies, in which the researcher explores in depth a program, an event, an activity, a process, or one or more individuals. The cases are bounded by time and activity, and researchers collect detailed information using a variety of data collection procedures over a sustained period of time." (p. 15)

Each research setting that I investigate is "bounded by time and activity" and uses "a variety of data collection procedures over a sustained period of time." The case study creates opportunity to investigate each bounded activity in its own situated context. For example, the academic expectations for an undergraduate college course are much different than those of learning activities for children in a public library—yet both explore the application of gamification in entry-level CS pedagogy.

I conducted four case studies in an effort to learn more about the ways that gamification affects student motivation and achievement of CS learning outcomes. The cases are *singular* (one particular entity event at a specific time), and *instrumental*, or a specific case that is representative

of a more general phenomenon (students learning introductory computer science) (Stake, 1995). In instrumental case studies, the research focus of the study is more likely to be known in advance and designed around established theories, methods, or research questions (Creswell, 2003). I used the instrumental approach for each singular case by framing the studies around the research questions and the chosen theoretical approach to student motivation and learning (SDT).

I investigated multiple cases and treated each case as a single case, so that each case's conclusions could be used as information contributing to the whole study (Stake, 1995). I employed this approach because it allowed me to consider contextually sensitive designs for each introductory CS learning environment, while providing data to help answer the central research questions across varying learning environments using a variety of methods. Previous work in introductory computer science education reflects awareness that the pedagogy is imbedded within iterative design processes that may undergo many revision cycles (Arpaci-Dusseau et al., 2013; CSTA, 2011). In addition, game elements are conceptually dependent upon the context and instructional design objectives for which they are implemented (Miller et al., 1999; Sheldon, 2012). Therefore, the case study is an appropriate research strategy to conduct this work.

4. Summary of Research Design

I was particularly interested in how females and minority groups were impacted and motivated by the gamification of introductory computer science. As described in Chapter 1, these four case studies include (1) an introductory undergraduate CS course intended predominately for non-majors, (2) an extracurricular CS program for high school girls, and (3) a public library CS workshop series for elementary and early middle school children and (4) online tutorial websites that use variations of gamification and game-based learning environments.

II. THEORETICAL DESIGN

1. CS Learning Outcomes

The CSTA Standards are used to structure the learning framework for each of the four research investigations. I drew from the so-called CSTA Standards' Levels and Content Strands to create learning objectives that are appropriately situated for each research site. According to the College Board (2014) and CSTA (2011) Standards, computational thinking practices for entry-level CS are designed to:

- Help students coordinate and make sense of knowledge to *accomplish a goal or task*;
- Enable students to engage with the course content by *developing computational artifacts* and *analyzing* data, information, or *knowledge* represented for computational use;
- Require students to learn to *collaborate* and to *build* computational artifacts to communicate their *purpose*.

Table 6 summarizes the intended pedagogical outcomes for each case study according to the Levels of CSTA Standards.

Level One	Level Two	Level Three
Case Study 3: Coder Dojo • Grades K-6	Case Study 2: Technovation • Grades 9-12	Case Study 1: Virtual Worlds • Undergraduate Case Study 4: Online Tutorials • Undergraduate

Table 6. Summary of CSTA Standards Curriculum Levels for each Case Study

Appendix A provides details of each Strand for each level of the curriculum. For example, AP CS Principles, a course intended for college-ready students, falls under Level Three of the CSTA Standards. Participants in Case Study Four are also undergraduate students, and concepts they are exposed to via the Online Tutorials are considered "core" learning objectives for introductory undergraduate CS courses, including variables, operators, conditionals, arrays, etc. The learning objectives for each case study are discussed in detail in Chapters 5 – 8.

2. Game Design Techniques

Game elements that incorporated the SDT characteristics of autonomy, competency, and relatedness were selected across all case studies[15]. Although Deterding et al. (2011) caution against explicit delimitations of "gamification" or "game elements," other prior work highlights four underlying game design principles that are more consistently successful than others when applied to learning environments (Stott & Neustaedter, 2013). These design elements include:

[15] I did not select the game elements incorporated into Case Study 4. In this instance, I evaluated the effectiveness of elements already present on the chosen websites.

- the freedom to explore and fail (Lee & Hammer, 2011)
- rapid feedback for player competency (Richards et al., 2014)
- experiential progression (Stott & Neustaedter, 2013)
- explicitly or implicitly defined player-goals (Nicholson, 2012)

More broadly, game elements are diverse in their application and scope. However, the most commonly implemented game elements include design features like quests, guilds, badges, levels, progress bars, experience points, avatars, narrative, leaderboards, and character skills (Malone, 1980a; Lee & Hammer, 2011; Squire, 2011; Gee, 2006; McGonigal, 2011). In addition, effective game elements in educational contexts encourage learners to engage with the content and to progress toward a goal (Kapp, Blair, & Mesch, 2014). Gaming elements that support self-directed learning, constructionism, and connectivism, were implemented into the learning curriculum for three of the case studies. Each study employed a combination of structural gamification and content gamification in its framework.

2.1. Self-Directed Learning Game Elements

The implementation of points, badges, and achievements has been previously shown to facilitate self-directed learning (Fuchs et al., 2014). Through these game elements, students can choose which goals and tasks to undertake and focus their efforts, thus supporting individual autonomy. The use of structural gamification provides several affordances to help learners to gain knowledge, skills, and abilities they need, while promoting student autonomy. Students had control over when they learn and how they decided to approach the learning process (Nicholson, 2012; Kapp, Blair & Mesch, 2014; Huotari & Hamari, 2012; Deterding & Dixon, 2011). Kapp, Blair, and Mesch (2014) argue that establishing clear goals for the user and that adding incremental

goals and rewards, a sense of learner progression, and transparent rules promotes effective structural and content gamification. Goals and rewards, learner progression, and transparent rules are implemented into the gamification three of the case studies (Case Study Four evaluated existing designs).

2.2. Constructionist Game Elements

Structural and content gamification can support constructionist learning, or the process of receiving meaningful feedback, overcoming failure, and gaining mastery. Some of the potential educational benefits that video games have over traditional learning environments include the frequency of feedback and encouragement to learn through failure (Malone, 1980; Gee, 2003; Squire, 2008). Games afford learners the freedom to fail, which encourages them to explore content, hypothesize solutions, take chances with their decision-making, and face consequences in the "safe" environment of a magic circle (Nicholson, 2012; Squire, 2008). The risk of failure without punishment can be engaging because learners can explore and examine causes and effects to content if it is safe to fail and learn from their mistakes (Deterding et al., 2011; Kapp, Blair & Mesch, 2014).

Gamification may support constructionist learning in the form of self-paced exercises, visual cues, and progress bars (Kapp, Blair & Mesch, 2014), which can be used to promote mastery through rapid feedback and overcoming failure. The "freedom to fail" design element is used within each case study's gamification structure by rewarding students on effort rather than generating the "correct answer." In addition, each case study encourages individualized self-pacing and seeks to provide students with rapid feedback on their learning through the use of progress bars, earned badges, and other similar visual gaming cues. The case studies use

gamification to facilitate rapid feedback and the ability to fail in an effort to support constructionist learning and promote skillful mastery for students.

2.3. Connectivist Game Elements

Video games frequently use design elements that create and support community, which may also support connectivist purpose. Leaderboards, guilds, "pick up groups," and other collaboration mechanisms have been known to connect people together to form communities and promote relatedness within the game (Richards et al., 2014; Ducheneaut et al., 2007; Behnke, 2012). These mechanisms may serve the community briefly for a very specific purpose[16] or create lasting community experiences that expand beyond the magic circle of the game (Ducheneaut et al., 2007; Behnke, 2012). Games can also promote connectivist purpose or relatedness by using social-media[17] elements, such as the use of friend-requests, chatroom communications, private messaging, bulletin-board systems, and other forms of communication. For the case studies that incorporate gamified websites, these social-media-like elements are implemented directly into the site design, allowing users to form connections through these design features. For the other case studies, connectivist elements are incorporated into the design by encouraging students to work and explore together in groups.

The impact of these gamification elements was evaluated through a variety of mechanisms. Qualitative analysis interpreted students' self-reported intuitions and expectations of the gaming elements through open-ended questions in surveys, focus-groups, and researcher-observation of the field site. Quantitative data collected both directly and indirectly was used to characterize

[16] In the context of Massively Multiplayer Online Games (MMOGs), "pick up groups" (PUGS) allow players to form small groups to solve an immediate task; the group disbands after the task is done. In contrast, *guilds* represent long-lasting communities that remain active and collaborative for long periods, perhaps years..

[17] Arguably, video games incorporated these elements years before any social-media website had done so.

specific impacts on specific groups of students. The remainder of this chapter details the evaluation methodologies employed.

III. MIXED METHODS

The mixed method framework combines both quantitative and qualitative methods of data collection. This research uses mixed methods to conduct data *triangulation*, the collection of data from multiple sources (in this instance, surveys, field notes, interviews, student coursework, among others).

1. Quantitative Methods

To construct and evaluate surveys for this research, I built on previous work utilizing *Item Response Theory* (Wilson, 2005) and *Likert Scaling* (Likert, 1932). Item Response Theory starts from the assumption that one attempts to measure a single characteristic, called a latent trait or variable. Surveys that measure more than one trait are viewed as multiple instruments administered together, with each instrument requiring validation and reliability testing (Wilson, 2005). In Likert Scaling, each specific question or "item" can have its response analyzed independently, or it can be summed with other related items to create a score for a group of statements.

All four case studies employed quantitative methods to measure student motivation and learning outcomes. "Student motivation" was quantitatively measured using the Intrinsic Motivation Inventory (IMI) surveying, Experience Sampling Method (ESM) and Situational

Motivation Scale (SIMS). "Learning outcomes" were measured from students' course grades[18], proxy pre-tests, and pre-test/post-test surveys. The learning outcomes for each case study are bounded by the context of that specific case. Proxy pre-tests and pre/post-test measures were used to establish the baseline assessment according to each specific learning context and planned objectives.

1.1. Motivation Assessment Surveys

Surveys used in this work build upon previous instrumentation techniques that have undergone validation and reliability testing. The Intrinsic Motivation Inventory (IMI) is a multidimensional measurement survey intended to assess participants' subjective experience related to a target activity in laboratory experiments. It has been used in several experiments related to human motivation and Self-Determination Theory (Ryan, 1982; Ryan, Koestner & Deci, 1991; Deci et al., 1994). The survey instrument assesses participants' interest and enjoyment, perceived competence, and sense of relatedness (Deci et al., 1994). The interest or enjoyment subscale is considered the self-reported measure of intrinsic motivation, and the perceived competence and relatedness concepts represent positive predictors of self-reported and behavioral measures for motivation (Deci et al., 1994). The IMI survey is often modified to fit specific activities, and shorter versions of the survey have been reliably used for predicting and measuring human motivation (Deci et al., 1994). Case Study One (i.e., Virtual Worlds) employs an IMI survey to determine factors relating to intrinsic motivation.

[18] Only the first case study uses students' course grades as a quantitative measure. The other three case studies occur within informal learning environments, so participants are not given "grades" as indication of learning outcomes.

The Experience Sampling Method (Hektner, Schmidth, & Csikszentmihalyi, 2007) is the most widely used method for identifying and measuring *flow*. The ESM repeatedly captures an individual's level of engagement and affect in activity (presumed to be high during flow), as well as the conditions that are theorized to give rise to flow experiences such as high challenge and skill. Case Study Two (i.e., Technovation) and Case Study Three (i.e., CoderDojo) used ESM surveys to measure moments of flow after engaging in curriculum activities. Using ESM surveys in Case Study One (i.e., Virtual Worlds) was not feasible due to the structure of the course, therefore, an IMI survey was used in a proxy pre-test and post-test evaluation to measure student intrinsic motivation.

The Situational Motivation Scale (Guay, Vallerand, & Blanchard, 2000) is designed to assess the constructs of intrinsic motivation, identified regulation, external regulation, and amotivation (Deci & Ryan, 1991). Case Study Four (Online Tutorials) used the SIMS scale rather than the IMI or ESM scale because of the findings that emerged from the previous case studies, showing that intrinsic motivations can be informed by both internal and external factors. In short, students are not only intrinsically motivated to engage in an activity because of personal interests, but are also motivated because of other external social pressures such as parental and teacher expectations, societal importance, grades, etc., thus motivating students to perform an activity. This issue is discussed in more detail in Chapter Eight.

Either IMI, ESM, or SIMS survey questionnaires were administered to students in each case study. Similar to previous studies using the IMI scale (Deci et al., 1994) and ESM survey (Shernoff, Hamari, & Rowe, 2012), I modified the length of these surveys according to the level-of-appropriateness for different groups of students (i.e., survey lengths differ for undergraduate students versus elementary aged children). The IMI, ESM, and SIMS attempt to translate "student

motivation" into ordinal, nominal, and interval variables. Responses to IMI, ESM, and SIMS surveys results were analyzed using standard statistical methods appropriate to the nature of the data collected.

1.2. Pre-Tests & Post-Tests

Each case study employed some form of pre-test and post-test survey in an effort to measure "learning outcomes." Pre- and post-testing is an established evaluation practice to measure knowledge gained from a course or training intervention (Kim & Wilson, 2010). The pre-test, given to participants before the course began, sought to determine students' prior knowledge level of the course content. After the completion of the course, participants were given the same set of questions, and the pre- and post0test results were compared. To determine differences among groups, I used inferential statistics to compare results across groups (e.g., male and female, gamers and non-gamers, etc.), and then used ANOVA to test differences between groups where the independent variables have two or more categories (e.g., freshmen, sophomores, juniors, seniors etc.).

1.3. Web Analytics

Three of the four case studies employ a website to evaluate the gamification intervention. Where possible, these websites gathered relevant data regarding participant activity. These data included participant identify, profile avatars, number of visits, number of comments, lesson submissions with time-stamps, and records of other forms of digital interaction with the website. These interactions were represented with descriptive statistics and analyzed to interpret user behavior in the context of the specific case study (Ochoa & Duval, 2011; Ducheneaut et al., 2007).

2. Qualitative Methods

Typically, there are three approaches to qualitative content analysis (Hsieh & Shannon, 2005), or the ways in which the researcher considers how text, images, video, and other ethnographic data are organized and categorized for analysis. *Conventional content analysis* is used when existing theory or research literature on a phenomenon is limited. In this approach, researchers avoid using preconceived categories (Kondracki & Wellman, 2002) and allow grounded methods (Lindlof & Taylor, 2011) or categories and names for categories to "flow from the data" (Hsieh & Shannon, 2002, p. 1279) to create new theories. Contrastingly, *directed content analysis* uses existing theory and prior research to identify key concepts or variables as initial coding categories (Potter & Levine-Donnerstein, 1999), in that operational definitions for each category are determined using theory. The last approach, *summative content analysis*, identifies and quantifies certain words or context in text into interval variables in an effort to understand the contextual use of words or content (Hsieh & Shannon, 2005). Typically, a summative approach to qualitative analysis goes beyond mere word counts to include latent content analysis or the process of interpretation of the content (Holsti, 1969).

I employed directed content analysis and summative content analysis techniques to analyze the qualitative data produced in the case studies. These procedures include the use of inductive methods, such as open-coding and axial coding procedures (Lindlof & Taylor, 2011). Because my work considers builds upon previous work on Self-Determination Theory, which included theoretical approaches to self-directed learning, constructionism, and connectivism, I created categories of codes derived from these theories, while also allowing the possibility of new codes and categories to emerge from the data. By collecting and analyzing tangible artifacts from the field site, which is another established practice of qualitative research methodology (Strauss &

Corbin, 1990), this allowed me to observe, categorize, and code data from the site into emergent themes.

2.1. Semi-Structured Interviews

Semi-structured interviews are a qualitative method that combines a pre-determined set of open questions (questions that prompt discussion) with the opportunity for the researcher to explore particular themes or engage participants to respond further (Strauss & Corbin, 1990). Semi-structured interviews are used to understand how interventions work and how they could be improved, and allows respondents to discuss and raise issues that may not have been considered by the researcher (Lindlof & Taylor, 2011). I collected qualitative data from the field sites through ethnographic video recordings and semi-structured focus group interviews. In some cases, it was difficult to facilitate structured or individualized interviews with participants, particularly in the case study involving young children. In addition, because two of the four case studies promoted group-work, focus groups were more situationally appropriate as an interview method for these participants.

2.2. Artifact Content Analysis

I collected and analyzed cultural artifacts (Strauss & Corbin, 1990) from the field as a method of qualitative data. Because each case study incorporated its own unique "cultural artifacts" of gamification, it was necessary to analyze each instantiation of this gamification within the relevant context. For example, gamification artifacts in the first case study included the use of badges, points, avatars, leaderboards, etc. on a website. In the third case study, gamification badges are tangibly represented through stickers and stamps displayed at the front a student

workbook. Therefore, considering gamification artifacts with respect to the cultural context for each case was important for the evaluation of this work.

3. Threats to Internal Validity

3.1. Personal Experiences and Biases

My personal experience and worldviews on this research, as well as my past experiences, personal beliefs, and other connections with the participants in the research can potentially influence the analysis of my findings (Creswell, 2003). For example, I consider myself to be a "gamer" and find intrinsic and extrinsic socio-cultural value within gaming spaces. These beliefs contributed to my hypothesis that gaming would positively support student motivation and learning. In addition, I believe that *gamification* as a practice has been largely ineffective, and I tend to agree with Bogosts's characterization of these efforts as "exploitationware." This view is supported by observations of successful examples of gamification in non-educational settings such as Frequent Flyer Miles, or the success of *Foursquare*.

In addition, as a woman who took an unconventional path to learn introductory computer science, I have also found many traditional methods for teaching CS to be off-putting, in that most CS learning environments did not appear to be "intended for me." This experience has helped frame my attitudes toward CS education.

3.2. Issues as the Teacher-Researcher

In two of the four case studies, I acted simultaneously as teacher and researcher. Teacher research is "intentional and systematic inquiry done by teachers with the goals of gaining insights into teaching and learning, becoming more reflective practitioners, effecting changes in the classroom or school, and improving the lives [of students]" (Henderson et al., 2012, p. 1).

Although the role of a *teacher researcher* (MacLean & Mohr, 1999) is a common methodological process in education research, it is less common in other fields of research. Previous work (MacLean & Mohr, 1999) indicates that teacher research offers an opportunity to shape educational practice and to validate, affirm, and improve teachers' personal practice. In addition, the role of a teacher-researcher is arguably less problematic in the context of a case study, since case studies are, by definition, situationally subjective and bounded by time and activity, which is directly dependent on the context of its environment. In this context, my role as teacher-researcher aligns with the case study research strategy. In two of the four case studies, I did not have any direct role as a teacher.

3.3. Triangulation of Sources

I collected information form a variety of sources including quantitative surveys, web-analytics, document analysis, observations, and interviews. This allowed me to triangulate findings and help establish the validity and reliability of these findings. I have endeavored to ensure that these findings did not originate from a single source, and that conclusions were supported by other data sources. For example, students who reported that they liked or enjoyed the self-directed component of the gamification system were compared in the quantitative analysis of a survey-item, which also indicated a high ordinal rank for liking the gamification system.

3.4. Presentation of Negative Findings

When relevant, negative and discrepant findings are presented and rival explanations are explored (Patton, 1995).

CHAPTER V
CASE STUDY ONE: VIRTUAL WORLDS

This chapter discusses the first case study conducted for this research. The case study takes place within an introductory computer science undergraduate course. During the fall 2014 semester, I investigated the impact of gamification on student motivation and learning of CS Principles by teaching two sections of the undergraduate course, CSCI/ATLS-1220 *Virtual Worlds: Introduction to Computer Science*. This is an introductory computer science class intended for non-majors at the University of Colorado Boulder. Acting as a *teacher researcher* (MacLean & Mohr, 1999), I investigated how CS principles pedagogy can support self-directed learning, constructionism, and connectivism, while also exploring whether gamification impacted self-directed learning, constructionist, and connectivist learning for undergraduate students.

I. SITE SELECTION

Between 2008 and 2012, I served as a teaching assistant for previous iterations of this course. During the fall 2014 semester, I became the instructor of record for *Virtual Worlds*, which enabled me to structure the course around CS Principles curricula. The course's reputation from previous years—many different majoring students taking the course, and a higher representation of women compared to other introductory CS courses[19]—provided a favorable demographical representation, which matches the theoretical objectives afforded by the CS Principles pilot curricula and the intended research objectives.

[19] FCQ responses from students can be publicly viewed online at https://fcq.colorado.edu/UCBdata.htm

II. RESEARCH DESIGN

1. Duration of Study

The duration of this case study is one academic semester (sixteen weeks) during the fall 2014 semester at the University of Colorado Boulder. One-year after the class took place, a follow-up survey was issued to students to measure lasting effects and impact of the course. This course is not a required course for any degree or major, although it does offer elective credit for the ATLAS Institute and CS Department undergraduate programs.

2. Control & Variable Assignment

Traditional randomized experiments cannot be used in most educational contexts because randomizing educational treatments for students is considered unethical (Brown, 1992). Therefore, I incorporate a quasi-experimental, *nonequivalent proxy pretest design* or post-test-only control group design with multiple levels of treatment and proxy pretest measures. The "proxy" variable (O_1) is used to estimate where the groups would have been on the pre-test. Proxy pre-test designs look like a standard pre-post design, but the pre-test in this design is collected during the post-test instrumentation, as shown in the table below.

(O_1)	X	O_2
(O_1)		O_2

Table 7. Non-randomized Treatment for Proxy Pre-Test & Post-Test

There are essentially two variations of this quasi-experimental design—in the first, the researcher asks participants to estimate where their pre-test level would have been at a given time. This is called the Recollection Proxy Pre-Test Design (Trochim, 2006). In the second format, the researcher uses record archives to stand in for the pre-test such as standardized test data, census data, etc.

I used control and variable groups as the basis for this nonequivalent group research design in an effort to measure the impact of gamification on student motivation and learning outcomes for CS Principles. The first section of the *Virtual Worlds* course, Section 001, serves as the variable group for this research. I chose the first section of the course to be the experimental platform because it enrolled more students and could potentially represent more diverse groups of students[20]. This variable group experienced the "gamification treatment," or the experimental use of game elements within the course curriculum. The second section of the course, Section 002, serves as the control group and did not facilitate gamification. The variable group incorporates gamification elements into the course structure, whereas the control group uses traditional educational constructs; otherwise, the curricula content is identical for both sections of the *Virtual Worlds* course. The gamification treatment is compared to the results from the control group through quantitative and qualitative procedures.

Enrolled students did not have prior knowledge that the course implemented CS Principles curricula. The University of Colorado Boulder course catalog reflected the course description from previous iterations of the *Virtual Worlds* course (describing the use and application of *Second*

[20] Because ATLS/CSCI-1220 had been taught during Fall since 2008, the first section had a dedicated room and time in the CU Boulder course catalog. The second section of the course was added to the course catalog during summer 2014 for the purposes of my research. Because the second section's enrollment and room was registered "late" in the summer, this resulted in a smaller room that could not enroll as many students as the first section.

Life[21] as the "laboratory" for the course). Prior to enrollment, students were unaware that *Virtual Worlds* would use CS Principles as the pedagogical structure to the course.

I was careful not to disclose that one section of the course incorporated gamification elements and that the other section did not—an effort to keep internal validity sustained between the two groups as much as possible. Although the gaming elements were discussed during the first day of class and in the course syllabus, the term "gamification" was not used on the syllabus or in class, nor was there mention that the other section of the course would not offer these game elements (and vice versa). I discussed on the course syllabus, the course website, and in class lectures that I am conducting research in *Virtual Worlds* as part of my doctoral work, but I did not disclose the intent behind this work to the students. This was done in an effort to prevent student predisposition toward a Hawthorne Effect (Roethlisberger & Dickson, 1939) or other similar phenomenon.

2.1. Criticisms of Quasi-Experimental Design

One of the criticisms for quasi-experimental research is the lack of random assignment, and that the potential nonequivalence between groups complicates the statistical analysis of the nonequivalent group design. For example, groups may be different prior to any treatment or non-treatment, and these group differences may affect outcomes. However, positivist randomization experiments are generally not feasible for educational contexts (Brown, 1992). A nonequivalent group design was chosen for the first case study because a traditional college classroom affords the assignment of control and variable groups between different course sections of the same class.

[21] *Second Life* is a virtual world, similar to a Massively Multiplayer Online Game (MMOG), which emphasizes social interaction and player-generated content. This player-generated content uses Linden Scripting Language (LSL), an object-oriented language similar to the C-family. Second Life and LSL was used as the programming tool for previous iterations of this course.

Another criticism is the use of a proxy pre-test instead of a fully designed pre-test. Critics argue that participants may forget where they were at some prior time or students may distort the pre-test estimates to make themselves look better (Trochim, 2006). Although this argument is valid, there are cases when researchers may not be interested in where they were on the pre-test, but rather in where participants *think* they were (Trochim, 2006). Since *Virtual Worlds* is an introductory course that is not required by any major, it is assumed that students would only have entry-level knowledge of the subject. In addition, measuring students on where they think they are in terms of CS expertise is more directly aligned to the research questions of this work, in which attitudes in this subject are socio-culturally discursive. To offset this concern, survey questions in the proxy pre-test were used to measure CS knowledge and programming experience in various ways. This issue is discussed further in the Instrumentation & Evaluation section of this chapter. In brief, the recollection proxy pre-test is a sensible way to assess participants' perceived gain or change in the context of this case study.

3. Description of Student Population

A total of sixty-six students enrolled into the first section of the course. The variable group had twenty-four women enrolled in the course (36%). The control group enrolled twenty-eight students into the course, three of whom were female (11%).

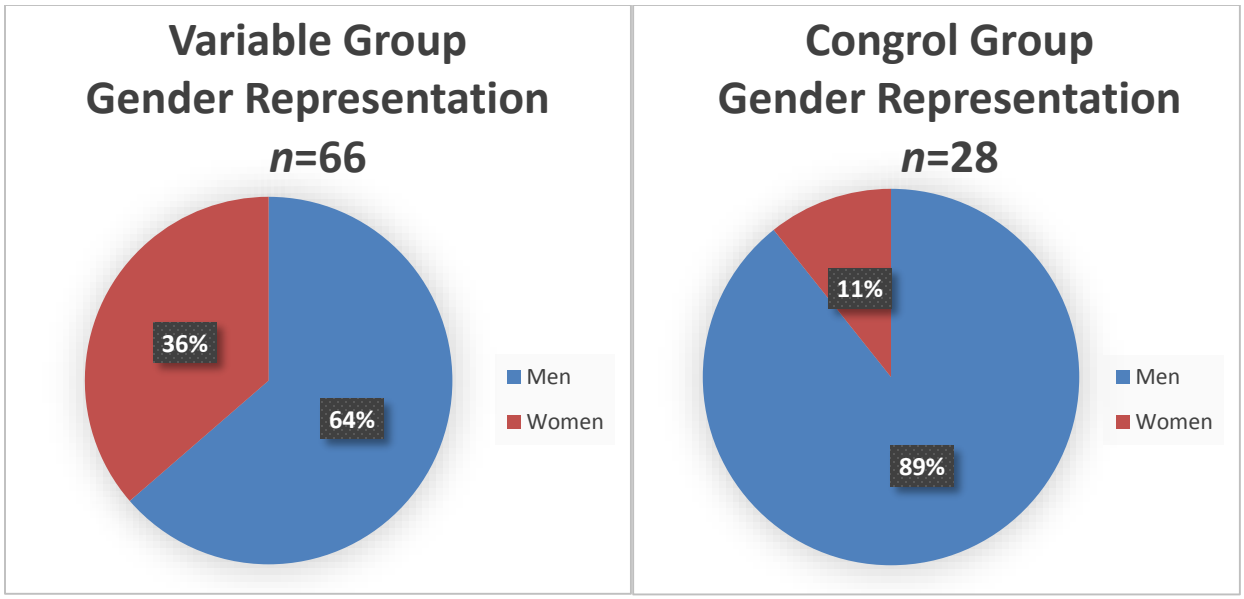


Figure 1. Gender Representation of Variable Group (left) & Control Group (right)

The control group therefore had a more "traditional" representation of women in computer science, whereas the variable group presented a more favorable representation intended. The control group's percentage of women (11%) is close to the national average (12%) of undergraduate CS degree recipients who are women (NCWIT, 2014).

The ethnic distribution in both sections was predominately White; however, non-majority populations were present in both sections of the course. The self-reported racial and/or ethnic distribution for each course section is shown below:

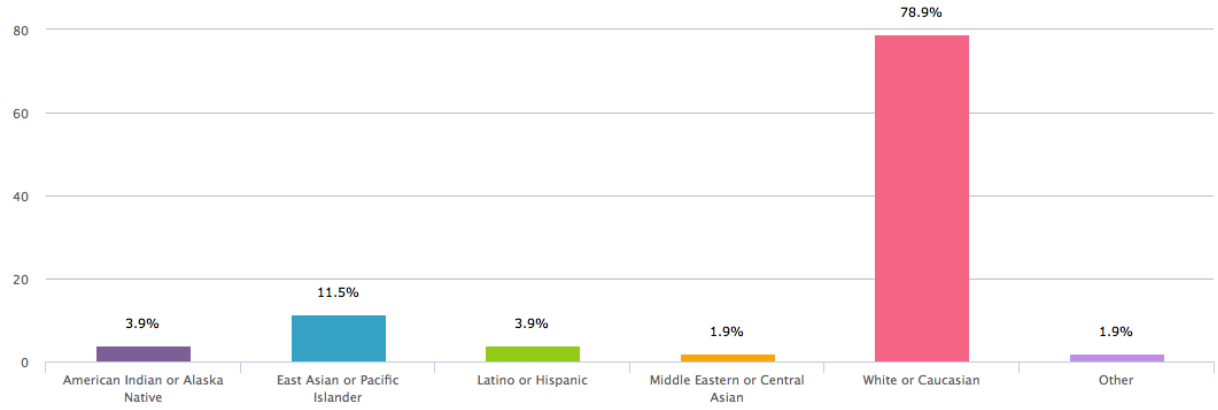


Figure 2. Ethnic & Racial Distribution in Variable Group

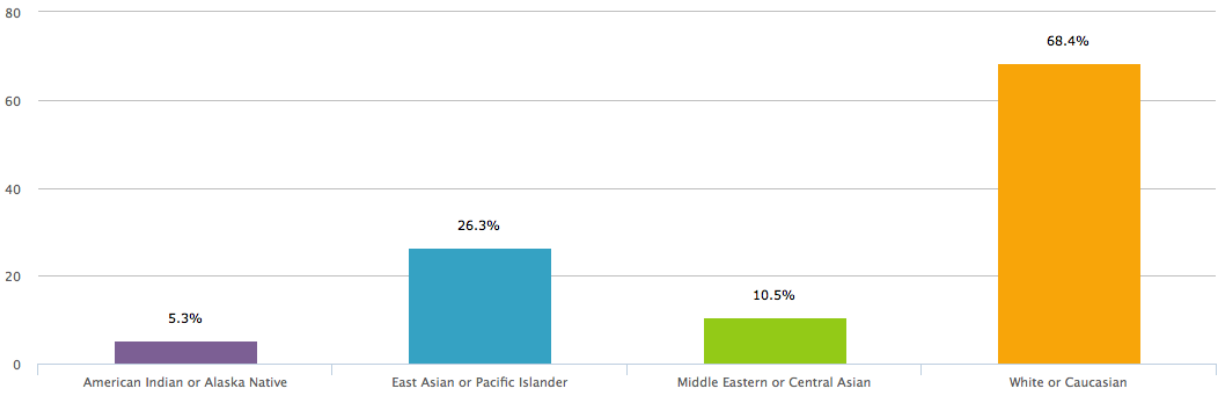


Figure 3. Ethnic and Racial Distribution in Control Group

Although there are more non-majority groups in the first section of the course, both sections reflect that White or Caucasian students were the majority group of students. This diversity spectrum of non-majority students is generally similar to those who take AP CS courses in high school (Ericson & Guzdial, 2014); but there are lower representations of Black and Hispanic populations than the average representation of students that take the AP CS A test in the U.S. (Ericson & Guzdial, 2014).

The breakdown for the enrollment status in each course section is shown below.

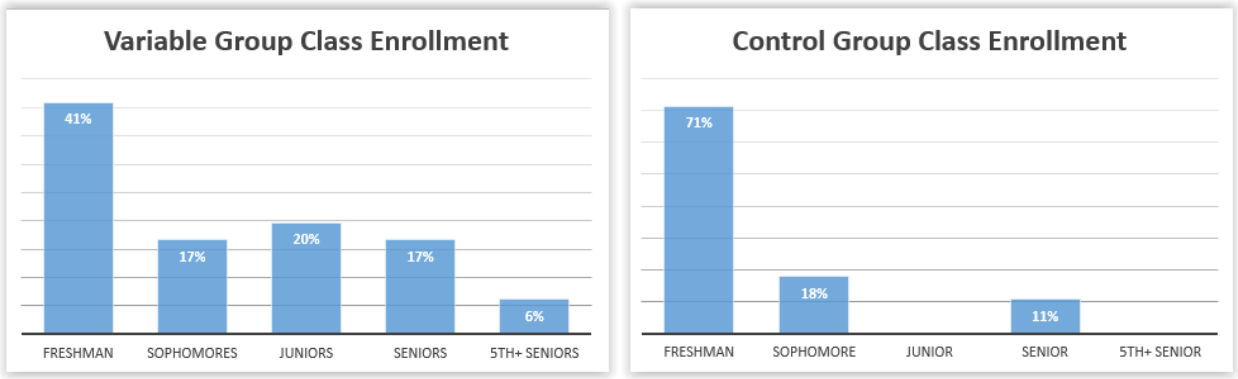


Figure 4. Student Enrollment in Variable (left) & Control (right) Groups

Most of the students taking *Virtual Worlds* were freshman; however, a range of other students also took the course, including sophomores to fifth-year seniors. This is likely explained because of the fact that *Virtual Worlds* is an elective course intended for non-majors, so it may fulfill multiple course elective requirements across CU Boulder campus.

In total, thirty-four different majors were represented in the course, including majors in dance, film, psychology, mathematics, physics, engineering, biochemistry, neuroscience, political science, architecture, environmental design, among others. In the variable group, thirteen students (19%) were undeclared, open-option, or pre-engineering majors, whereas in the control group, twelve students (43%) were undeclared, open-option, or pre-engineering majors. In addition, five students (1%) in the first section were declared Computer Science[22] majors and six students (21%) were declared Computer Science majors in the control group. The breadth of major disciplines

[22] No students majoring in Computer Science in the Bachelors of Science (CS-BS) program were represented in this course. All CS students were registered in the new Bachelors of Arts program (CS-BA).

indicates that *Virtual Worlds* reached a range of students from multiple disciplines across CU Boulder campus.

IV. IMPLEMENTATION

1. Course Website

Each section of the course had its own website with a carefully designed learning-management system (LMS) for the students to use throughout the semester. The course website served as the main platform for students to retrieve lesson materials and to upload their coursework throughout the semester. Each section of the course also had a website and unique domain name[23] in an attempt to prevent student-crossover from the respective control and variable groups. The figure below shows a screenshot of the course home page or index page, which is virtually identical for the control and variable groups.

[23] The variable group's course URL is http://CSPrinciples4Buffs.com.
The control group's course URL is http://CSPrinciples4UCB.com

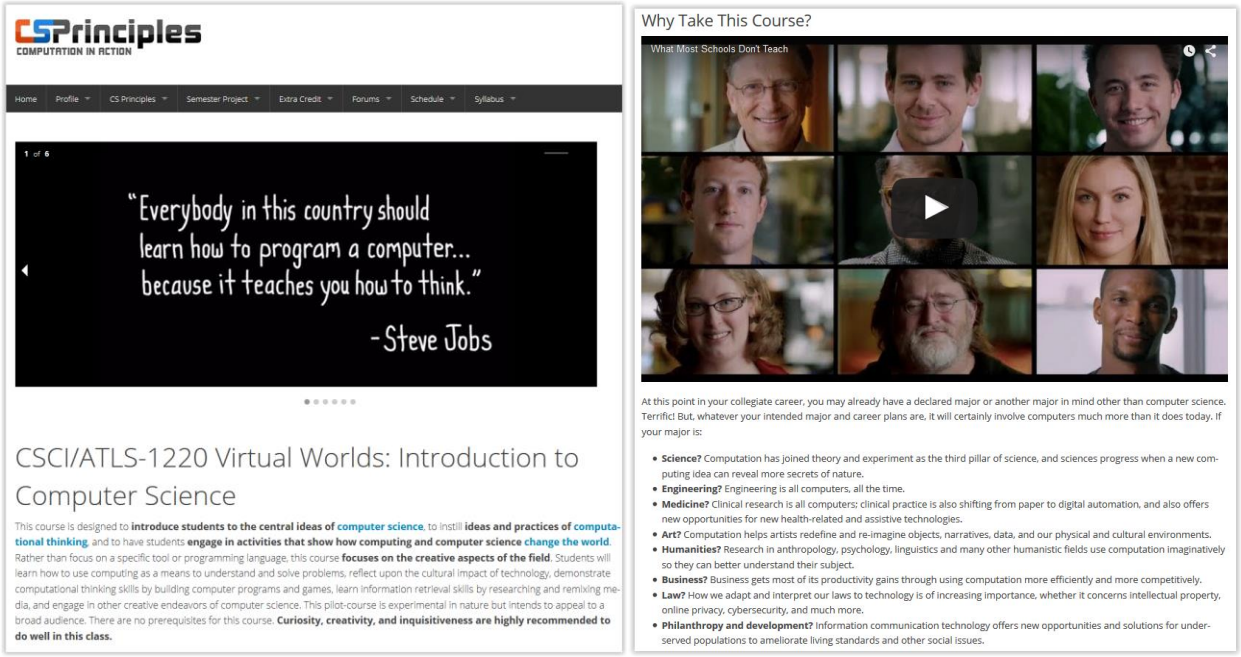


Figure 5. Screenshots of the *Virtual Worlds* Course Homepage

The course websites were built using WordPress, an open-source content management system that uses HTML, CSS, PHP, and MySQL to manage Web content. Each website was identical in its content and course requirements for the students—the exception being the added game elements, which were incorporated into the variable group's course website. The "gamified" website used a commercial gamification plugin called BadgeOS™, used to implement and assess "typical" gamification practices used in online instructional design (Deterding & Dixon, 2011; Nicholson, 2012; Hamari et al., 2014).

The course objectives, student requirements for completing homework assignments, the midterm examination, quizzes, and a semester-long project (used instead of a final examination), are identical for each section of *Virtual Worlds*. For the variable group (gamified classroom), traditional learning terminology was replaced with metaphors typically found in video games, such as missions, quests, boss fights, a bonus round, and a journey book, as shown in the figures below.

OBJECTIVES: COURSE MATERIALS	PERCENTAGE OF GRADE
Quests: Labs & Assignments	50%
Epic-Quest: Semester Project	20%
Mission Check: Midterm Examination	15%
Journey Book: Reflection Posts	10%
Boss Fights: Online Tests	5%
Bonus Round: Extra Credit	up to 5% of your final grade

Figure 6. Gamified Course Objectives (Variable Group)

Course Objectives	Percentage of Grade
Laboratory assignments (Labs)	50%
Semester Project	20%
Midterm Examination	15%
Reflection Posts	10%
Online Tests	5%
Extra Credit	up to 5% of your final grade

Figure 7. Traditional Course Objectives (Control Group)

The variable-group LMS implemented other gamification mechanics, including the use of a leaderboard, a point system, and an achievement system represented through the earning and receiving of badges. No mention of these badges, leaderboards, or points was discussed in class—an attempt to sustain internal validity with student intrinsic/extrinsic motivations and conceptual understandings of gamification. Students in both sections of the course received "self-directed learning," "constructionist," and "connectivist" mechanisms through the website. This topic is discussed further below.

2. Course Structure

Virtual Worlds used the CS Principles computational thinking practices to structure the course. These practices are: *Connecting Computing*, *Developing Computational Artifacts*, *Abstracting*, *Analyzing Problems & Artifacts*, *Communicating*, and *Collaborating* (College Board, 2013). Below, I summarized the definition for each computational thinking practice, connect it with the seven "Big Ideas" that define the CS Principles curriculum, and theoretically situate these items within the relevant Self-Determination Theory categories:

1) Self-Directed Learning CS Principles (Autonomy)

- Connecting Computing: Developments in computing have far reaching effects on society and have led to significant innovations. The developments have implications for individuals, society, commercial markets, and innovation. Students in this course study these effects, and they *learn to draw connections* between different computing concepts.
    - Big Idea #1: Creativity
        - Computing is a creative activity.
- Analyzing Problems & Artifacts: The results and artifacts of computation and the computational techniques and strategies that generate them can be understood both intrinsically for what they are as well as for what they produce… Students in this course design and produce solutions, models, and artifacts, and they *evaluate and analyze their own computational work* as well as the computational work others have produced.
    - Big Idea #2: Algorithms
        - Algorithms are used to develop and express solutions to computational problems.

2) Constructionist CS Principles (Mastery)

- Developing Computational Artifacts: Computing is a creative discipline in which creation takes many forms… Students in this course engage in the creative aspects of computing by *designing and developing* interesting computational artifacts, as well as by applying computing techniques to creatively solve problems.
    - Big Idea #3: Programming
        - Programming enables problem solving, human expression, and creation of knowledge.
- Abstracting:[24] Computational thinking requires understanding and applying abstraction at multiple levels… Students in this course use abstraction to *develop models and simulations* of natural and artificial phenomena, use them to make predictions about the world, and analyze their efficacy and validity.
    - Big Idea #4: Abstraction
        - Abstraction reduces information and detail to facilitate focus on relevant concepts.
    - Big Idea #5: Data & Information
        - Data and information facilitate the creation of knowledge.

3) Connectivist CS Principles (Purpose)

- Communicating: Students in this course describe computation and the *impact* of technology and computation, explain and justify the design and appropriateness of their computational choices, and analyze and describe both computational artifacts and the results or behaviors of such artifacts.
    - Big Idea #6: The Internet
        - The Internet pervades modern computing.
- Collaborating: Innovation can occur when people work together or independently… Learning to collaborate effectively includes drawing on diverse perspectives, skills, and the backgrounds of peers to address complex and open-ended problems. Students in this course *collaborate* on a number of activities, including investigation of questions using data sets and in the production of computational artifacts.
    - Big Idea #7: Global Impact
        - Computing has global impact.

Table 8. Summary of CS Principles Instantiated via SDT

[24] I incorporated two big ideas to the *Abstracting* computational thinking practice because there are six computational thinking practices and seven big ideas in total. To account for the odd number of big ideas, I associated Data and Abstraction with the *Abstracting* computational thinking practice because both of these big ideas conceptually match this computational thinking practice and related constructionist practices.

A review of the computational thinking practices and their associating big ideas leads to an association of *Connecting Computing* and *Analyzing Problems and Artifacts* with self-directed learning mechanisms, *Developing Computational Artifacts* and *Abstracting* with constructionist-learning methods, and that *Communicating* and *Collaboration* practices with connectivist connections, as explained below.

Connecting Computing and *Analyzing Problems & Artifacts* encourage self-directed learning, or student autonomy, by encouraging students to "learn to draw connections," to be "creative" in their learning, to "evaluate and analyze their own computational work," and "develop and express their own solutions" to computational problems. The focus on self-development and the freedom of creative expression underscores autonomy. *Developing Computational Artifacts* and *Abstracting* support constructionist learning or help students gain mastery of skills through "learning-by-making." For these computational thinking practices, students learn by "designing and developing interesting computational artifacts," and "develop models and simulations of natural and artificial phenomena," which leads to the "[facilitation of] the creation of knowledge." Students engage in constructionist learning by modeling and building artifacts, which engages the student into the construction of knowledge. Finally, the *Communicating* and *Collaborating* practices support connectivist learning by requiring students to "describe computation and the impact of technology and computation." Students contemplate and communicate purpose, "collaborate on a number of activities" with other students, and promote a sense of relatedness by expressing how "computing has global impact." The connection and communication with others to create knowledge employs the foundation of connectivism. I used these computational thinking practices, theoretically framed by SDT to structure the course objectives and intended learning outcomes for the course.

3. Learning Objectives

The learning objectives directly correlate to the CS Principles computational thinking practices and big ideas. The computational thinking practices frame the "policy" or larger theoretical implications of the course, whereas the big ideas implement the "mechanisms" to facilitate these practices. The learning objectives are the same for both control and variable groups; however, the variable group uses gamification to frame the course objectives, whereas the control group uses academic nomenclature. The table below juxtaposes the terminology used to indicate the student learning objectives.

Gamification Objectives (Variable Group)	**Traditional Objectives (Control Group)**
6 Missions	6 CS Principles
35 Quests	35 Laboratory Assignments
6 Boss Fights	6 Online Tests
Mission Check	Midterm Examination
Journey Book	Self-Reflections
Epic-Quest	Semester Project
Bonus Round (optional)	Extra Credit (optional)

Table 9. Course Objectives for ATLS/CSCI-1220

Students must complete the course requirements by the end of the semester[25]. For the variable group, the six computational thinking practices were gamified as "Missions," whereas for the

[25] ATLS/CSCI-1220 is a four-credit course. Although an introductory computer science course for non-majors, universities recommend that students spend two to three hours per credit on any given course. As a four-credit course, students are expected to spend between eight to twelve hours a week on this course or complete approximately 2.5 assignments per week to stay on track.

control group, they were referred to as "CS Principles." Each "Quest" or "Lab" represented a specific CS Principle big idea: Creativity, Algorithms, Programming, Data, Abstraction, the Internet, and Impact. In order to complete a Mission, students had to complete each Mission's set of Quests and complete the assessment test or "Boss Fight." I use the *boss fight* (a design element used in many successful games) as a gamification metaphor for an online test. Games often use boss fights to test the player on all of the cumulated skills she learned while overcoming challenges; "winning" the boss fight demonstrates a mastery of these gained skills, which subsequently completes the level. Therefore, in order for students to "complete a Mission," they must gain the necessary "Skills" (i.e., Creativity, Algorithms, Programming, Data, Abstraction, Internet, and Impact) by completing all of the Mission Quests and winning the corresponding Boss Fight. For example, to complete the *Developing Computational Artifacts* Mission, a student must complete all Programming Quests and win the Boss Fight to complete this course objective. In other words, students had to complete all laboratory assignments and online quizzes for each computational thinking practice in order to achieve that learning outcomes for the course.

In addition to completing all six computational thinking practices (Missions or CS Principles), students had to take the "Mission Check" or midterm examination, post self-reflections about their learning progress (i.e., write in the "Journey Book"), and complete an "Epic-Quest" or semester-long project. Rather than take a final examination, students were required to plan, develop, and submit a semester-long project by the end of the course. The semester project was intended to be broad enough to allow students to express individual creativity, and structured enough so that students produced approximately the same amount of work. The semester project required students to submit a Project Proposal, Digital Artifact (construction of a computational artifact), and Written Report (communicating purpose) of their work. Students also had

opportunities to receive extra credit toward their grade by participating in a "Bonus Round," which provided "Side Quests" or additional labs for students to complete.

Each assignment for the course was grounded in self-directed learning, constructionism, or connectivism, and designed to reflect the core affordances of the relevant big idea and computational thinking practice. The table below lists all 35 Quests/Labs and corresponding theoretical application:

"Big Idea" Quests or Labs	**Computational Thinking Practices**
Abstraction	***Abstracting***
BlueJ: Installation	connectivist
BlueJ: Hello Haiku	constructionist
Play *Light Bot 2.0*	self-directed
BlueJ: Fantasy RPG	constructionist
BlueJ: Shaping Your House	constructionist
Data	***Abstracting***
Digital Detective	connectivist
Hindsight: Machine Architecture	self-directed
Torrents: Is This Legal?	self-directed
Mining Bitcoins	connectivist
Infographs	constructionist
Algorithms	***Analyzing Problems & Artifacts***
Logic Gates exercise	constructionist
Chatbots & Watson	self-directed
State Machines	constructionist
Processing: Drawing & Redrawing	connectivist
Processing: Jitterbug	constructionist

Impact	***Collaborating***
Read & Reflect: Helprin's "Acceleration of Tranquility"	connectivist
Read & Reflect: "Blown to Bits," chapters 1-2	connectivist
Read & Reflect: "The Medium is the Message"	connectivist
StatWorld: Exploring the Digital Divide	connectivist
Explore an ICTD	self-directed
Internet	***Communicating***
Learning Styles Assessment Quiz	connectivist
Read & Reflect: Vannevar Bush's "As We May Think"	connectivist
Surfing the Net: Who Sees You?	self-directed
Anonymity Online: TOR & 4Chan	self-directed
Surveillance & You	connectivist
Creativity	***Connecting Computing***
"What I Value" Writing Exercise	connectivist
Gameplay Reflections	self-directed
Processing: Homage to a Square	constructionist
Processing: Animator	constructionist
Your Life as a Database	self-directed
Programming	***Developing Computational Artifacts***
Processing: Installation	connectivist
Processing: Hello World!	constructionist
Processing: The P5 Robot	constructionist
Processing: Blinky (Part 1)	constructionist
Processing: Blinky (Part 2)	constructionist
Bonus Round / Extra Credit	N/A
Play *Cybersecurity Lab* Game	self-directed
Processing: Tech Drawing	constructionist

Table 10. Required Quests/Labs (Except Bonus Round/Extra Credit)

Assignments that incorporate constructionism required the student to develop a computational artifact, such as writing a computer program, creating a data visualization, writing or drawing an algorithm, etc. Some of these assignments also required the student to post self-

reflections about their construction or learning process, thus facilitating their construction of knowledge. Quests/labs with a self-directed theme encouraged the student to explore and create content according to their personal interests, while still giving guidance to support their learning. For example, students could choose which video games to play and review, reflect on their own personal goals for college or career, chose which AI Chatbot simulations to "talk" with, among other choices that facilitated engagement with the content. Connectivist assignments required the student to connect multiple themes or lecture topics together, for example by understanding why Java and JDK plugins are needed to run BlueJ software, using data visualizations to simulate the digital divide, or reflecting on a particular author's prescient thesis about the Internet and use of technology in modern society. Rather than give "one true answer" for each assignment, students were encouraged to explore, hypothesize, and build according to their individual interests. Therefore, the learning outcomes were directly linked to the amount of effort put into the work. As course syllabus states, "What you get out of this course will equal the amount of effort you put into it."

4. Theoretical Framework

4.1. Self-Directed Autonomy

Virtual Worlds was designed to encourage student autonomy by implementing self-directed learning so that students was capable of taking ownership of his or her learning experience. According to Knowles (1975) and Hiemstra (1994), self-directed learning requires that:

1) students take initiative and responsibility for their own learning;
2) students select, manage, and assess their own learning activities;

3) motivation and volition are critical for success;
4) independence in setting goals and defining what is worthwhile to learn;
5) teachers provide scaffolding, mentoring, and advising;
6) peers provide collaboration to facilitate the group's learning.

Self-directed learning was implemented in the case study by employing a "self-paced" structure for the course. Each of the requirements enumerated above was realized in a particular aspect of the course.

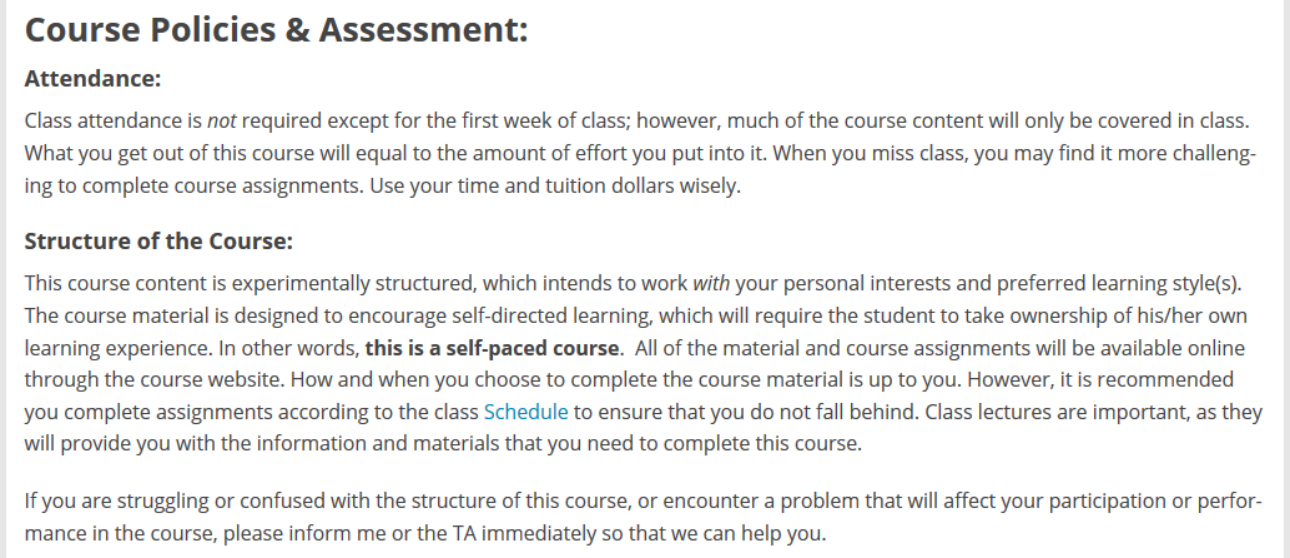
Students could choose where to put their efforts, with in a hard deadline for all coursework at the end of the semester. In order to facilitate curriculum objectives in a self-paced course, all the course materials and assignments were available on the LMS website during the first week of class[26]. I also implemented *scaffolding* techniques (Sawyer, 2006) to support student autonomy. This included setting semester-long goals for the students, but allowing enough flexibility for them to focus their efforts onto their own interests. The official learning objectives for the course were:

1) Demonstrate your mastery of six CS Principles/Missions, with each Principle/Mission requiring you to take an online Quiz/Boss Fight for completion
2) Research and develop a Semester Project/Epic-Quest
3) Complete the Midterm Examination/Mission Check
4) Write self-reflections about your learning experiences/write in your Journey Book throughout the semester on the website Forums *or* submit privately on the course website

[26] It is uncommon for undergraduate instructors and professors to disseminate all course materials at the beginning of the semester. This required a substantial amount of preparation before the start of the academic term.

For example, the semester project or Epic-Quest was open-ended so that students could focus on a particular topic that was of interest to them. In addition, the requirement for students to self-reflect about their learning experiences by writing about them was a mechanism designed to assess students' individual learning activities, a critical component to self-directed learning. Finally, students were given an opportunity to receive extra credit (labeled as "Bonus Round" in the variable group and "Extra Credit" in the control group) for completing additional assignments. Students could independently choose whether or not to complete these additional assignments, just as they could set their own goals by choosing which assignments to focus on first, last, etc.

"Self-directed autonomy" is also facilitated by not requiring class attendance (except for the first week of class for student waitlist purposes). This approach is unconventional for undergraduate classes. Allowing students to choose whether or not to attend course lectures is considered instructionally "risky" (Brockett & Hiemstra, 1991) yet this design mechanism matched the theoretical underpinnings for autonomous, self-directed learning and intrinsic motivation. Self-directed learning postulates that students can demonstrate learning in a variety of ways. For example, instead of having all students take exams or give presentations, students should be able to select the way in which they demonstrate how they have met learning outcomes (Rose & Meyer, 2002). The result is a course that is meaningful for a wider variety of learners (Nicholson, 2012). I did not want to potentially thwart student autonomy by externally coercing them to attend the course lectures—this could have had an inadvertent effect on the independent variables, such as intrinsic motivation. As it stated in the course syllabus:



Course Policies & Assessment:

Attendance:

Class attendance is *not* required except for the first week of class; however, much of the course content will only be covered in class. What you get out of this course will equal to the amount of effort you put into it. When you miss class, you may find it more challenging to complete course assignments. Use your time and tuition dollars wisely.

Structure of the Course:

This course content is experimentally structured, which intends to work *with* your personal interests and preferred learning style(s). The course material is designed to encourage self-directed learning, which will require the student to take ownership of his/her own learning experience. In other words, **this is a self-paced course**. All of the material and course assignments will be available online through the course website. How and when you choose to complete the course material is up to you. However, it is recommended you complete assignments according to the class Schedule to ensure that you do not fall behind. Class lectures are important, as they will provide you with the information and materials that you need to complete this course.

If you are struggling or confused with the structure of this course, or encounter a problem that will affect your participation or performance in the course, please inform me or the TA immediately so that we can help you.

Figure 8. Syllabus Policies & Assessment

The self-pacing and optional course attendance is "experimentally structured, which intends to work *with* your personal interests and preferred learning style(s)." To promote a self-directed learning approach, I emphasized that "how and when you choose to complete the course material is up to you," but warned that they should "use your time and tuition dollars wisely."

Students were not left completely unguided and without direction—prior work in self-directed learning has shown that some guidance from the instructor is needed in order to properly support the intended learning outcomes (Hiemstra, 1994). Therefore, the course website offered several ways to scaffold and guide student work, while preserving student autonomy. One of these scaffolding mechanisms was a class schedule with recommended deadlines and objectives on a week-to-week basis, as shown in the figure below:

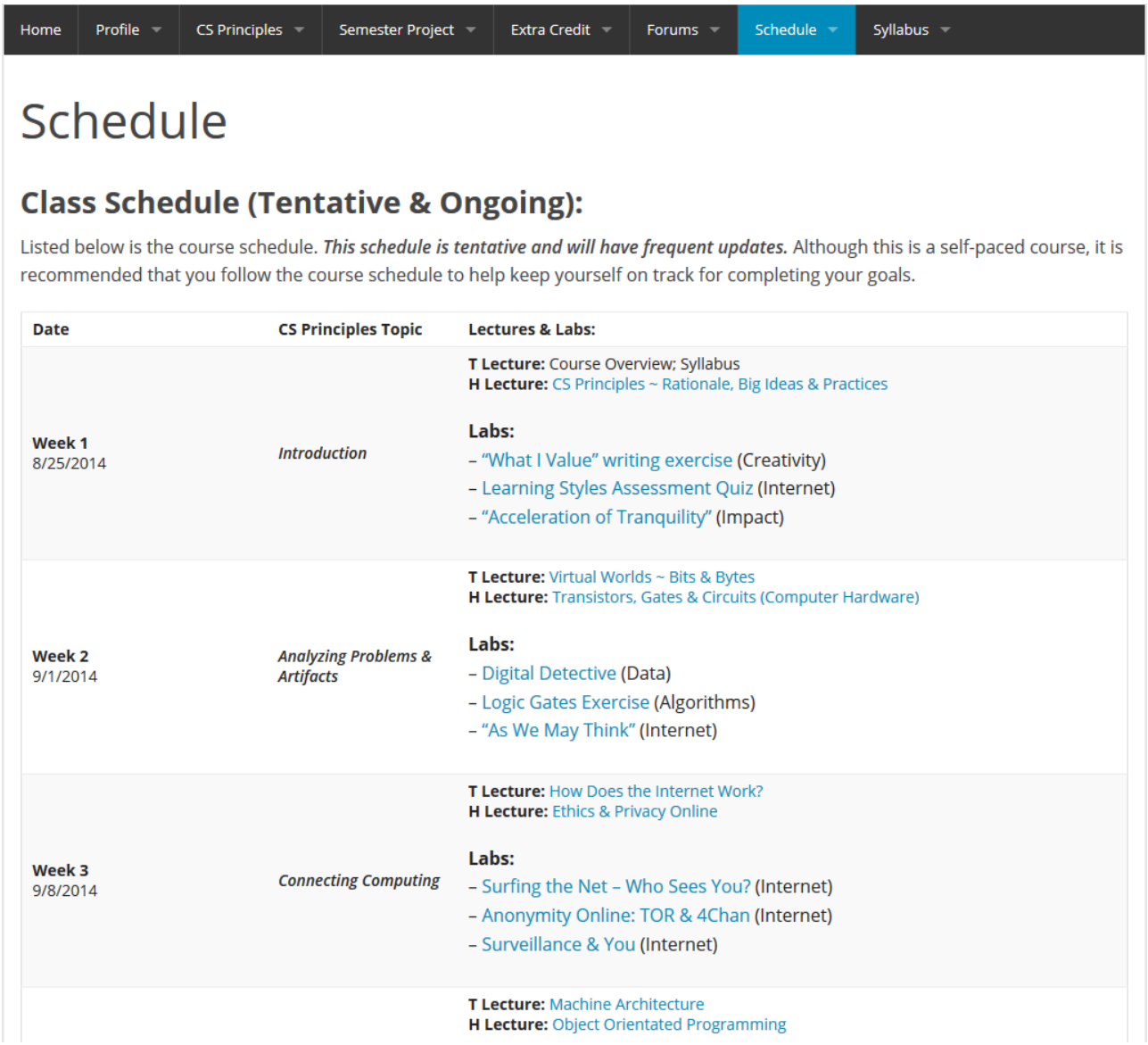
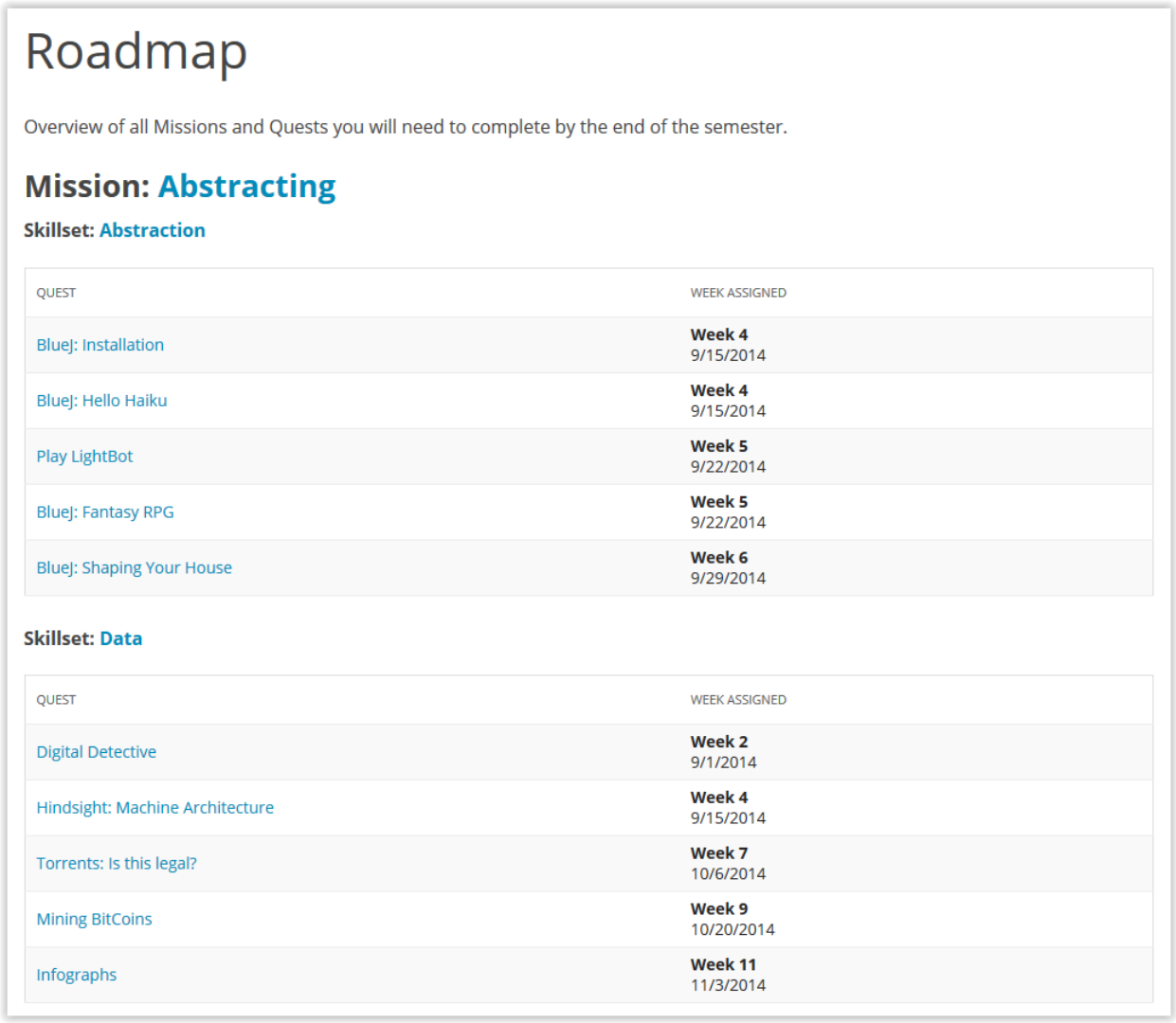


Figure 9. Daily Class Schedule (Control Group)

As stated in the schedule (and multiple times in the website and syllabus), "Although this is a self-paced course, it is recommended that you follow the course schedule to help keep yourself on track for completing your goals." The course schedule attempted to scaffold and support student milestones by offering recommended deadlines for each Quest/Lab and its corresponding Mission/CS Principle. Similarly, a course "Roadmap" was also provided to help students break

down tasks and set goals for completing coursework. The figure below is a screenshot of this roadmap, structured by Missions category and associated Quests.



Roadmap

Overview of all Missions and Quests you will need to complete by the end of the semester.

Mission: Abstracting

Skillset: Abstraction

QUEST	WEEK ASSIGNED
BlueJ: Installation	**Week 4** 9/15/2014
BlueJ: Hello Haiku	**Week 4** 9/15/2014
Play LightBot	**Week 5** 9/22/2014
BlueJ: Fantasy RPG	**Week 5** 9/22/2014
BlueJ: Shaping Your House	**Week 6** 9/29/2014

Skillset: Data

QUEST	WEEK ASSIGNED
Digital Detective	**Week 2** 9/1/2014
Hindsight: Machine Architecture	**Week 4** 9/15/2014
Torrents: Is this legal?	**Week 7** 10/6/2014
Mining BitCoins	**Week 9** 10/20/2014
Infographs	**Week 11** 11/3/2014

Figure 10. Roadmap for Missions & Quests (Variable Group)

The roadmap organizes homework material (Labs/Quests) by indicating the week it was "assigned" (and recommended completion dates) with its correlating CS Principle or Mission objective. The roadmap was integrated into the course website as another way to scaffold the semester-long learning objectives.

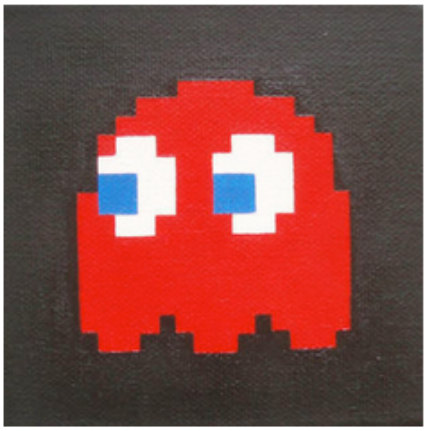
Quests and Labs also provided appropriate scaffolding mechanisms to guide the student through the content. This was done by explicitly listing the objectives for that particular assignment (see *Figure 12*), offering step-by-step guidance when appropriate (see *Figure 13*), and by providing hints for the more complex or challenging lessons (see *Figure 14*). The figures below exemplify how the LMS scaffolds students to complete the Blinky *Programming* Quest, in which they draw the *Pac-Man* character, Blinky, using the Processing programming language:

Processing: Blinky (Part 1)

SEPTEMBER 3, 2014 KARA BEHNKE 1 COMMENT EDIT

Quest Information:

In the old-school game of Pac-man one of the characters was a red ghost called Blinky. Blinky is made up of rectangles of three colors, and the easiest way to draw him is to draw one color at a time, beginning with red.



The purpose of this quest is to give you more experience programming in Processing, in which you will draw Blinky just by using code. At the end of this quest, we will have Blinky looking in different directions.

Quest Objectives:

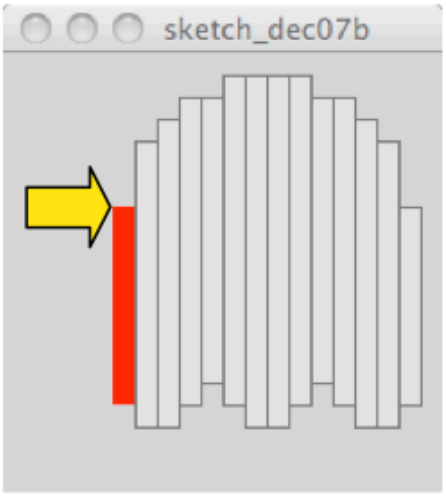
- You will follow the steps below and draw Blinky from scratch using nothing but code!
- **Upload a screenshot** of your completed Blinky (you will have 3 different images by the end)
- **Upload your .pde file** with all your code

Figure 11. Example of Quest/Lab Objective

Step 1: Setting up the Canvas and project

Open Processing and set up a 200×200 size canvas. Be sure to name your project and save it somewhere on your computer. You probably don't want your rectangles outlined, so specify *noStroke*(); and, you will want the *fill color* to be red.

Step 2: Abstracting; what first?



We take each "square" region (as seen in the Blinky picture here) as covering 10×10 pixels on the screen — so his blue irises are 20×20. The easiest way to draw the ghost is to draw vertical strips each of 10 pixels wide.

So, we begin at some arbitrary position on the canvas, say, 50, 70, and begin drawing; the spot corresponds to the yellow arrow marked in the figure. Recall that this position is 50 pixels across and 70 pixels down from the upper left corner. Then we use the rectangle drawing function

```
rect( 50, 70, 10, 90); //From left, bar 1
```

remembering to end the command with a semicolon. Recall from the book Getting Started with Processing (p. 17) that the parameters are

```
rect( <x>, <y>, <width>, <height>)
```

So this is a 10 wide by 90 tall rectangle, positioned so it's upper left corner is at 50,70 on the canvas. It will be the left side of the ghost's body.

Step 3: Drawing the Body

To complete the red body, you will add 10 wide vertical bars properly positioned so the figure comes out looking like Blinky — **be sure to add comments!**

Figure 12. Scaffolding Learning via Quests/Labs

What to Submit:

- **Submit your .pde file with all your code** using the upload button below.
- **Submit a screenshot or video** of your interactive Blinky following your mouse and changing color.

If you get stuck, here are some helpful hints:

```
int x = 1;
int y = 1;
int look = 0;
int dir = 1;  // Dir Right / Left
int dir2 = 2; // Dir Up / Down
int chgColor = 0;









  // Botom of Draw
  x = x + dir;
  y = y + dir2;
`









void mousePressed() {
  if (mouseX < (120 - 70 + x) ) {
    look = 0;
    dir = -1;
  } else if (mouseX > (120 + 70 + x) ) {
    look = 2;
    dir = 1;
  } else {
    look = 1;
    dir = 0;
  }

  if (mouseY < (85 - 65 + y) ) {
    //look = 0;
    dir2 = -1;
  } else if (mouseY > (105 + 65 + y) ) {
    //look = 2;
    dir2 = 1;
  } else {
    //look = 1;
    dir2 = 0;
  }
```

Figure 13. Provided “Hints” & Submission Requirements

Even though the "helpful hints" were provided at the end of the Quest/Lab, students could not simply look up the answer or write the code from what they saw on the lesson page. They had to analyze the information given to them, use that information to construct knowledge, and "think computationally" to solve the problem at hand. This scaffolding structure was implemented into every available Quest/Lab to help guide the students into taking responsibility for their own learning.

4.2. Constructing Mastery

One of the basic principles of gamification is to provide feedback. Feedback gives players information about their performance or about the game state, and with this information the players can change their behavior. Generally speaking, there are three types of feedback provided by games—positive, negative, and neutral (Kapp, Blair & Mesch, 2011). Positive feedback tells the players they are doing well and should maintain their current strategy. An example of positive feedback is showing the player a visual cue of "Achievement Unlocked!" or "Trophy Awarded!" Negative feedback tells the players that they are not doing well and should change their current strategy to be more successful. Examples of negative feedback could be the "Wrong!" buzzer noise or a screen displaying "You Are Dead" in a role-playing game. Neutral feedback addresses the players' current situation or status and gives them information about their particular circumstances. A progress bar indicating level-of-experience or a "tool tips" window are examples of neutral feedback.

In order to provide meaningful feedback to students so that they can adjust their behavior as needed or know their current standing in the course, the LMS for both sections of the course incorporated neutral feedback. Only the gamified section incorporated positive feedback loops

through the use of badges, points, and leaderboard status. Neutral feedback is provided to students through the LMS progress bar and highlighting the completion of course objectives. *Figure 14* and *Figure 15* demonstrate how the LMS provided neutral feedback on student progress for completing course objectives:

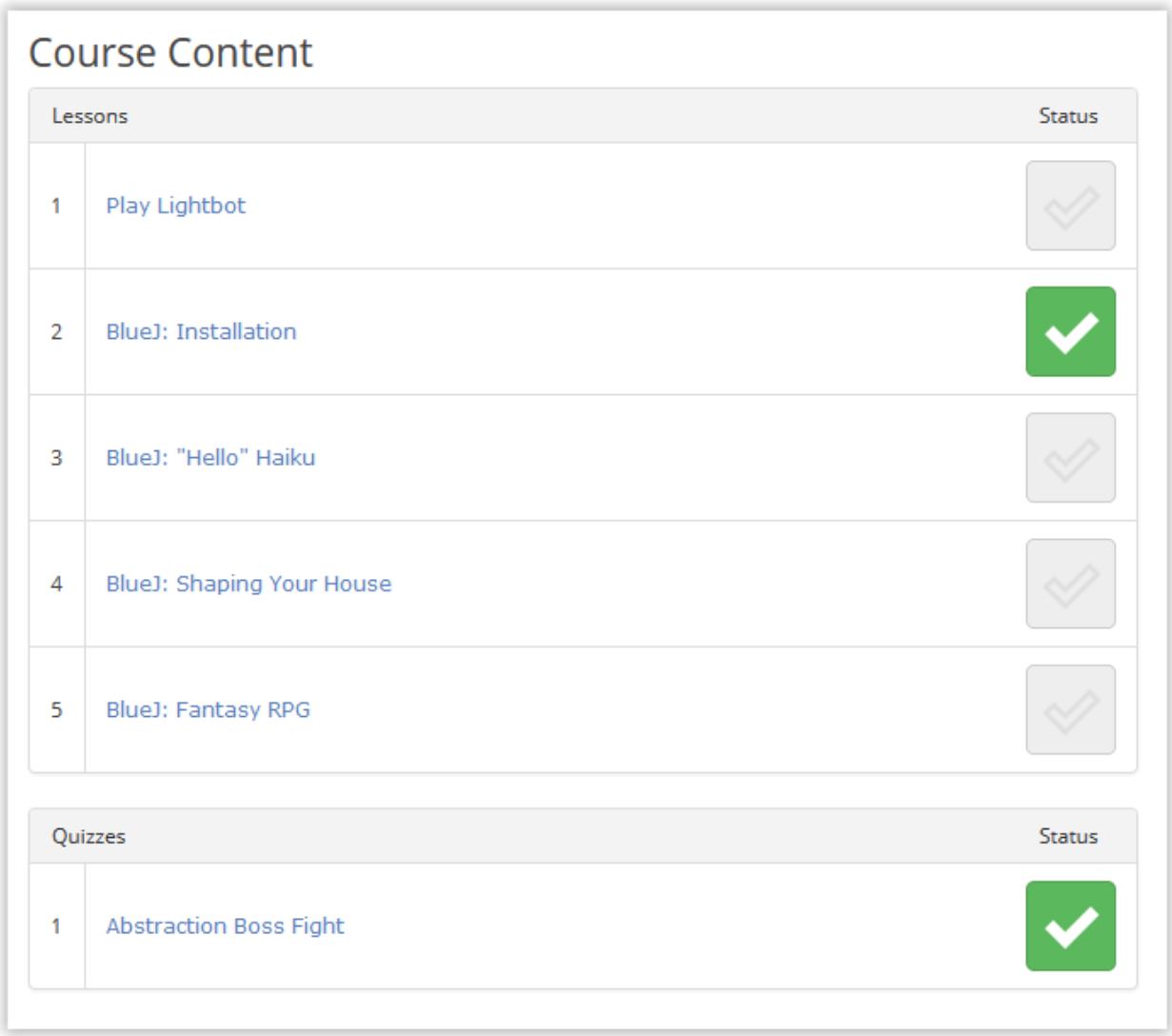


Figure 14. Feedback for *Abstracting* Mission.

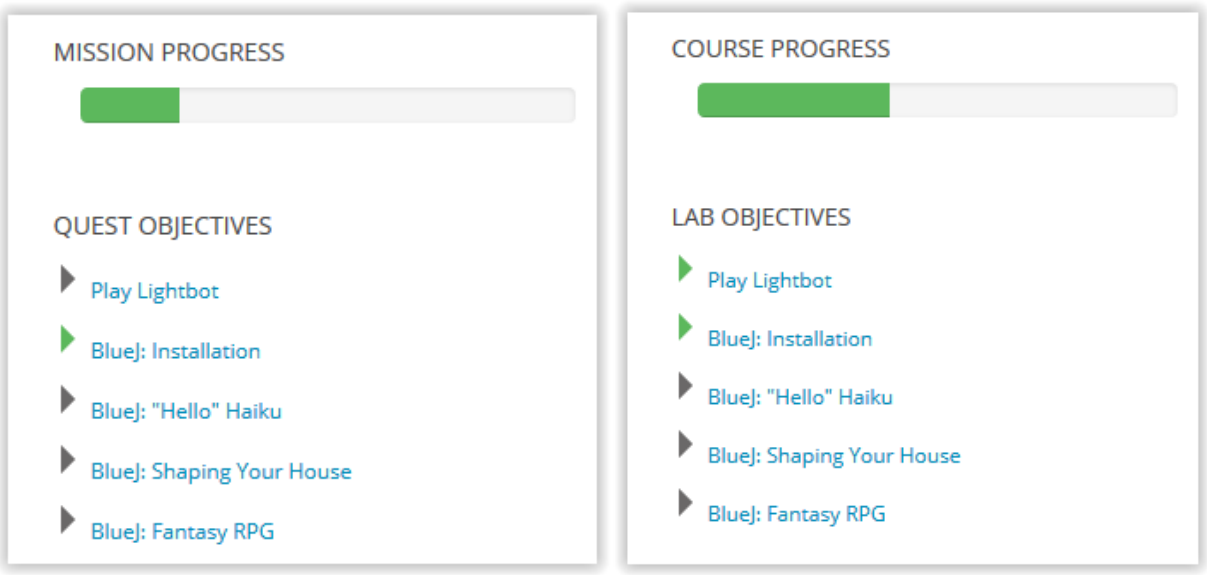


Figure 15. Feedback for the *Abstracting* Mission (left) & CS Principle (right)

Students in both control and variable groups had the same neutral feedback to indicate their completion rate of course objectives. Students were *not* given conventional grades (i.e., A, B, C, D, and F) throughout the semester, but rather were scored on the number of course objectives that they had completed by the end of the semester. The feedback in this LMS system has two options: (1) give the student immediate feedback when they upload or submit an assignment (i.e., the progress bar fills up immediately) or (2) the progress bar and completed objectives only change when a site administrator "approves" the uploaded submission. In an effort to prevent cheating or students "gaming the system," the feedback was updated only after the instructor or teaching assistant approved the lesson submission. This practice allowed the instructor or teaching assistant the ability to "grade" student work, ensuring that students aren't cheating by submitting irrelevant or poor quality work. This neutral feedback employs the constructionist learning principle of "learning through effort."

Students in the variable group were given positive feedback through the gamification prototype. This is the main independent variable of the study. Both sections of the course received neutral feedback, but the use of badges was an effort to measure and assess whether "positive extrinsic feedback" influenced student motivation and academic achievement. This feedback was implemented through the process of students earning badges and receiving the awarded points associated with the badge for completing course objectives. *Figure 16* and *Figure 17* illustrate how badges were earned and displayed on the course website:

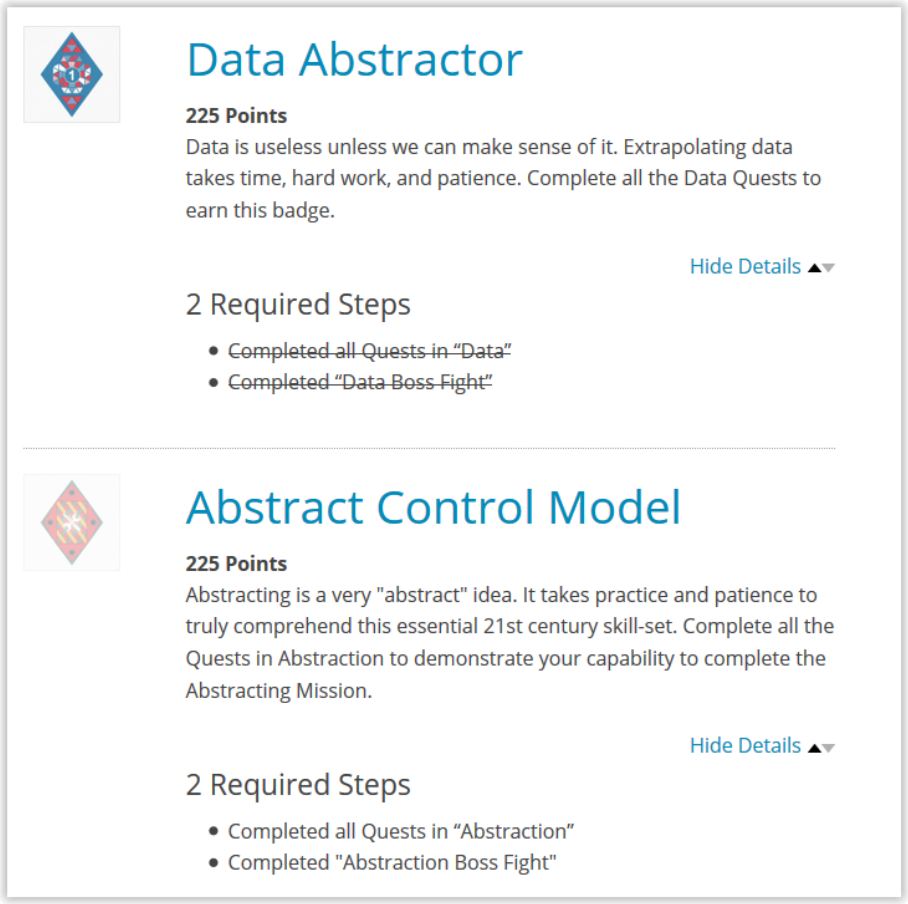


Figure 16. Earned Badges for the *Abstracting* Mission

Note the differences between displays of these two badges. The picture of the badge for Abstract Control Model is semi-transparent, whereas the thumbnail for Data Abstractor is completely opaque. In addition, the "2 Required Steps" to earn the badge for Data Abstractor are stroked-through, indicating that these steps are already completed, whereas they are normal text for the Abstract Control Model badge. These changes in visual cues were part of the feedback for the earning and receiving of badges. *Figure 17* illustrates another way that badges were earned and displayed on the course website:

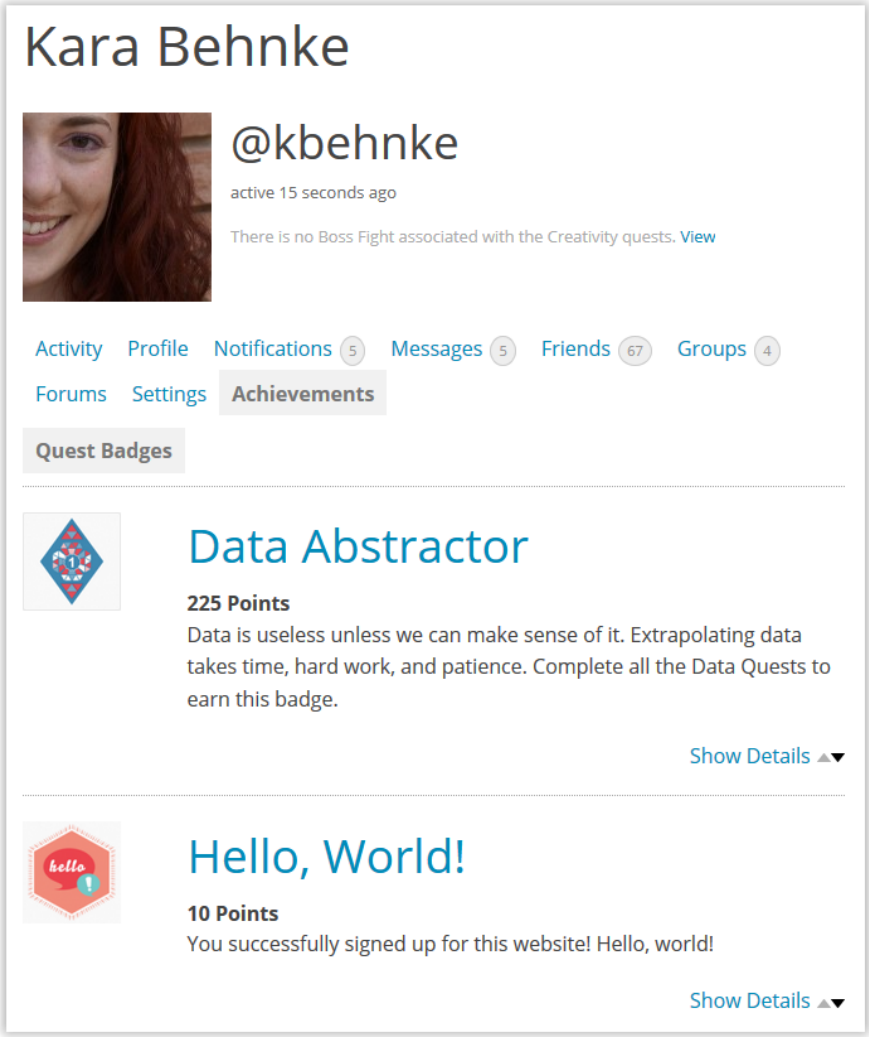


Figure 17. Displayed Badges on Profile Page

Points were awarded to the user according to its badge type. Appendix B details a list of all possible badges students can earn throughout the semester. Points were weighted according to the amount of effort required to earn the badge. Therefore, the major course objectives, such as completing all "big idea" homework assignment and assessment tests, rewarded the number of points. Students were awarded points just for participating in basic aspects of the course—like signing up for the course website. The "badge type" indicates whether a badge is specifically related to the coursework or associated with the community features of the site. For example, it was not required for students to participate in the course forums, but active participation earned participating students badges and extra points, which could raise their leaderboard rank.

A leaderboard is a list of the individuals who have the highest scores or the most points of a given group. It is a list of the "top players" in structural gamification, so whoever else is involved can see everyone else's name or scores. Only the gamified course implemented a leaderboard, which showcased the students with the highest rank and most earned badges and points, as shown in Figure 18.

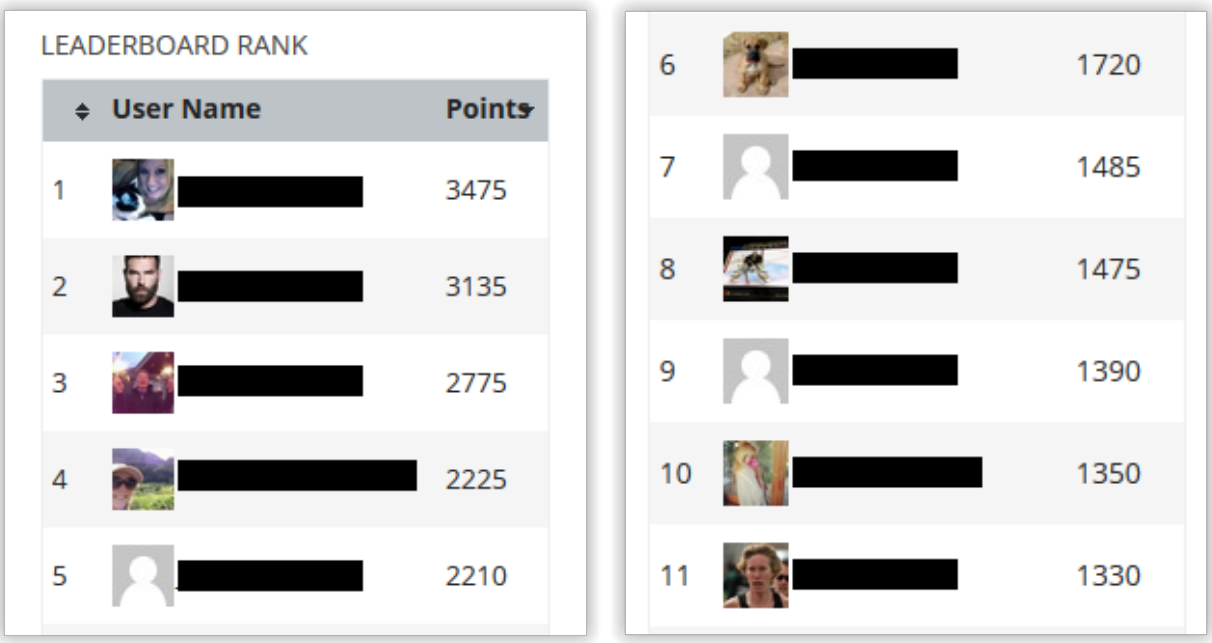


Figure 18. Leaderboard Rank for Variable Group

Leaderboards can be a powerful motivator, as well as a chance to interact socially in discussions around the leaderboard (Nicholson, 2012; Kapp, Blair & Mesch, 2014). Leaderboards can provide "social capital" to the users who are at the highest levels, which can potentially create and support connectivist purpose.

A critical component of "constructing mastery" is learning from failure. Games create and maintain this positive relationship with failure by providing immediate feedback, and keeping the stakes low. Students were afforded the ability to overcome failure or receive "redemption" through the structure of the LMS and course assessment procedures. Students were able to re-take quizzes as many times as they wished without penalty, and could also submit assignments as many times as needed. For the variable group, students who attempt a Boss Fight (online quiz), and failed, receive a badge for failing (see *Figure 19*). This badge was an attempt through gamification to reinforce the concept that failure was to be considered a form of constructive feedback.

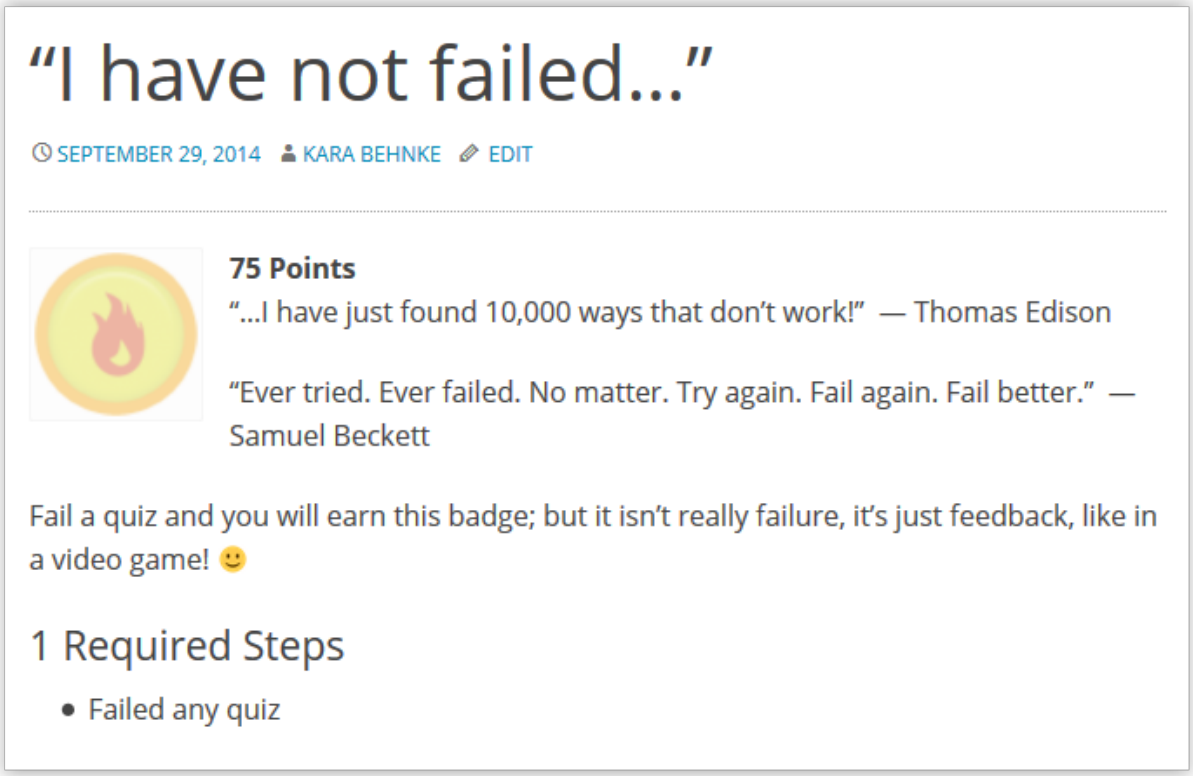


Figure 19. Badge Awarded to Students Who Fail a Quiz

To reinforce that failure was a necessary step on the path toward mastery, the "failure" badge was named after the famous quote by Thomas Edison. Students could retake the online assessment as many times as they wish; the highest scored quiz is counted toward their grade; theoretically, every student had the possibility of earning 100% for every quiz.

Allowing unlimited resubmission assignments the opportunity to take online quizzes as many times as desired, without penalty, were designed to encourage students to master the material, rather than simply submit low quality work to receive a grade. This approach was implemented into the course structure in an effort to promote a growth mindset for students and encourage them to overcome failure and master difficult challenges.

4.3. Connectivist Purpose

As society becomes increasingly interconnected through technology, learning can employ a multi-channel approach where different communication technologies are used to deliver learning content in ways most appropriate to the learning context (Mukhopadhyay & Parhar, 2001). Twenty-first century learners need to recognize when knowledge may no longer be valid so they can acquire new knowledge (Prensky, 2003). "Digital natives," as Prensky (2003) calls such learners, need to stay up-to-date with knowledge and be active participants in the network of learning.

Connectivist theory holds that individuals learn and work in a networked environment (Siemens, 2005). As a result, learners do not have full control over what they learn since others in the network are actively changing the available information, thus requiring new learning, unlearning old information, and relearning current information (Ally, 2008). Connectivism is fundamentally driven by individuals creating purpose and communal meaning within a network.

Students are more likely to be motivated in the learning process when they experience support toward their autonomy, and where they feel connected to and supported by significant others (Ryan & Powelson, 1991). In an effort to create a sense of purpose and social engagement for students, the course websites implemented a connectivist framework similar to conventional social media. Students have the ability to create avatars (upload a picture to their profile), update status on an "activity stream," the ability to "friend" other students or make peer-to-peer connections, communicate through private messaging and public chat forums, and even form "groups" to collaborate with specific sets of users.

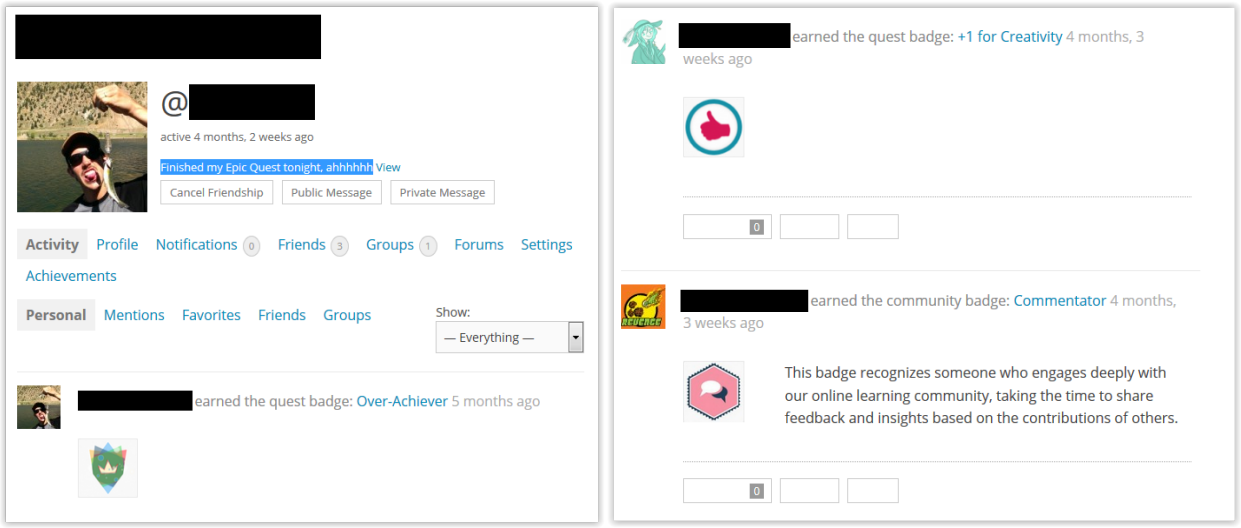


Figure 20. Student Profile & Activity Stream

In addition to a personal profile, the course websites displayed a site-wide activity feed, similar to the manner in which Facebook and Twitter updates their data streams for users to view.

Bulletin-board systems (BBS), more commonly known as forums, were also implemented into both course websites to promote connectivism. BBS functionalities were implemented into both LMS websites to provide a mechanism for students to collaborate and share (see *Figure 21*).

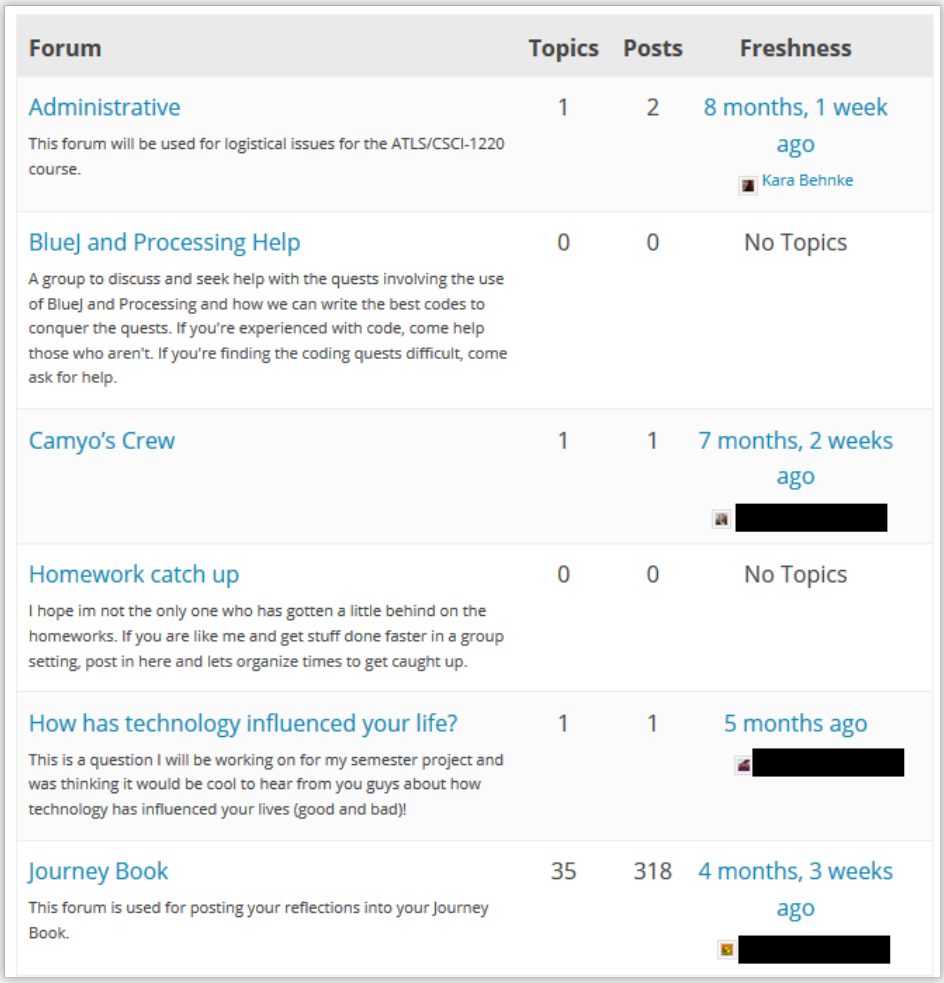


Figure 21. Screenshot of Forums (Variable Group)

Forums allowed users to post to specific "threads" to organize content. Students could also create private or public "groups" that invited specific people to that microcosm. These, BBS functionalities fostered connectivist purpose by affording users the means to communicate and share meaning.

Many gamification frameworks create affordances of competition and collaboration with other members in that community of practice (Nicholson, 2012). Leaderboards, guilds, groups, and other collaboration strategies in games can bring together large, like-minded communities to

promote a sense of relatedness (Richards et al., 2014; Ducheneaut et al., 2007). CS Principles and gamification support and facilitate connectivist networks, which may potentially create a sense of purpose into the implemented environment.

By initializing self-directed learning (autonomy), constructionism (mastery), and connectivism (purpose) into the course structure and design of the LMS websites, I instantiated Self-Determination Theory for the CS Principles curriculum.

V. INSTRUMENTATION & EVALUATION

1. Procedures

This case study implemented a quasi-experimental, proxy post-test design (Rovai, Baker & Ponton, 2014) to test the central hypothesis. Primary data was collected in the form of quantitative and qualitative surveying, student course grades, website metadata, and content analysis.

1.1. Student Course Grades

Because the *Virtual Worlds* case study is situated as a traditional undergraduate course, student course grades were used as a source of quantitative data for analysis. Student grades resulted in a non-normal distribution, which was skewed in a positive direction. Therefore, non-parametric tests were used to evaluate and analyze the grades as quantitative data. This analysis is discussed in more detail in the Findings section of this chapter.

1.2. Proxy Pre-Test & Post-Test

Using the Intrinsic Motivation Inventory (IMI) is an established methodological practice for measuring motivation in Self-Determination Theory research (Deci et al., 1994) and gaming

scholarship (Lieberoth, 2014). The pre/post-test survey employed a five-point Likert scale to compute the dependent variable of "student motivation" into an ordinal,[27] interval[28], and nominal[29] data scales. Proxy pre-test instrumentation is shown in Table 12.

Q9	Which of the following programming languages have you used before this class?
Q10	Have you taken a course related to computer science or programming before college (e.g., high school, middle school, etc.)?
Q11	If you have taken a course related to computer science before college, when did you take it? (e.g., high school, middle school, etc.)
Q12	Before taking ATLS/CSCI-1220, have you taken a computer science or programming course in college?
Q13	Before taking this class, did you think computer science was a creative practice?
Q14	During your enrollment in ATLS/CSCI-1220 this fall, were you also enrolled in another computer science or programming course here at CU?
Q15	Before taking this class, have you ever built a website?
Q16	Before taking this class, have you ever built a desktop computer?
Q17	Before taking this class, have you ever developed a video game?
Q18	Before taking this class, have you ever developed a mobile app?
Q19	Before taking this class, how would you rate your programming skills?
Q20	Before taking this class, how would you rate your level of interest in computer science?

Table 11. Case Study One Proxy Pre-Test Itemization

[27] Ordinal scaling classifies variables into ordered categories (e.g., never, rarely, sometimes, often always), but there is no information about the magnitude of differences between categories.
[28] Interval scaling is assigned to responses to indicate magnitude of difference between items, but there is no absolute zero point (i.e., denotes equal differences between scales)
[29] Nominal scales have no relative ordering of the categories (e.g., sex of a person, color, etc.)

Post-test indicators are shown in Table 13.

Q33	After taking this class, how would you rate your programming skills?
Q34	After taking this class, how would you rate your level of interest in computer science?
Q35	After taking this class, do you think computer science is a creative practice?
Q36	After taking this class, I will enroll in additional computer science classes here at CU Boulder ()
Q37	After taking this class, I have a better understanding of how computing systems work.
Q38	This course broadened my general perspective of what computer science involves.

Table 12. Case Study One Post-Test Itemization

The full documentation to this proxy pre-test and post-test survey is listed in Appendix C of this thesis.

1.3. One Year Follow-Up Survey

During the Fall 2015 semester, I conducted a one year follow-up survey for students who participated in the *Virtual Worlds* course. This survey was used to assess whether the findings in the pre/post-test were reflected in follow up students' actions or perceptions. For example, itemized questions asked if students continued to learn computer science in traditional undergraduate courses, through free online tutorials, in non-traditional forms such as hack-a-thons, etc. The survey was emailed directly to students who had participated in the course the previous year. A total of twenty-six participants responded to the one year follow-up study (28% response-rate of students whom participated in the course). Appendix D contains the full text of the one year follow-up survey. The evaluation procedures for the follow-up survey were similar to the pre/post-test survey, including open and axial coding, and the use of summative and inferential statistics.

1.4. Content Analysis

Web analytics were collected as a primary metric of quantitative and qualitative observation. Information provided by participants through the website, such as user names, profile avatars, number of visits, number of comments, lesson submissions and time-stamps, among others, were represented through descriptive summaries. In addition, website information was included as part of the qualitative content analysis procedures, which included open-coding and axial-coding techniques. Appendix E contains the full code book used for content analysis of these data.

VI. CASE STUDY ONE FINDINGS

1. Summary of Findings

Findings from the first case study indicate that the gamification supported some students' motivation to complete course material, affording a sense of competition and collaboration amongst classmates. The gamification supported both extrinsic and intrinsic student motivations to complete the coursework. However, gamification did not make a significant difference in supporting learning outcomes for the course. Results also indicate that students in both control and variable groups had a bimodal "love-hate" relationship to the self-directed learning focus of the course. Students both positively and negatively reported on the impact that self-paced learning had on their grades. These finding suggests that self-determined learning is most effective for students with high autonomy and intrinsic motivation, but not for students who are motivated solely by extrinsic motivations.

Findings also indicate that CS Principles had a positive effect on students in both control and variable groups, indicating that the CS Principles curriculum framework supports student

interest and motivation to learn introductory computing. In addition, CS Principles enabled students to consider CS as a creative field of practice, fostered more positive attitudes about CS as a field of practice, and increased the level of interest for students to pursue learning computer science in multifaceted ways. Therefore, CS Principles as a curricular framework was observed to be a success in this case study. However, many students reported that the course was "CS-lite," in that they were expecting and wanting more assignments related directly to programming. Nevertheless, students reported overall positive attitudes about the course and CS as a field of practice. The findings from the follow-up survey suggest that CS Principles established a strong foundation for students to pursue CS as a major field of study or to use it as a tool to support other academic interests. To this end, *Virtual Worlds* accomplished the overarching goals of the Computer Science Principles framework.

Although gamification positively impacted some students in the course, the CS Principles framework itself had the most impact on student motivation and interest learning in introductory computing. This suggests that CS educators should focus explicitly on creating engaging, creative content that focuses on the breadth and impact of CS on society, and that the implementation of gamification should be a secondary priority. My findings suggest that CS Principles is a suitable framework to pursue these efforts. Nevertheless, the students that did enjoy the gamification elements did very well in the course overall, suggesting that highly motivated students benefited the most from gamification techniques. Gamification, therefore, acts as a lens to enhance the impact of pedagogical content, but does not create engaging pedagogical content in and of itself. This result reinforces Vygotsky's (1978) analogy of using play as a magnifying glass to focus learning.

In addition, the CS Principles curricula broadened students' perceptions about computer science as a field of practice. For example, pre-test and post-test showed that the curriculum made a significant impact on their perceptions of CS—that it is creative, socially important, and supports interdisciplinary endeavors. Further, students indicated both during the course and one year following the course that they continue to pursue the study of CS in multifaceted ways, including changing their major to CS at the university, learning it on their own via online training, and embedding computational thinking practices in their personal productivity.

In summary, CS Principles is an effective introductory course for students who do not know much about computer science as a field and practice, creating opportunities for students to pursue computing as both a major field of study or to continue to learn CS after the course had finished.

The remainder of this chapter details these results.

2. Finding 1: Gamification Supports Student Motivation

2.1. Gaming Metaphors Make the Course More Interesting

Overall, the gamification of the course was well received by students. Of the total participants surveyed from the variable group (n=51; 77% of total students enrolled in the variable group), 61% of students agreed that the gaming metaphors of the course, or the terminology of Quests, Missions, Boss Fights, etc., made the course more interesting. 33% of surveyed students felt neutral about the gaming elements, whereas only three students (6%) reported that they disliked the gaming elements. These results are depicted in the figure below.

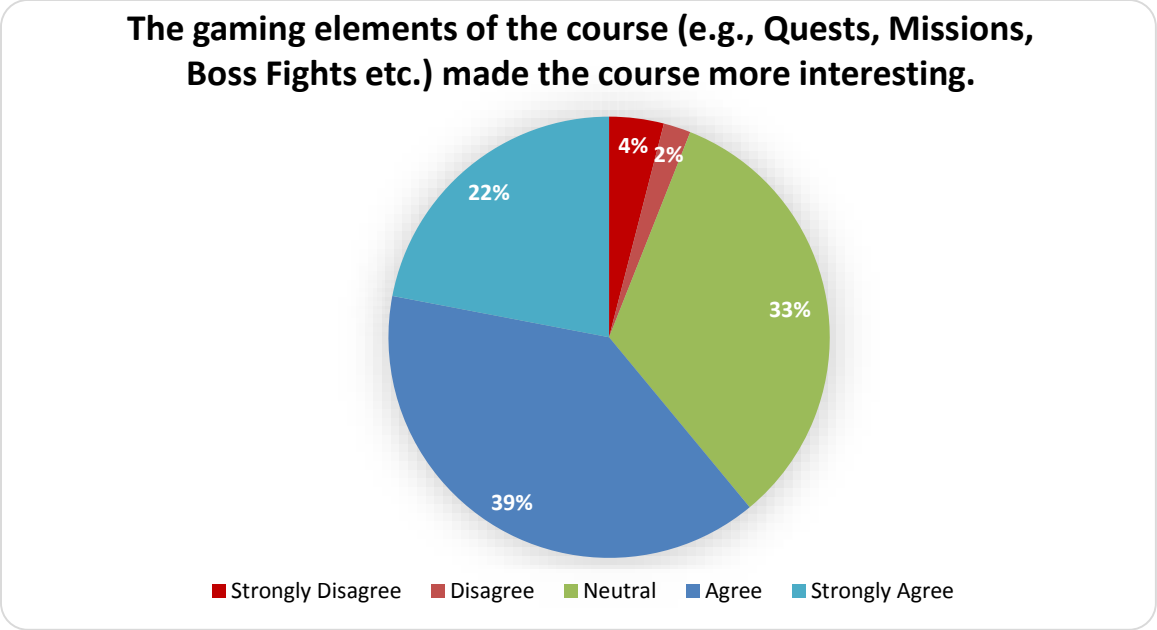


Figure 22. Item Q50: 61% Approve of Gaming Metaphors

This finding indicates the using gaming nomenclature instead of traditional academic terms (i.e., homework, labs, midterm examination, etc.) helped make the course more playful and interesting for some students. Students' comments supported the finding that for some students, learning made the course more engaging [emphasis added]:

> *"[I liked] taking a **different approach to traditional methods** of teaching a class, **self paced** homework, creating an online **class community.**"*
>
> *"I really like the way [the website was] formatted. The **quests are motivating** and the **forums make it interactive**."*
>
> *"I liked the idea of grading us based on if **we finish all our Quests** and then having two major assignments (**Epic Quest**, Midterm)."*

Interestingly, the students who made these comments in the course evaluation open-responses all received A's in the course.

However, some students reported that they did not like collecting badges or gaining rank on the leaderboard, even though they reported that the gaming terms made the course more interesting. Since *interest* and *enjoyment* are scales used to measure intrinsic motivation, this suggests that using gaming metaphors are a form of "playful design" that can support student motivation. Badges, on the other hand, were not as effective in supporting student motivation as initially hypothesized.

In addition, some students implicitly indicated that gamification didn't go "far enough" toward a game-based learning environment as they might have hoped [emphasis added]:

> *"…I was **excited** to have **a class more like a video game**, but this simply **renamed the same tasks. that didn't really excite me**"*

> *"I believe that the **gaming elements** in the course in regard to **labeling things mission, quests**, etc was **confusing** and **hard to equate** to the course."*

> *"The **gaming elements were interesting, but were underused**. What I mean by this is simply that calling assignments **"quests" or "boss fights" isn't really changing anything**. I didn't mind the elements, but I think there's **missed potential there**."*

These data points support previous work that *gamification* remains "superficial," in that it does not promote the best affordances of video games (Bogost, 2011). This finding provides additional insight, identifying a distinction between game-based learning and gamification. Arguably, game-based learning implements more immersive game elements, such as the use of narrative, fantasy themes, character creation, skills, guilds, experience points, among other elements (Schutter & Abeele, 2014). Regardless, the "superficial" gaming elements, such as badges and the use of leaderboards, were effective for some students.

2.2. Badges Effective for Some, Not All

Badges were effective for some students as a gaming element to support student motivation and to complete learning outcomes. Some students (44%) reported that they liked collecting badges on the course website. In addition, over a quarter of surveyed participants (27%) indicated that the badges helped motivate them to complete coursework. The two figures below summarize student perception and action about the badges.

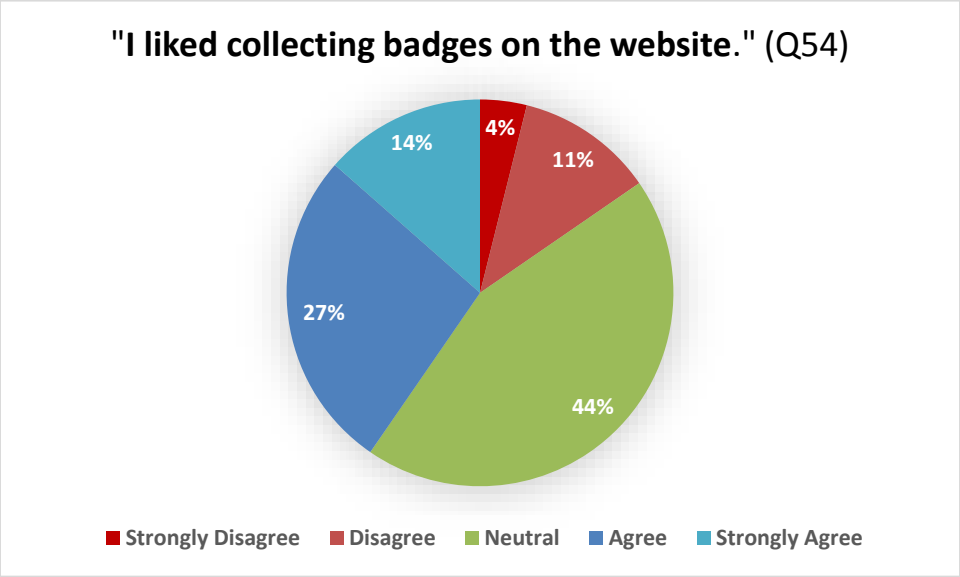


Figure 23. Summary of Responses to Item Q54 in the Variable Group

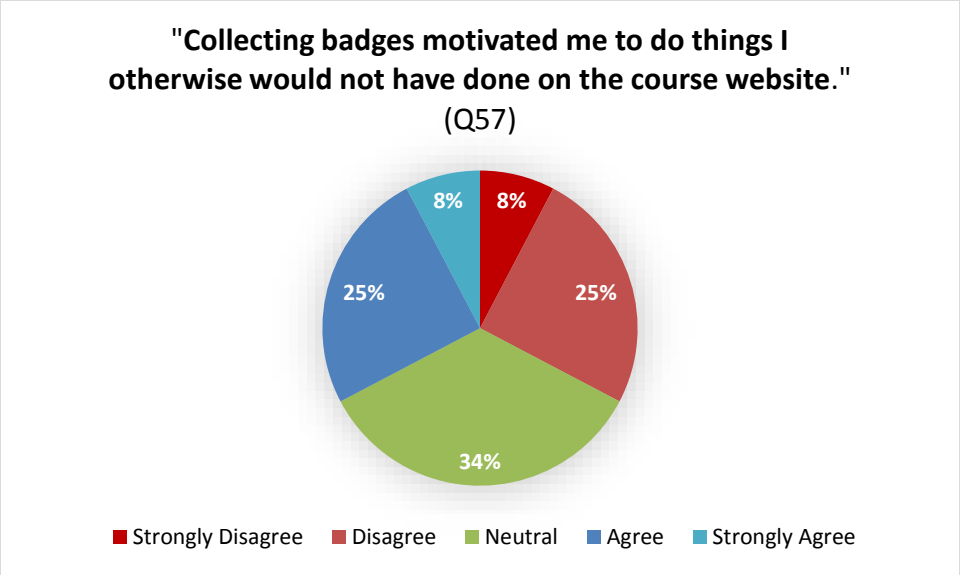


Figure 24. Summary of Responses to Item Q57 in the Variable Group

Survey results indicate that 34% of students reported that they were confused about how the badges and point system worked on the website. Even though the median and the mode for Q49 (i.e., "The gaming elements of the course [e.g., Quests, Missions, Boss Fights etc.] made sense to me") was a 4 or an "agree" statement, responses for Q47 (i.e., "The badges and points on the website were confusing") were not as positive and somewhat contradicted Q49 responses. Q47 used a reverse-scale itemization, in that "strongly agree" indicates a negative finding and "strongly disagree" indicates a positive finding. The figure below summarizes Q47 results.

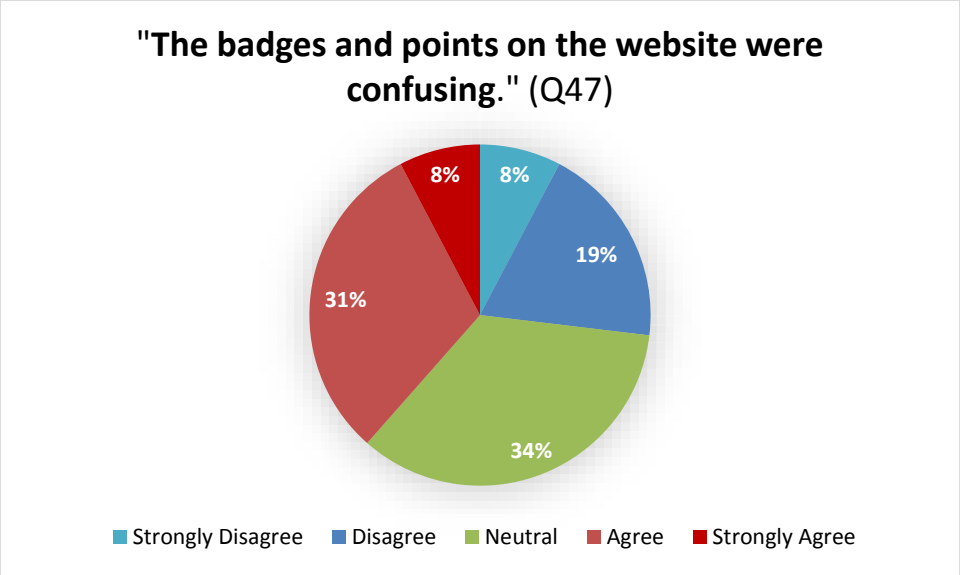


Figure 25. Summary of Responses to Item Q47 in the Variable Group

The mode for Q47 was a 4 "agree" and the median was a 3 "neutral." To further illustrate the problem of badge ambiguity, one student stated that,

*"I **didn't** really **realize** that the **badges were that big of a deal** [or] **recognize them**."*

Overall, the impact of badges on student motivation and learning outcomes was mixed; some students reported that the badges made sense, that they liked collecting them, and that this helped motivate them to complete the coursework. Most of these students achieved high grades in the course (A's and B's). However, almost half of students felt indifferent or negative towards badges or did not believe the badges facilitated their motivation to complete the coursework. This finding suggests that students would benefit from more clarity about the badge-point process. In order for badge-collecting to be successful, students may need to be explicitly told in class[30] or through other means about the purpose of badges and points. In addition, website features such as

[30] I did not explicitly explain how badges or points were used in an effort to observe whether students could figure this out on their own (as would be typical in a game play situation).

"pop-ups" could help facilitate this purpose—for example, when a student received or was awarded a badge, a window could pop up on the screen and say, "Congratulations, you earned this badge!" This common gaming feature is used in many online gaming communities[31]. In hindsight, the implementation of this could ameliorate confusion about the badge process.

2.3. Leaderboard Facilitated Competition

Attitudes about the leaderboard were generally positive; results were normally distributed. Most students "agreed" that the leaderboard brought a sense of competition to the course; however, not all students liked competition of the course. Most felt "neutral" whether they liked the use of a leaderboard. However, 41% of students indicated that raising their rank on the leaderboard motivated them to complete coursework throughout the semester. The figure below summarizes the results about leaderboard items in the survey.

[31] Popular gaming frameworks such as Xbox Live, Playstation Network, and Steam all overlay a window that an achievement, trophy, or award was earned after completing a certain task.

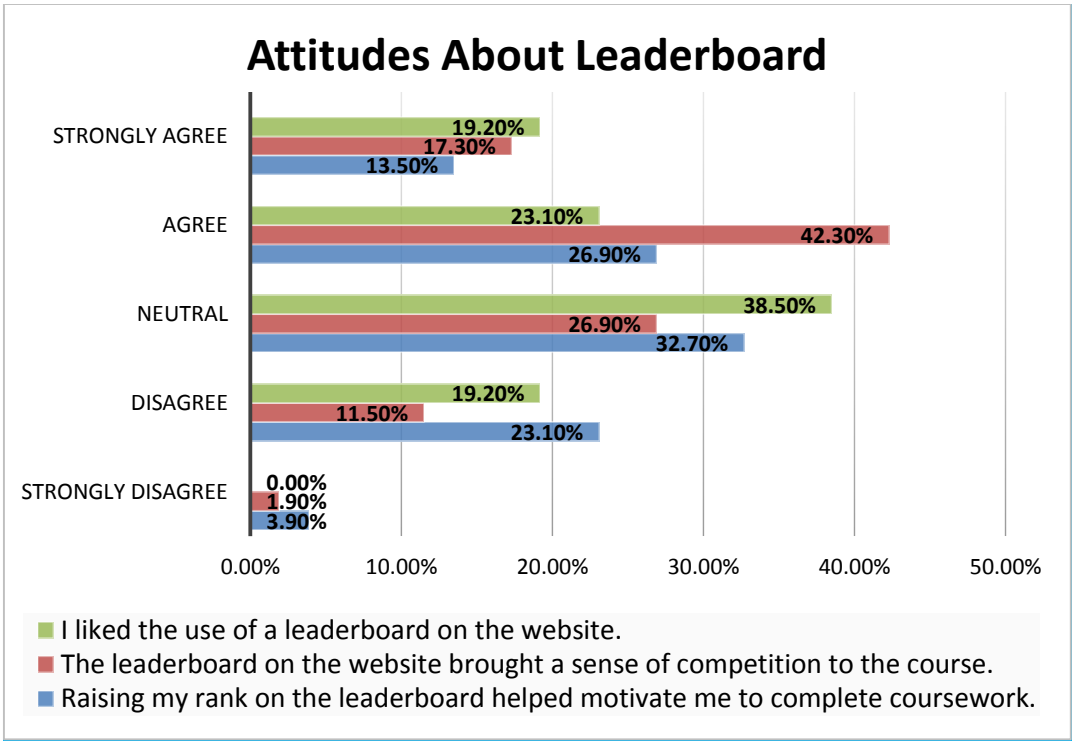


Figure 26. Summary of Leaderboard Attitudes for Q52, Q53, and Q56

The median and mode for Q52 (i.e, "I liked the use of a leaderboard on the website") was a 3 or "neutral." The median and mode for Q53 (i.e., "The leaderboard on the website brought a sense of competition the course") was a 4 or an "agree" statement. The median and mode for Q56 (i.e., "Raising my rank on the leaderboard helped motivate me to complete coursework") was a 3 or "neutral." Some students clearly benefited from the leaderboard use, as the following student comments reflect [emphasis added]:

> *"[getting onto the leaderboard] it's just kind of a **sense of accomplishment**."*
>
> *"**Motivating myself** to turn everything in and **get a spot on the leaderboard helped me**."*
>
> *"I… like how it was kind of a **competition to get more points** on the scoreboard than other students."*

No students reported that they strongly disliked the use of a leaderboard on the course website. However, two students "strongly disagreed" that raising their rank on the leaderboard motivated

them to complete coursework. It is difficult to assess if this had a negative impact on their ability to do coursework or if they were simply indifferent to its use.

2.4. "A" Students Strongly Benefited from Gamification

Seven students were observed to have had strong "positive gamification scores," (for questions Q34 through Q57 – questions about liking the course curriculum and gaming elements) they indicated "Strongly Agree" for the majority of these questions on the survey. Of these seven students, three were women (43%). These seven students all had outstanding grades in the course, with an average course grade as a 97%; two of these students scored over 100% in the course, indicating that they also completed "Side Quests" or extra credit material. Throughout the semester, these students consistently remained in the top ten of the Leaderboard, with some variation throughout the semester. The first and fourth students on the Leaderboard were female. None of these women self-identified as "gamers," leading to the next finding of the study.

2.5. You Don't Need to be a "Gamer" to Benefit From Gamification

Questions were used to identify gaming habits of students in order to determine if being a "gamer" or "non-gamer" had any impact of the tendency to favor the gamification in the course. Figure 27 below summarizes the current gaming habits of students in the control group:

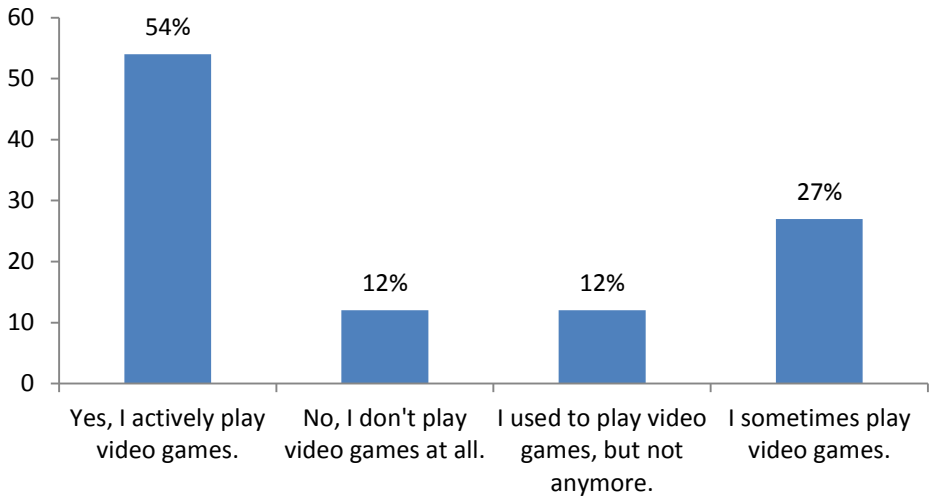


Figure 27. Summary of Q21 Results: Participants' Overall Gaming Habits

Figure 28 summarizes the percentage of students who self-reported as "gamers."

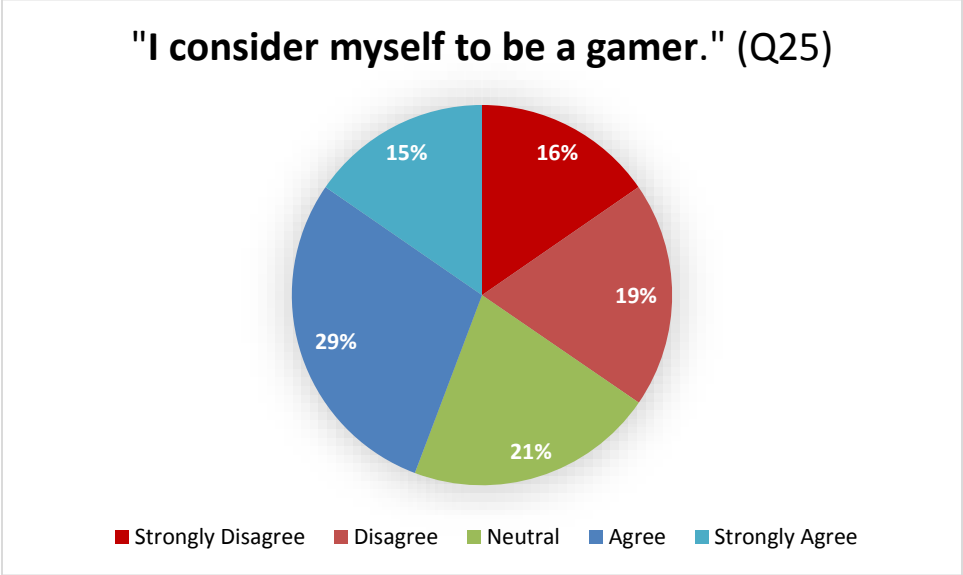


Figure 28. Summary of Q25 Results: Self-Identified Gamers

Participants were considered to be "gamers" if they indicated a "Strongly Agree" for Q21[32] and Q25[33] items. Using this metric, eight participants were identified as "gamers," six participants were identified as "non-gamers."

A Chi-Square test was used to determine if there was a correlation (positive or negative) between gender and being identified as a "gamer" in the variable group.

	Gamer	**Non-Gamer**	**Marginal Row Totals**
Female	5 (8.8) [1.64]	11 (7.2) [2.01]	16
Male	17 (13.2) [1.09]	7 (10.8) [1.34]	24
Marginal Column Totals	22	18	40
Chi-Square	6.0774		
***p*-value < 0.05**	0.0137		

Table 13. Chi-Square test of Male/Female Gamers & Non-Gamers

As shown in Table 13, males were more likely to identify as "gamers," whereas females were more likely to self-report as "non-gamers."

Other questions[34] further characterized "gamer" identification. For example, the men and women who self-reported as "non-gamers" still identified other acts of participation in gaming culture, such as engaging in gameplay with friends, family members, and even listed titles of games

[32] Q21: "Do you actively play video games?"

[33] Q25: "I consider myself to be a gamer."

[34] Q26 through Q31 identify other gaming habits.

they actively played in their spare time. The median and the mode for item Q49[35] in the survey was a 3 (e.g., "Agree"), indicating that the constructs used in the course were generally recognizable. These findings suggests that participants do not have to be a gamer to understand or even enjoy the gaming elements of the course. Gaming culture is pervasive, so gaming constructs are likely to be generally recognizable even if an individual is not prepared, for whatever reasons, to assume the identify of "gamer."

Regardless whether students self-identified as gamers or non-gamers, participants generally reported that they found the gamification of the course enjoyable. As one "gamer" stated:

> *"I was interested in Computer Science* ***the technology world really interests me****, and gaming (the way we interact with technology) definitely interested me*. ***I was excited to see how a game could teach in the modern classroom****."* [emphasis added]

In addition, two of the female non-gamers "strongly agreed" that the gamification made the course more interesting (Q50), that they liked the use of the leaderboard on the website (Q52) and that the leaderboard brought a sense of competition to the course (Q53), that raising their rank on the leaderboard helped motivate them to complete the coursework (Q56), and that collecting badges helped motivate them to do things on the website that they otherwise would not have done (Q57). These particular students also received some of the highest grades in the class, and thereby, were ranked "top five" on the course leaderboard (and consistently throughout the semester). In one of the open-ended questions, a high-achieving, female, non-gamer stated:

> *"Even though I* ***don't play video games****,* ***I really liked the video game setup*** *of the website and assignments."* [emphasis added]

This female student went on to say that:

[35] *Q49:* The gaming elements of the course (e.g., Quests, Missions, Boss Fights etc.) made sense to me.

> *"[the course] was a lot of work, but it was fine and **I learned something from every quest."*** [emphasis added]

This text was coded to indicate that there was a *high level of challenge*, but it appropriately met the student's skill level, resulting in that something was "learned from every quest." Another high-achieving, female, non-gamer student stated that,

> *"I **like the course set up** and how it **encouraged students to actually learn the material** instead of having to complete assignments quickly to meet a deadline."* [emphasis added]

This statement was qualitatively coded to indicate that the course website "encouraged students to actually learn the material" (constructionism) instead of "having to complete the assignments quickly to meet a deadline" (self-directed learning).

2.6. Gamification Did Not Have a Statistically Significant Impact on Learning Outcomes

Course grades were used as a measure for the dependent variable "learning outcomes." The average final grade for the variable group was 81% (i.e. B-), whereas the average grade for control group was a 73% (i.e., C). The difference between the mean grades was Δ=7.46%. In the variable group, thirty-two students (48%) received an A in the course; in the control group, eleven students (39%) received an A in the course.

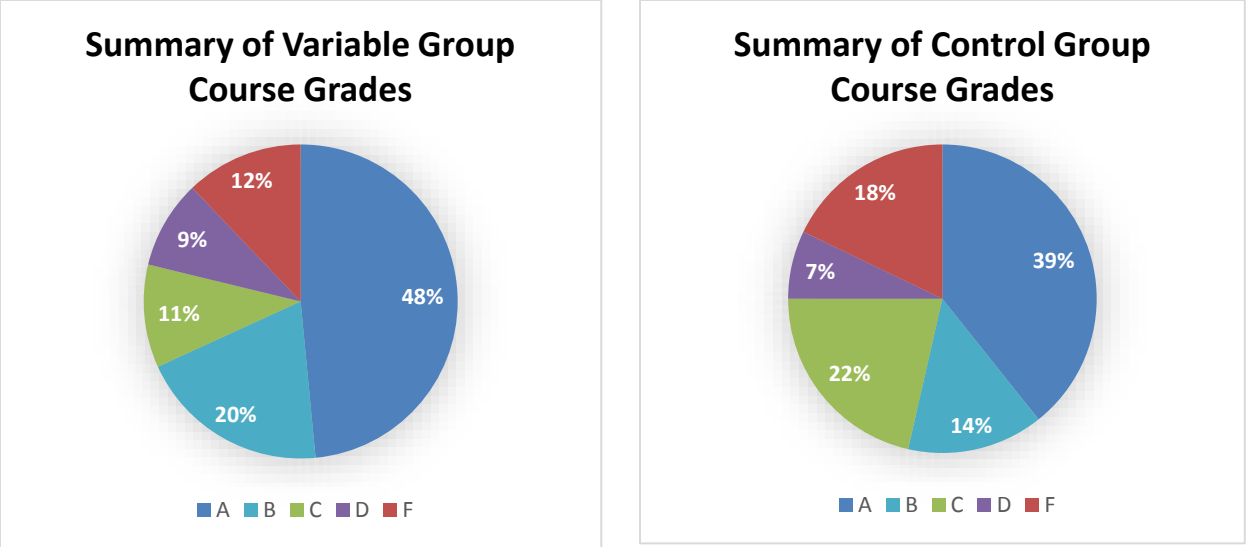


Figure 29. Course Grade Breakdown for Variable Group (left) and Control Group (Right)

Eight students in variable group failed the class; five students in the control group failed. Some failing students did not register for the course website, reflecting a complete lack of participation in the class. The majority of students in both groups received an A or a B in the course. Figure 30 shows the overall grade summary for the control and variable groups.

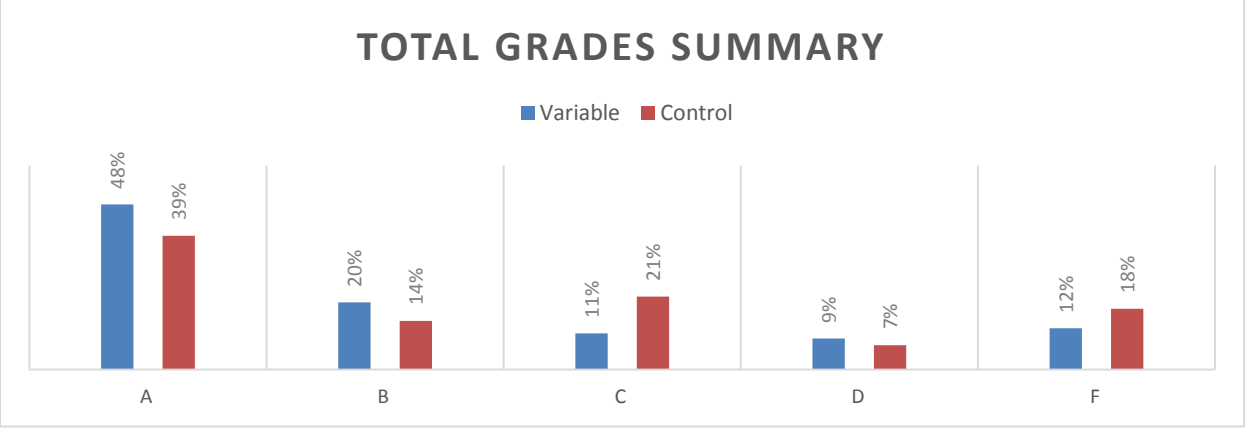


Figure 30. Total Grade Summary Variable and Control Groups

The course grades in both groups have a non-normal distribution; it is skewed in a positive direction, as shown in Figure 30. To determine if there was any statistical difference between course grades in the control and variable groups, the non-parametric Mann-Whitney U test, depicted in Table 14.

Mean Ranks for Sample A	Mean Ranks for Sample B	U Value	$P_{(1)}$ (one-tail)	$P_{(2)}$ (two-tail)	Z-value
49.8	42.2	$U_A = 775.5$	0.1112	0.2225	1.2236
$n_a=66$	$n_b=28$	$U_B = 1072.5$			

Table 14. Mann-Whitney U Results ($p \leq 0.05$)

Using a two-tailed test, The U-value was 775.5, indicating that this distribution was approximately normal. Therefore, the Z-value above can be used. Because the p-value is 0.2225, the result is not considered significant. Therefore, gamification did not have a statistically significant impact on learning outcomes (as measured by student course grades).

3. Finding 2: CS Principles Positively Impact Student Motivation

Our findings indicate that CS Principles positively impacted student motivation in learning computer science. This result was determined by first coding and analyzing open-ended questions from the mid-course survey. Emergent categories and themes indicated that students had shifting attitudes about CS as a result of the course. Next, proxy pre-tests and post-test responses measuring overall student interest in CS (Q20, Q34) were analyzed, and triangulated by qualitative data survey responses.

3.1. Students Motivated to Learn in the Course

Survey results for item Q43 show that students from both sections of the course were motivated to learn in this course. These responses are normally distributed, as shown in Figure 31.

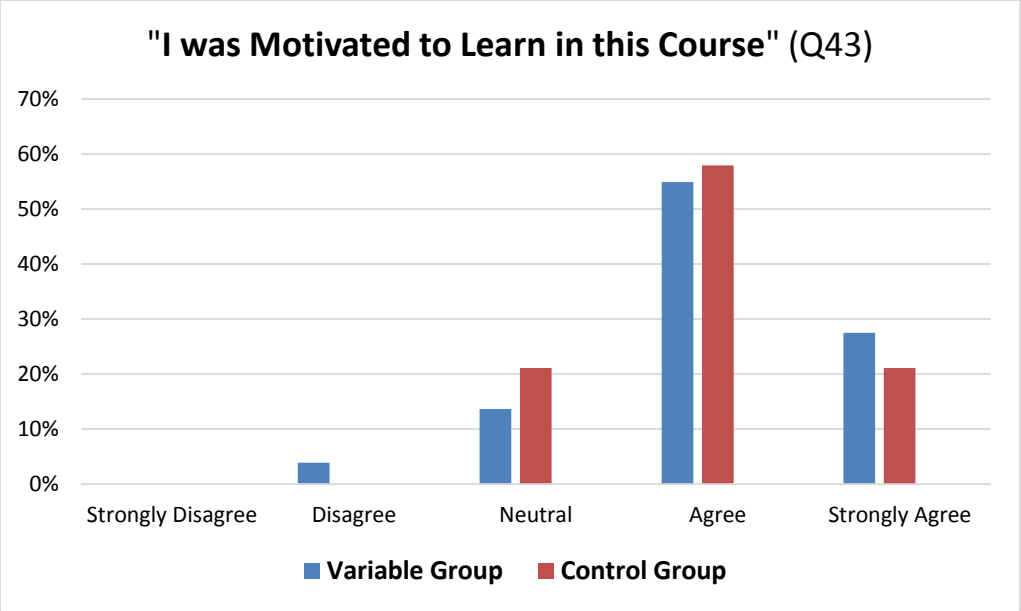


Figure 31. Summary of Q43 Responses for Variable & Control Group

These data suggests that "student motivation" was derived more from the course content itself rather than the gamification, since there is no significant difference in this measure between the groups. Thus the CS Principles curriculum itself had a more direct impact on student motivation than the gaming intervention. Although *Virtual Worlds* is an elective course and students self-select to enroll in the course, it is likely that students already had some intrinsic motivation to enroll in the course. However, the data suggest that CS Principles also changed student attitudes and levels of interest in computing, in addition to sustaining intrinsic motivation in the course.

3.2. CS Principles Improve Student Attitudes About CS

Our results suggest that the CS Principles curriculum promoted a positive attitude for students toward computer science. Midterm extra-credit survey questions (see Appendix F) were transcribed, categorized, and coded by hand to explore this question. Eighty students participated in survey (85%). First, survey responses for ATLS-001 students were coded, and then the same codes were applied to CSCI-001 students. Emergent themes in the data required additional codes. These codes were then applied to the ATLS-002 and CSCI-002 groups of students. After these

codes were applied, categories of data emerged with the second pass of coding. Table 16 summarizes the major categories and emergent codes; Appendix E shows the full list of final codes used to survey overall attitudes of students toward CS.

The dominant category most frequently referred to in student open-ended questions were attitudes, knowledge, and definitions of CS.

Major Categories	Final Codes
Attitudes toward CS	fun, interesting, exciting, like, cool, enjoyable, good, useful, beneficial, important, doable, afraid, hard, confusing, for nerds, bad, for other people, certain people, no opinion, change, no change
References	major, job, family, friend, website, previous experience, important person, hobbies, using computers, other fields
Knowledge Identifiers	math, jargon, code, previous experience, change, no change, concepts, sub-discipline, uses, capabilities, online class, homework, self-work, hardware, fluency, broad, interest, humanities, impact, powerful
Recommend	yes, no, neutral, you, specialize, everybody, other students, people, society, CS major, CS interest, TAM (Technology, Arts & Media), general population
Lectures	relevant, interesting, informative, lots of work, more code, useful, beneficial, introductory CS, computers, code, society
Definitions of CS	code, math, logic, not code, better than other fields, creative, sub-discipline, uses, capabilities, hardware, fluency, CS lite, impact, other fields, intro, relative, society

Table 15. Major Categories and Emergent Codes for Midterm Evaluation Survey

Coded data indicated that students had shifting perceptions about computer science. These shifting perceptions included: attitudes before the class to current perceptions; current perceptions to future perceptions; and before the class to future perceptions. These shifting perceptions examined what the student thought of CS and computing before the class started, a "current" point in time (i.e. a mid-point analysis during the midterm examination) and what they thought of CS outside of the class in the "future." In the proxy pre-test, "before" data is recall data. The midterm

was given during Week 13 of the semester. The post-course surveys also used proxy post-test indications. The emergent themes from this coded data are:

- **Positive Change** – students thought negatively of CS before, but now they think positively of CS
- **Positive Same** – students thought positively of CS before and still think positively of it now
- **Negative Change –** students thought positively of CS before, but now they think negatively of CS
- **Negative Same** – students thought negatively of CS before and still think negatively of CS

The final categories of codes were (1) attitude, (2) definitions, (3) and knowledge identifiers about CS. These categories were re-coded to include the positive/negative and change/same codes. The table below summarizes the results of coded data:

32 ATLAS Students	**48 CSCI Students**	**Total # of Codes**
• 38 *attitude* codes • 33 *definition* codes • 30 *knowledge identifier* codes	• 50 *attitude* codes • 54 *definition* codes • 36 *knowledge identifier* codes	• 88 *attitude* codes • 87 *definition* codes • 66 *knowledge identifier* codes
• 29 ***positive*** *attitude* codes • 9 ***negative*** *attitude* codes	• 39 ***positive*** *attitude* codes • 11 ***negative*** *attitude* codes	• 68 ***positive*** *attitude* codes • 20 ***negative*** *attitude* codes
• 16 ***positive*** *definition* codes • 19 ***negative*** *definition* codes	• 34 ***positive*** *definition* codes • 20 ***negative*** *definition* codes	• 50 ***positive*** *definition* codes • 37 ***negative*** *definition* codes
• 13 ***positive*** *knowledge identifier* codes • 17 ***negative*** *knowledge identifier* codes	• 18 ***positive*** *knowledge identifier* codes • 18 ***negative*** *knowledge identifier* codes	• 31 ***positive*** *knowledge identifier* codes • 35 ***negative*** *knowledge identifier* codes

Table 16. Summary of Coding Scheme per Group (*n*=80)

Analysis after coding indicated that 77.3% of students had a positive *attitude* towards CS (either a changed-to-positive or stayed-positive). 57.5% of students had a positive *definition* of CS. Interestingly, ATLAS students had more negative definitions of CS than CSCI students (19 more codes than 16), while CSCI students had many more students with a *positive definition* than a *negative definition*[36] of CS (34 positive and 20 negative). In addition, 47% of students had a *positive knowledge identifier* about CS—an example of a positive knowledge identifier would be "I like to build computers." Presumably this student would expect to be good at CS because of this interest and experience. 53% of students had a *negative knowledge identifier*, a student who assumes that one has to be good at hardware in order to be good at CS, leading to a negative knowledge identifier. Overall, the definition of "computer science" by students was often closely tied with specific knowledge identifiers and student perceptions and attitudes about their own ability and skill level in that knowledge.

In the CSCI student group, there were nearly twice as many students who had a *positive change in definition* of CS (22 versus 12, 13, and 7) than the students who had a *positive same*, or *negative* attitude about CS. This is somewhat surprising, since the CS students seem to have been affected more than the ATLAS students, even though it must be expected that the CSCI students should already have an idea of what CS involves as a field of practice.

To summarize, emergent codes from the qualitative data indicate that CS Principles helped give students more positive attitudes about computer science. Not all experienced a change, however. Some students reported that the course was "CS-lite" or not "real CS" (this was one of the original codes; five students mentioned this explicitly). These students were expecting a class focused more explicitly on programming than the broad fundamentals presented by CS Principles.

[36] A "negative definition" of CS would include codes like "nerd," "not for me," and other indicators that did not suggest the creative, social, or collaborative side of CS.

That said, most students experienced a positive shift in attitude about CS as a field of practice as a result of the CS Principles curriculum, as discussed below.

3.3. CS Principles Increase Student Interest in CS as a Field of Study

Students not only improved in their attitude about CS, their interests in CS as a field of study also increased. In the proxy pre-test and post-test survey, items Q20 and Q34 sought to measure whether the course increased their interest in CS. Figure 32 summarizes the results from students in the variable group.

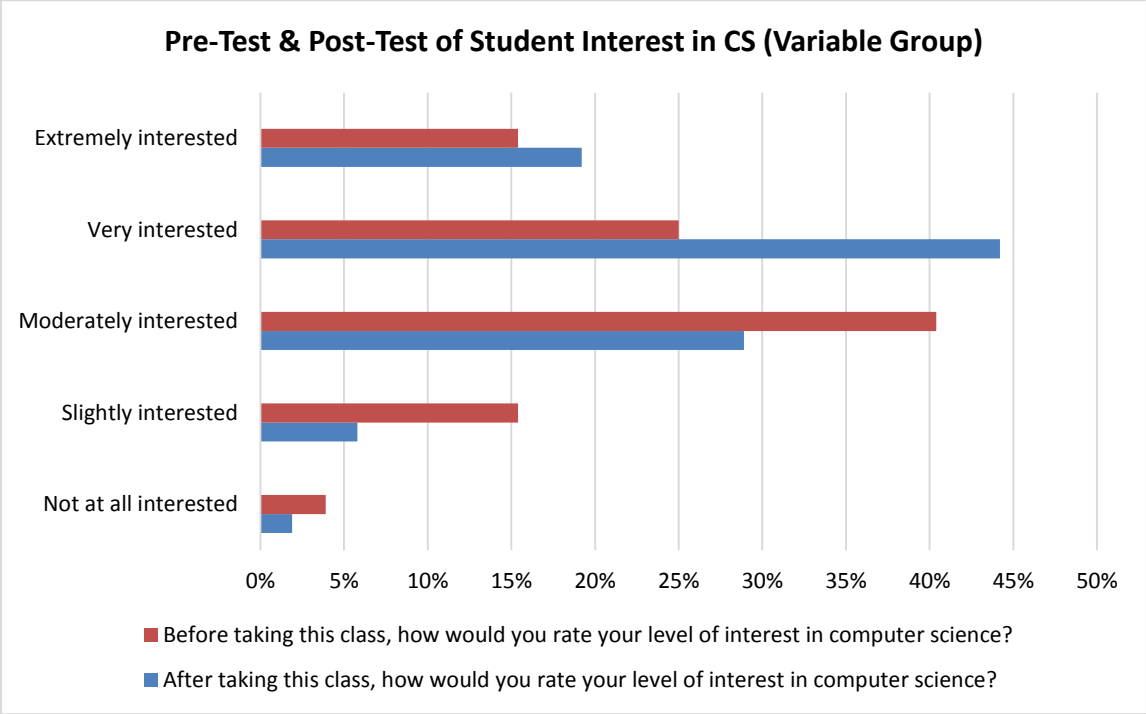


Figure 32. Pre-Test & Post-Test Comparison of Student Interest in CS (Variable Group)

Figure 32 illustrates that student responses to this query were normally distributed. Two students reported on the pre-test that they had "no interest" in CS, and in the post-test, this dropped to one student. Overall, the majority of surveyed students (96%) were at least "slightly interested" in topic of CS, which is to be expected, since *Virtual Worlds* is an elective course, not required for any major or minor at the university.

To determine if there was an increase in student interest in CS as a result of the course, a Student *t*-test was conducted to determine if the change was a result of chance. Table 17 depicts the results of the *t*-test for the variable group.

	Pre-Test (Q20)	**Post-Test (Q34)**
Mean	2.41176	2.78431
Mean Difference	0.37254	
***p*-value (two-tailed) ≤ 0.05**	0.011889	

Table 17. *t*-Test for Variable Group Pre-Test (Q20) and Post-Test (Q34) where $p \leq 0.05$

These results indicate that there is a statistically significant difference between the variable group's pre-test (Q20) and post-test (Q34) measuring of student interest in CS as a result of the course.

Figure 33 illustrates the summary of pre-test and post-test measures for student interest in CS for the control group.

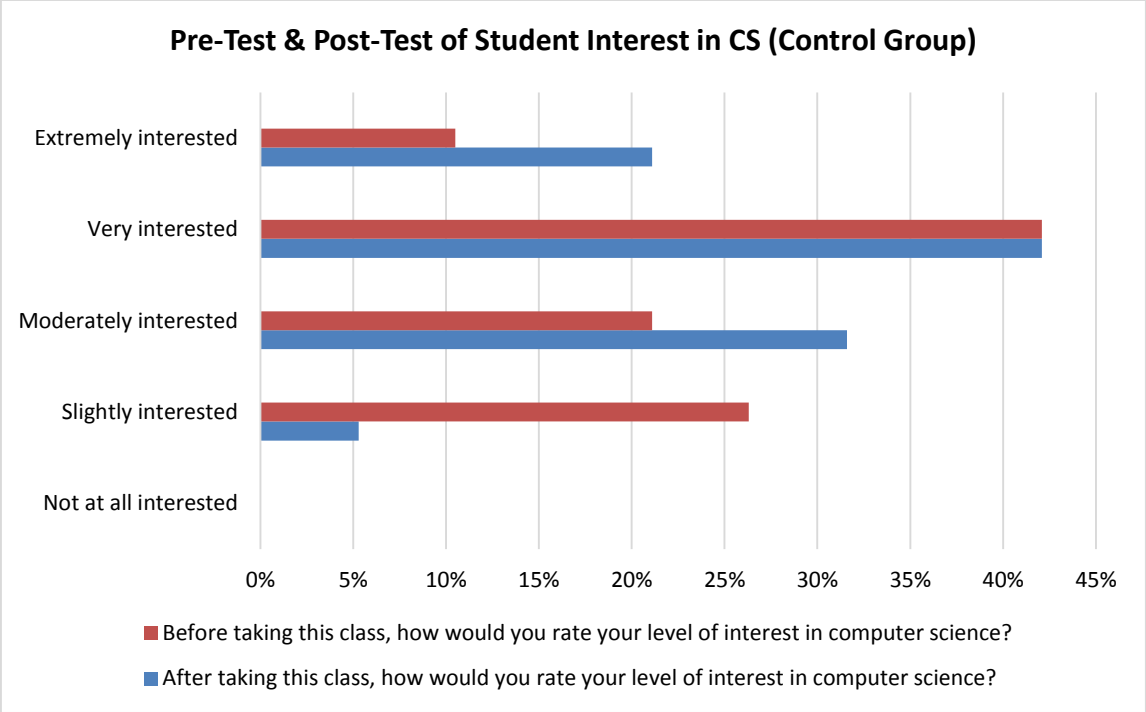


Figure 33. Pre-Test & Post-Test Comparison of Student Interest in CS (Control Group)

For the control group, no students reported in either the pre-test or post-test that they had "no interest" in CS. 26% of students in the pre-test indicated they were "slightly interested" in CS, compared to the post-test, where only 5% of students reported they were only slightly interested. Therefore, the control group also experienced an increase in student interest in CS as a result of the course. Another *t*-test was performed to determine if this increase in student interest was a result by chance, as shown below.

	Pre-Test (Q20)	Post-Test (Q34)
Mean	2.36842	2.78947
Mean Difference	0.42105	
***p*-value (two-tailed) < 0.05**	0.07229	

Table 18. *t*-Test for Control Group Pre-Test (Q20) and Post-Test (Q34) where $p \leq 0.05$

The value of *t* is 1.909337, and the value of *p* is 0.07229. The result is not significant at $p \leq 0.05$, which means the null hypothesis cannot be rejected. However, these results are significant at $p \leq 0.10$, so there is reasonable likelihood that these results were not a result of chance. One explanation of these results may be that students in the control group already had a higher level of interest in CS, particularly since more engineers were represented in that class than in the variable group. Nevertheless, it appears that students in the variable group experienced a greater increase in interest in CS as a result of the course, compared to students in the control group.

Qualitative data also supports these findings. Student comments below were coded to indicate an increase in "interest" about computing and how that potentially may impact the students' everyday lives [emphasis added]:

> *"**This course made me more interested in computer science** and now I understand what it really is and have enough knowledge to educate myself more, and start doing some **computer science in my everyday life**. **My way of thinking has changed**."*

> *"**Since taking this course my interest in computer science has greatly increased**, because of all the possibilities that computer science has to offer."*

> *"I think anyone with a moderate interest in computers or code should take this class. **Even if CS does not interest you there are many useful concepts in this course** that I think **many people would benefit from**."*

*"I actually found **the majority of the assignments really enjoyable and interesting**."*

*"**This course has made me much more interested in and motivated to continue with CS**. I feel like I already know so much more about technology in general, and how crucial it is to global development."*

*"Computer science education is very important today, and **this class approaches it in a very interesting and applicable way**."*

*"I think it's important to understand basic CS principles, and this class does that well while **being interesting and enjoyable at the same time**."*

*"I like this course because **it's truly interesting and fun**; it **maintains my interest well and has taught me a ton** (and given me career ideas), so bravo. "*

*"**I found the course very interesting** and it is a good class to be in if you want to learn about CS but have no prior knowledge."*

*"Generally speaking, **this class has strengthened my interest in comp sci**"*

*"This course **only supported my perception of computer science**. The reason is because **I was interested in the materials** for most of the class."*

Some students specifically noted their hesitation toward CS was a result of previous assumptions or perceptions about it; the course helped students overcome this reticence to look at CS in a different way [emphasis added]:

*"I was **always interested in CS, but always a bit afraid of taking a CS course** because I am not that good at math, **but now I would not hesitate to take more CS classes** even if they have more math."*

*"**I was interested, but also scared**. I thought it was going to involve a lot more numbers and math."*

*"Before this class **I was interested in CS , but didn't have a clear image of what it included. CS also has a lot of stigma around it.** I was curious but hesitant…. I've learned a lot, and it has sparked my interest. **I'm definitely not hesitant anymore**."*

*"Before this course, **I had always wanted to try something in computer science**, **but I was never good with math or science**. Bunches of numbers would easily overwhelmed and confuse me. I've always been interested in computers and gaming."*

> *"I really, really enjoyed [the] class. Before I took it**, I was very doubtful of my potential programming and computer science** abilities, and **the class cleared all of it away**."*

These comments reinforce previous work that identified how many students have negative perceptions of CS as a field of study, even if they are interested in technology, gaming, or other industries related to computing. Some students explicitly mentioned that they had an increase interest in CS because of the breadth of topics that were introduced to the course [emphasis added]:

> *"I thought it was **very interesting to learn about the effects of technology in multiple different aspects**. For example, the **medical benefits, business efficiency, and solving social problems**. It was also interesting to learn about who does not benefit from technology and the digital divide."*

> *"I enjoyed the **range of topics** most of all. It was **very beneficial to learn a great deal about computer science overall**, rather than just one small aspect."*

> *"This course is a wonderful introduction to computer science. **While the topics are broad, they are relevant and interesting**. The lectures were thorough and insightful, and I think **my understanding of computer science has increased tenfold** through the lectures alone."*

The breadth and scope of CS Principles content evidently helped increase student interest in the field. Regardless of the gamification, the course content appears to have positively impacted student perceptions of CS as a field of study.

4. Finding 3: CS Principles Achieved Intended Learning Outcomes

CS Principles is designed to introduce students to the central ideas of computing and computer science, to instill ideas and practices of computational thinking, and to have students engage in activities that show how computing and computer science change the world. CS Principles thus seeks to facilitate learning outcomes around three key themes: (1) that computer science is a creative practice, (2) understanding the use of technology as a means for solving computational problems and exploring creative endeavors, and (3) focus course content on people

and society, not just machines and systems, in order to help the course appeal to a broad audience (Astrachan & Briggs, 2011). As described below, the *Virtual Worlds* course was successful in achieving all of the intended learning outcomes intended of CS Principles.

Results indicate that CS Principles facilitated positive programming experiences for students, helped increase student interest in CS as a field of study, helped students consider CS as a creative practice, and also helped encourage students to continue learning CS after the course had finished. Proxy pre-tests and post-tests were used to evaluate whether CS Principles had positive impact on how much was learned in the course (FCQs; Q37; Q38), student programming skills (Q19, Q33), and whether students consider CS as a creative practice (Q13, Q35). In addition, results from a one year follow-up survey were analyzed to determine the extent to which students pursued the study of CS after the course had finished.

FCQs were used to determine how much students felt they learned in the course. The figure below summarizes students' self-reported measures about how much was learned in the course.

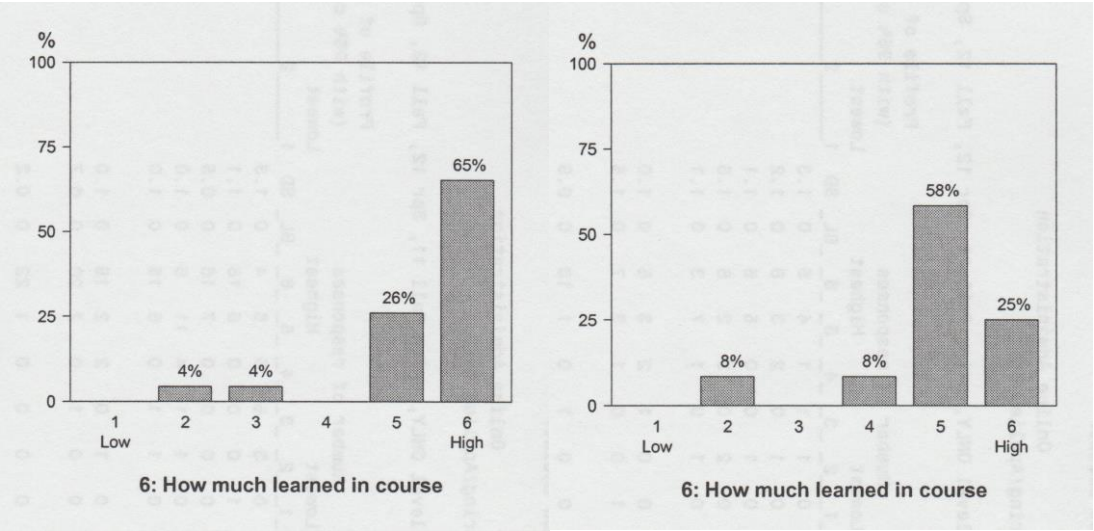


Figure 34. FCQ "How Much [was] Learned" between Variable (left) & Control (right) Groups

When determining "how much [was] learned" in the course, the variable group has a more favorable growth-index for the amount learned. The FCQ results indicate that students in the variable group believed that they "learned more" than students in the control. Although there were more freshman in the control, this group had a larger percentage of declared-engineering majors and students who indicated previous experience with programming. The variable group indicated a stronger correlation for more positive learning outcomes.

Survey instruments also inquired whether students felt that they had acquired a better understanding of how computing systems work (Q37), and if the course broadened their general perspective of what CS involves as a field of practice and study (Q39). Figure 35 summarizes these results.

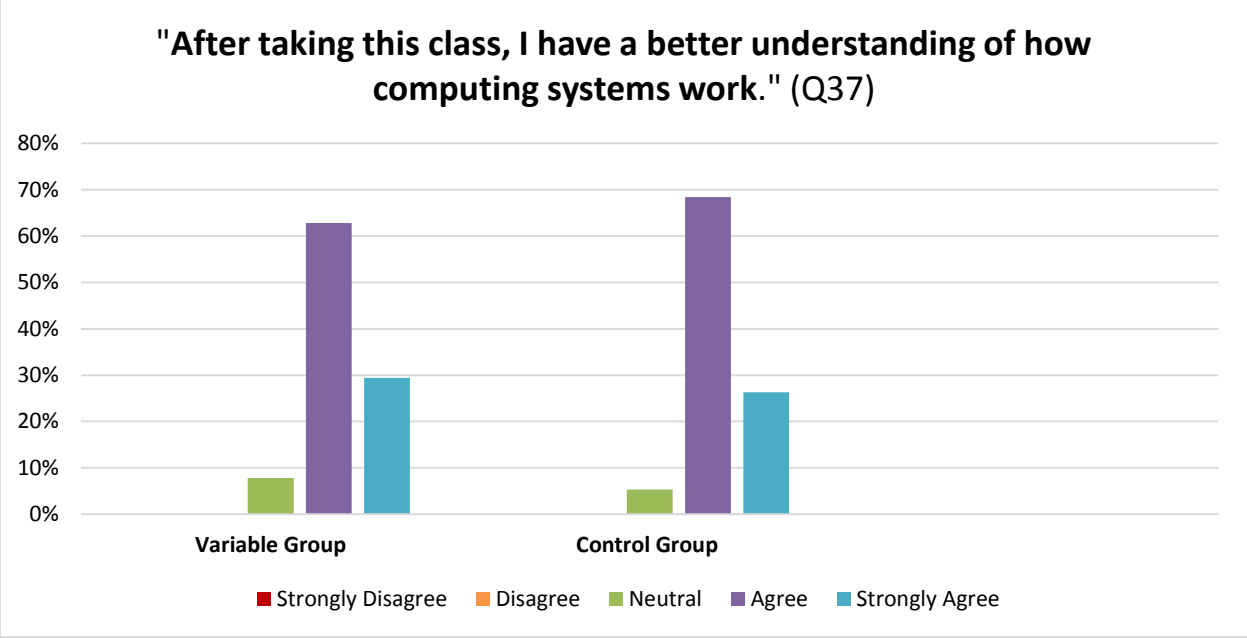


Figure 35. Summary of Q37 for Variable Group (Left) and Control Group (Right)

The median and mode for Q37 in both the control and variable group was a 4 or "agree." This result suggests that students in both the control and variable groups improved their understanding

of how computing systems work as a result of the course content, particularly because there were no negative items ("Strongly Disagree" or "Disagree") reported.

Figure 36 below summarizes responses of students in the control and variable group regarding whether they agreed or disagreed that the course broadened their general perspective of what CS involves as a field of study and practice (Q39).

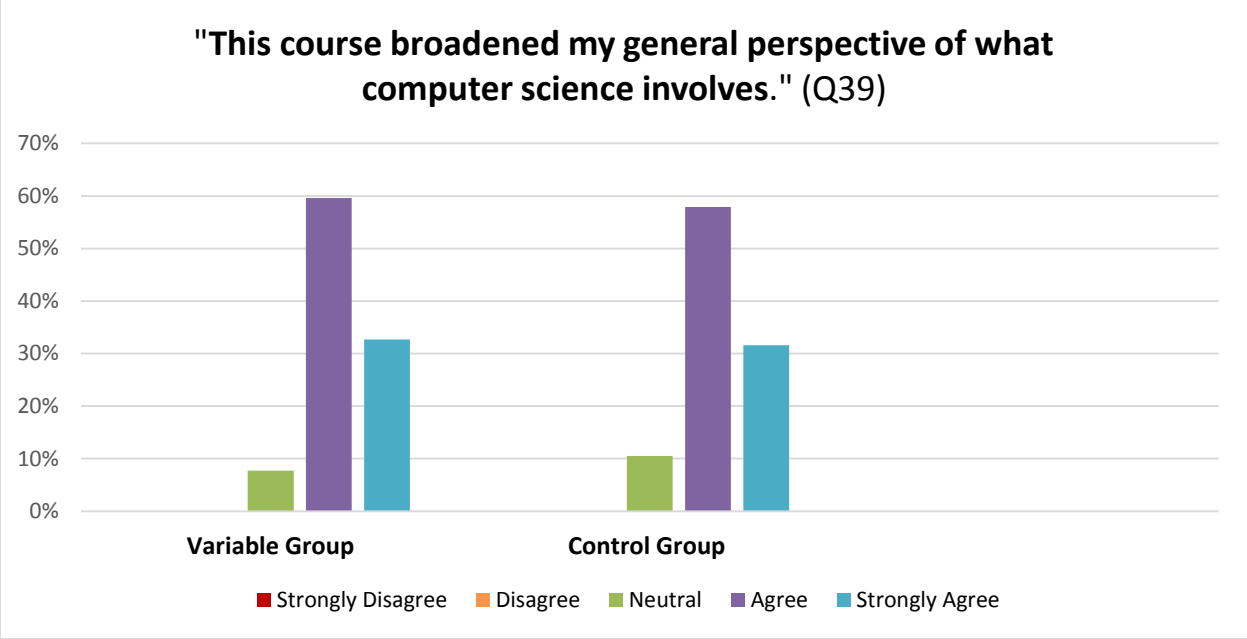


Figure 36. Summary of Q39 for Variable Group (Left) and Control Group (Right)

The median and mode for Q39 in both the control and variable group is a 4 or "agree" on the Likert scale. Again, there were no negative items ("Strongly Disagree" or "Disagree") reported. These data suggests that students in both control and variable groups broadened their perspectives of CS because of the course content.

Qualitative findings also supports this result. The comments below were coded to indicate an overall "positive change" in student perception and understanding about what CS involves and how that potentially may impact the students' everyday lives [emphasis added]:

> *"This course **has completely changed my mind about CS**. A large part of this course has used **creative thinking and design**. As a creative thinker, **I have found CS to be accessible and fun!**"*
>
> *"This course has **changed my perception of CS.** I found that, with time and patience, **learning CS is no more difficult than other fields** and that, for me, it may be more intuitive."*
>
> *"I like the knowledge of the course and **the new outlook it gave me on CS knowledge**."*
>
> *"This course has **changed my perception of computer science** and made me feel like **I actually could do this stuff**."*

One student even distinguished the conceptual difference between programming, computer science as a field, and computer literacy skills [emphasis added]:

> *"I **really liked the programming** quests, I learned a lot in general about **computer science** and **how to better use computers** (even stuff like compressing zip files was new to me.)"*

Both quantitative and qualitative data indicate that students learned more about CS as a field of practice, which broadened their general perspective about the nature of computer science.

4.1. No Grade Difference Between Genders

Since the CS Principles curriculum seeks to appeal to a broad audience, I used a Spearman's Rho test to measure if there was any correlation between student gender and student grades in the course. The association between grade and gender was found to not be statistically significant; therefore, students in the variable group did not fare better or worse in the course strictly because of gender. This result is received positively, because the CS Principles is intended to appeal to both men and women.

4.2. CS Principles Higher Course Rating than Equivalent Courses

Faculty Course Questionnaire (FCQ) were also used as a measure for overall attitude about the course. The figure below shows the FCQ course ratings for both control and variable groups.

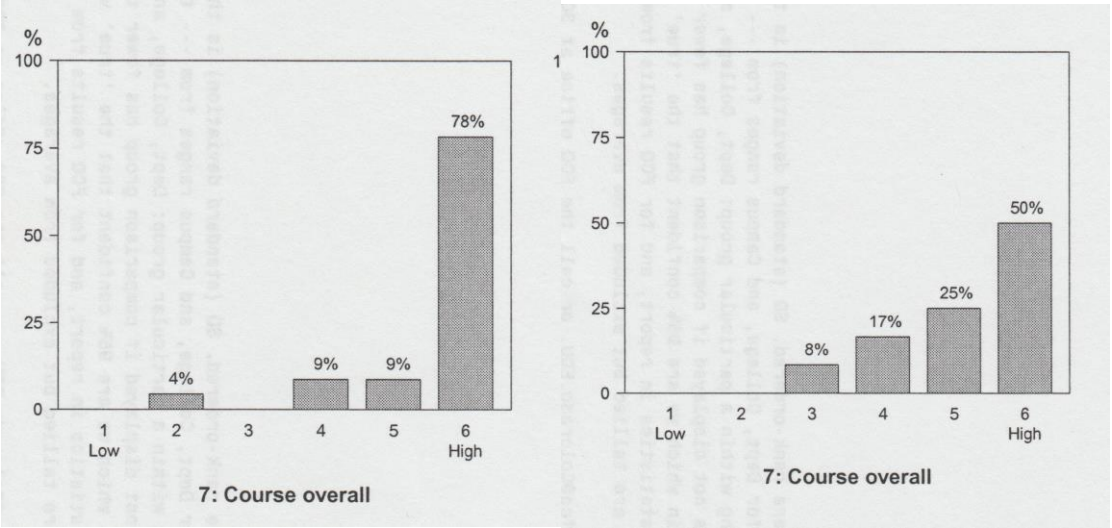


Figure 37: Course Rating via FCQs for Variable (left) & Control (right) Groups

At least half of students enrolled in the course (for both sections) indicated a six or "High" rating for the overall course. For the variable group, the average rating for "course overall" was 5.6. For the control group, the average rating for the "course overall" was a 5.2

When comparing to older iterations of the course, the average "course overall" rating for ATLS/CSCI-1220 between Fall 2008 to Fall 2013 was a 4.8. Figure 38 shows the course ratings compared to the mean of course ratings in the Department of Computer Science and the College of Engineering.

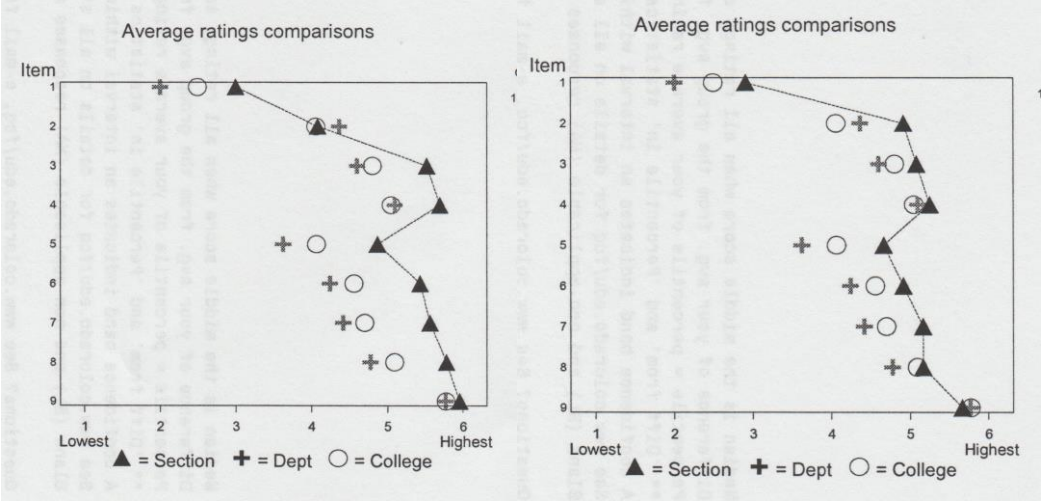


Figure 38. Average Ratings between Variable (left) & Control (right) Groups

When compared to the 215 course-sections of CSCI-1300, the introductory programming course for CS majors in the College of Engineering, the average "course overall" for these sections from Fall 2008 to Spring 2014 was 4.7. This iteration of CSCI/ATLS-1220 was apparently at least as well received as another instantiation of introductory computer science courses at CU Boulder over the past five years. Although there is a slight increase for the overall rating for the variable course, both sections of are well above the average rating for both the CS Department and the College of Engineering. This illustrates a potentially positive correlation for CS Principles as a course framework.

4.2.1. CS Principles Facilitated Positive Programming Experiences

Overall, students in both sections indicated a "low confidence" score in their programming skills at the beginning of the semester. This was measured by students rating their overall programming experience on a 1-5 Likert scale[37]. Of the surveyed students in the variable group (n=51), 88% indicated they had "no experience" or "little experience" in programming (σ=0.8);

[37] 1=No Experience, 2=Little Experience, 3=Some Experience, 4=Moderate Experience, 5=Advanced Experience

only 12% of students indicated "some experience," and only one student in each course indicated a moderate or high confidence in their programming skills.

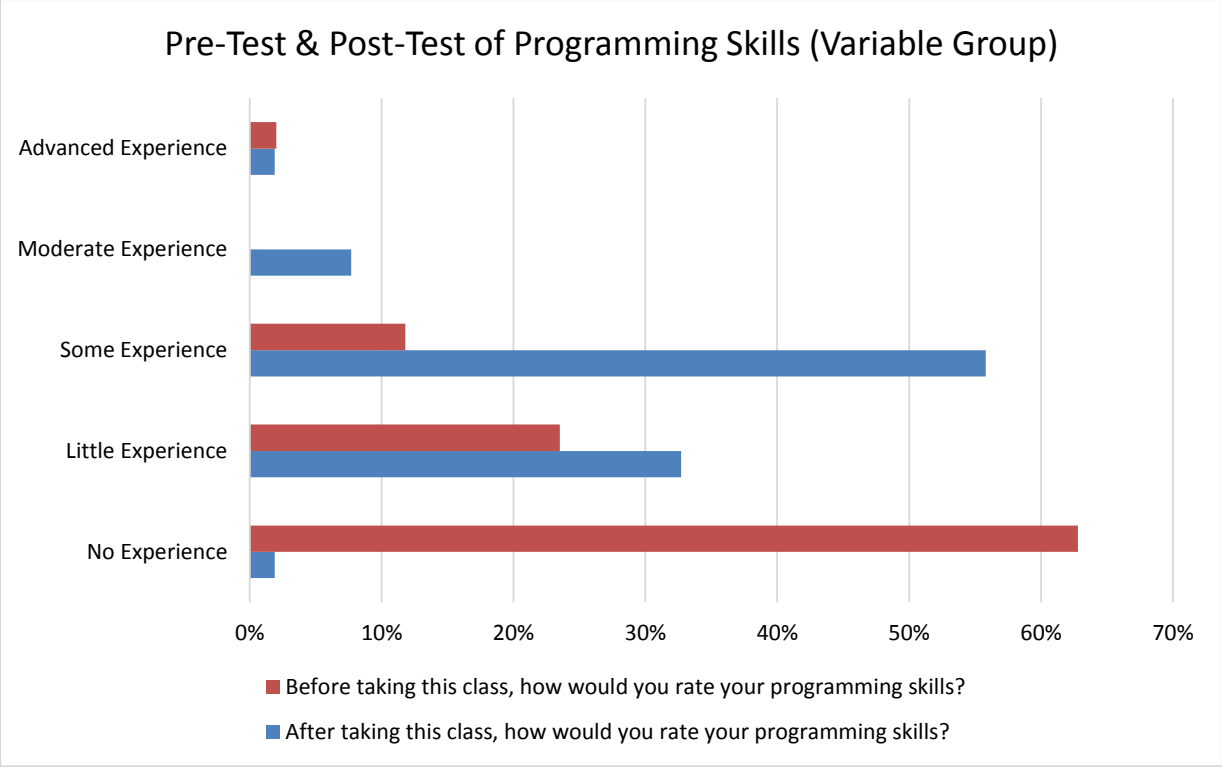


Figure 39. Pre-Test (Q19) & Post-Test (Q33) of Programming Skills (Variable Group)

However, after taking the course, a majority of students moved from "no experience" to "some experience." A *t*-test was performed to determine if there was a significant difference on programming skills from the pre-test to the post-test results. Table 19 illustrates these results.

	Pre-Test (Q19)	Post-Test (Q33)
Mean	0.47059	0.74510
Mean Difference	0.27451	
***p*-value (two-tailed) ≤ 0.05**	0.014815	

Table 19. *t*-Test for Variable Group Pre-Test (Q19) and Post-Test (Q33) where $p \leq 0.05$

The value of *t* is 2.5243. The value of *p* is 0.015, which is significant at ≤ 0.05. Therefore, students in the variable group had a meaningful increase in their level of confidence for programming.

In the control group (n=19), 84% of surveyed students indicated in the pre-test that they had "no experience" or "little experience" in programming (σ=0.9). In addition, 11% of students indicated "some experience" and only one student reported "moderate programming experience" (this student was a declared major in Computer Science). Figure 40 summarizes these results.

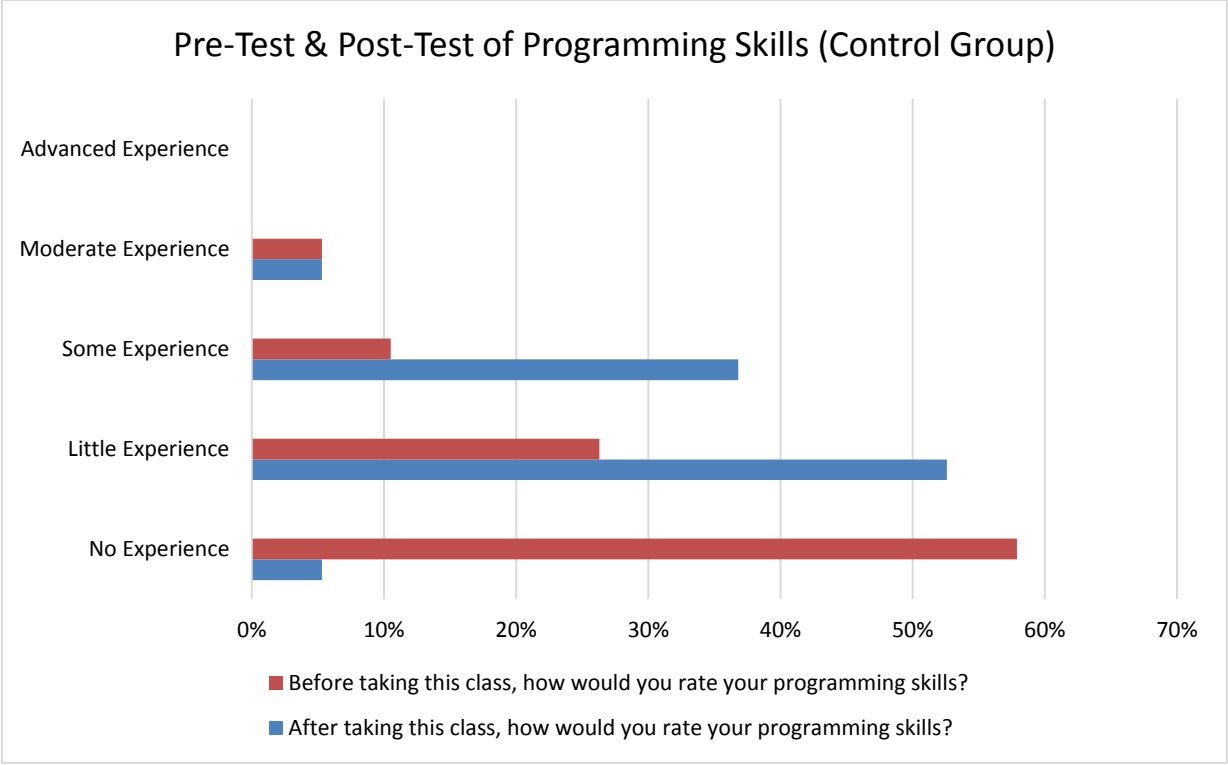


Figure 40. Pre-Test (Q19) & Post-Test (Q33) of Programming Skills (Control Group)

Similar to the variable group, students in the control were found to have a higher confidence score for their programming skills after the course. Another *t*-test was used to validate that students increased their level of confidence in programming as a result of the course. Results of this test are shown in Table 20.

	Pre-Test (Q19)	**Post-Test (Q33)**
Mean	0.63158	0.78947
Mean Difference	0.78947	
***p*-value (two-tailed) ≤ 0.05**	3.5E-05	

Table 20. *t*-Test for Control Group Pre-Test (Q19) and Post-Test (Q33) where $p \leq 0.05$

The value of *t* is 2.5243. The value of *p* is 3.5E-05, which is significant at ≤ 0.05. Therefore, students in the control group also had a meaningful increase in their level of confidence for programming.

Qualitative data supported the claim that students had an increase in confidence for programming. Overall, the reaction to the nature of the programming assignments was positive. CS Principles appears to be a good fit for those interested in computing with little to no prior experience, while encouraging the diverse and creative components of the field, which include programming. Examples of student comments related to this factor are shown below [emphasis added]:

> *"As an **introductory course**, the **level of programming and topic choices were right on point**. I just wished I had gone into CSCI 1310 (which is kind of what I thought I had enrolled into, even so, **I'm glad I took this course**."*
>
> *"I had never taken a class that [includes] **how to code**, and it was **very helpful and interesting to see how it all works**."*
>
> *"So many CS courses either assume knowledge about CS and computers as a whole that **they gloss over what makes CS a cool area of study**. **I feel more like a computer scientist than a programmer."***
>
> *"I wanted to take a course that gave me the **basics of computer programming** and that would teach me a couple of **things that I didn't know** before I showed up. **This course did both**."*
>
> *"I found the course **very interesting** and it is **a good class** to be in **if you want to learn about CS but have no prior knowledge**."*
>
> *"I think **anyone with a moderate interest in computers or code should take this class**. Even if CS does not interest you **there are many useful concepts in this course** that I think **many people would benefit from**. Knowing how to control your computer is **important**!"*
>
> *"I thought CS was just crazy smart people doing super hard things that I would never understand. I still think it's super smart people doing crazy hard things **but now I know there is also simple CS problems and activities that I understand**."*

Several students made the distinction of "level-of-appropriateness" about programming for this class, depending upon whether the student was going to major in CS [emphasis added]:

> *"For someone **looking to be a CS major**, I don't think I would [recommend this class]. I would tell them to go into a {intrans} class **covering a more in-depth analysis of computers and programming** specifically. **For anyone else: absolutely**!"*

> *"I would definitely **recommend this course** to students who **have an interest in taking computer science but don't have any kind of background** in the field. It's a great **introduction** to the concepts and ideas that CS majors need to be familiar with. **I'm not sure I would recommend this class to people already in computer science**, though, because while I'm sure they'll learn new things, I think come of it may be a bit **redundant**."*

In contrast, some students indicated that the course would be beneficial even for CS majors [emphasis added]:

> *"This class is very interesting and also **important** for TAM minors **or CompSci majors to try to better understand their majors**."*

Some students self-reported that they would have preferred more focus on programming [emphasis added]:

> *"I **dislike the lack of programming**; but this is an introductory course so 'se la vi'"*

> *"I like the self guided structure of the homework, though **I would be more interested to see more programming taught**."*

Even though programming is not the sole focus of CS Principles (as is generally the case with AP CS A or other introductory CS courses), students perceived an increase in their level of confidence in their programming ability. Although these students do not identify as being experts, they demonstrated a better understanding and appreciation for programming. This exposure to programming seemed appropriate in an introductory CS course for students with minimal prior experience.

4.2.2. CS Principles Helps Students Consider CS as a Creative Practice

One of the key themes of CS Principles is to show students that CS is a creative practice. Students in CS Principles are expected to demonstrate that creativity through practice. Survey questions sought to measure student's preconceptions about whether they considered CS as a creative practice (Q13) and whether their exposure to the course changed this perception in some way (Q35). The figure below illustrates the pre-test assumptions of CS as a creative practice and the post-test results for the variable group.

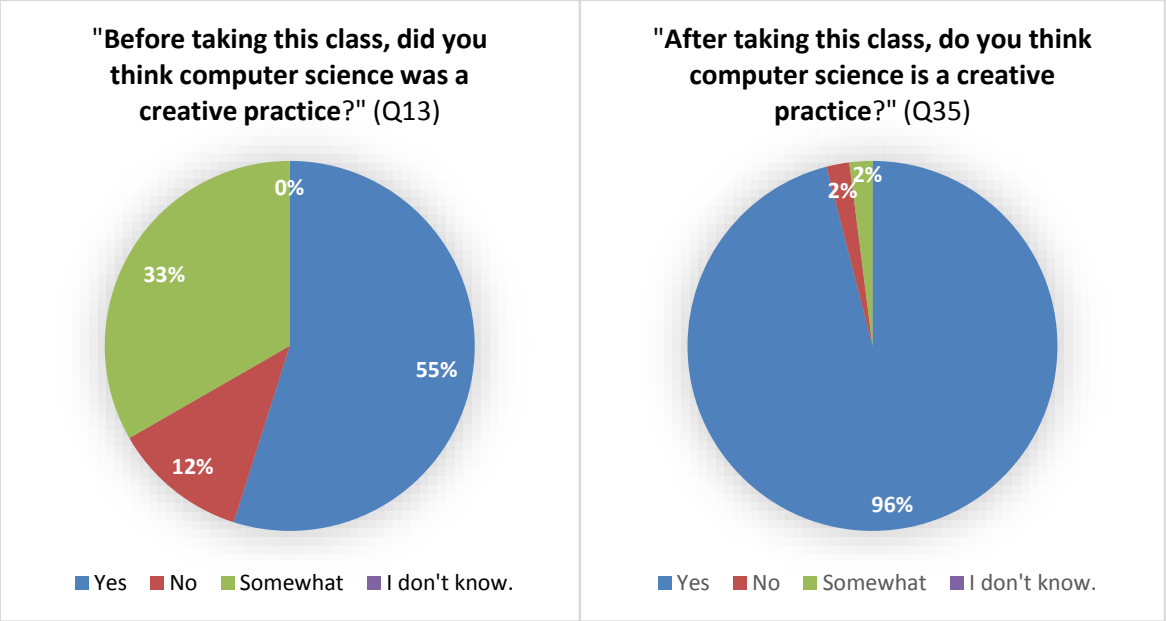


Figure 41. Pre-Test (Q13) & Post-Test (Q35) for CS as a Creative Practice (Variable Group)

The figure shows that, for the pre-test, a majority of students thought that CS was a creative practice (55%), where 12% thought CS was not a creative practice, and 33% were unsure. After taking the class, almost no students were unsure about whether CS was a creative practice or not. The post-test also shows a significant increase of students' positive perception, where 96% considered CS to be a creative practice. These data suggest that the content of the course helped shift student perceptions about CS as a creative field of practice for the variable group. Another

t-test was used to determine if there was a statistically significant difference between the pre-test and post-test measures. Table 21 summarizes the results of this test for the variable group.

	Pre-Test (Q13)	**Post-Test (Q35)**
Mean	0.705882	0.970588
Mean Difference	0.264705	
***p*-value (two-tailed) ≤ 0.05**	2.62E-05	

Table 21. *t*-Test for Variable Group Pre-Test (Q13) and Post-Test (Q35) where $p \leq 0.05$

The *p*-value is 2.52E-05, which is significant at ≤ 0.05, indicating a significant increase in students' perception of CS as a creative practice.

Students in the control group reported similar attitudes about CS as a creative field of practice, as shown Figure 42.

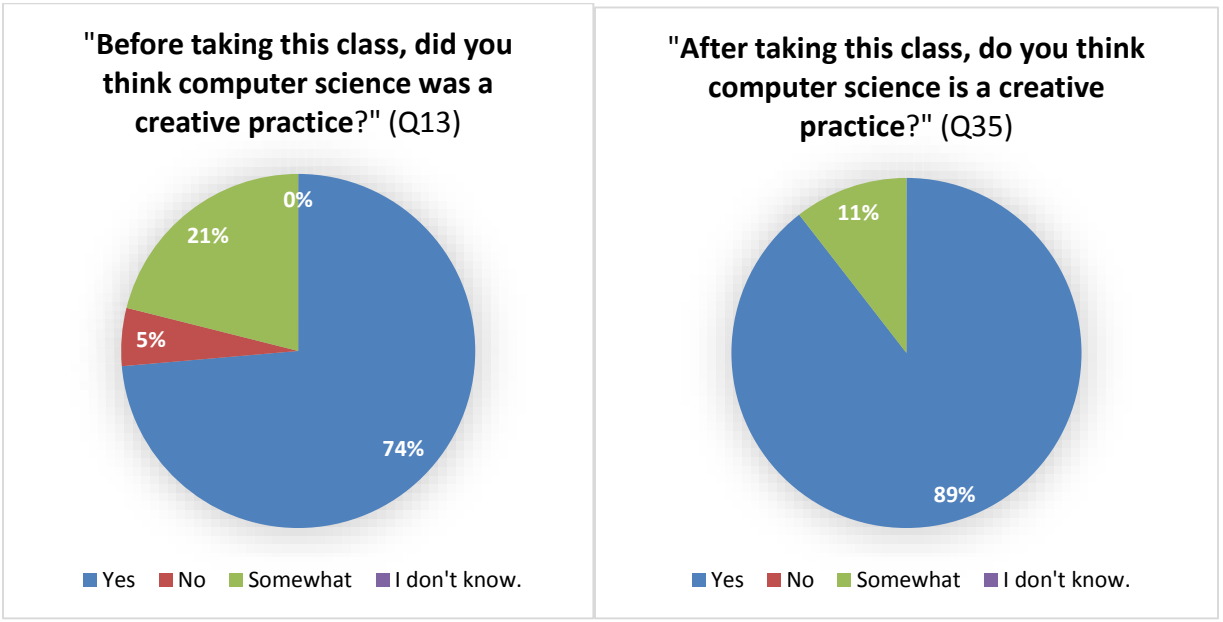


Figure 42. Pre-Test (Q13) & Post-Test (Q35) for CS as a Creative Practice (Control Group)

Similar to the variable group, a majority of students in the pre-test (74%) indicated that they thought CS was a creative practice, where 21% thought "somewhat" and 5% indicated it was not a creative practice. Again, no students reported that they were unsure if CS was a creative practice or not, further indicating students have some sort of assumption or perception about the field before taking the course. The post-test results also show that there was an increase in the amount of students that considered CS as creative practice, although not to the same degree as students in the variable group (89% "Yes" and 11% "No," compared to the variable group's 96% "Yes" and 2% "No," 2% "Somewhat"). Another *t*-test was used to determine this degree of difference.

	Pre-Test (Q13)	**Post-Test (Q35)**
Mean	0.842105	0.947368
Mean Difference	0.105263	
***p*-value (two-tailed) ≤ 0.05**	0.10364	

Table 22. *t*-Test for Control Group Pre-Test (Q13) and Post-Test (Q35)

Overall, students in both the control and variable group show an increase perception that CS is a creative practice, but for the control group, the increase was not as statistically significant.

Qualitative data reflect the quantitative findings. The following examples of student comments student attitudes about CS and creativity [emphasis added]:

> *"I realized that **my creative approach is actually beneficial to the sciences**, and it's really enriching to find new ways to merge the two (…I particularly loved the discussions and project on abstraction). Looking back on my college career so far, it was **a class that truly left a lasting impression on me**."*

> *"I now see **computer science as a challenging but creative pursuit**."*

*"I enjoyed learning about **new inventions created through the use of programming and coding**."*

*"The **programming** aspect was **very cool to create something of my own**."*

*"**CS is applicable to almost every subject or major!** I think everyone would benefit from learning this stuff. They can take what they want from it and **apply it to their majors/lives/careers in their own creative ways**."*

*"A large part of **this course has used creative thinking and design**. **As a creative thinker**, I have found **CS to be accessible and fun**!"*

*"I really liked the **creative outlets** that **computer science has connection with**."*

These comments highlight the benefit of CS as creative practice and an understanding of how CS can have meaningful connection and application to other fields of practice, whether in their "majors/lives/careers" or generally in the sciences. In addition, the students reported that programming assignments helped them demonstrate their creativity by "creat[ing] something of [their] own." These findings support the quantitative result that a majority of students come to think of CS as a creative practice, confirming this CS Principles learning outcome.

4.2.3. CS Principles Encourages Students to Continue Learning CS

Another goal of the CS Principles framework is to encourage students to pursue CS as a field of study in an effort to increase participation and diversity in the field. Several measures were used to assess whether students would pursue (or did pursue) computing after the course ended. First, survey items Q62, Q63, and Q64 attempted to measure if (1) students would change their major to CS as a result of their experiences in the course; (2) if their experiences in the course led them to consider taking additional CS courses at the university, and (3) if students planned to learn more about programming on their own by taking online courses, participating in hack-a-thons, etc. In the variable group, 28% of surveyed students stated that they were already majoring in CS and 42% of surveyed students in the control group were already majoring in CS. Some

students did extend their study and continued to pursue CS as a result of this course. Data from the Q62 item identifies that six students (five men, one woman) from the variable group (25% of freshmen and sophomores; 12% of the total survey sample) self-reported that they will change their major to computer science after taking this class. Several other students in the variable group also reported that they would have liked to change their major to computer science but were unable to because it is "too late" or they indicated that they were "about to graduate," and changing majors at that point would be not be feasible. Two students in the control group stated that they would change their major to CS as a result of the course, others noted similar reasons why they would not—some already had a declared major, CS "still seems very difficult," some are close to graduation, etc. Some students stated that they simply didn't want to pursue CS as a major.

Figure 43 summarizes the responses for Q63, which sought to measure if students would pursue other CS courses at the university (but not necessarily change their major to CS).

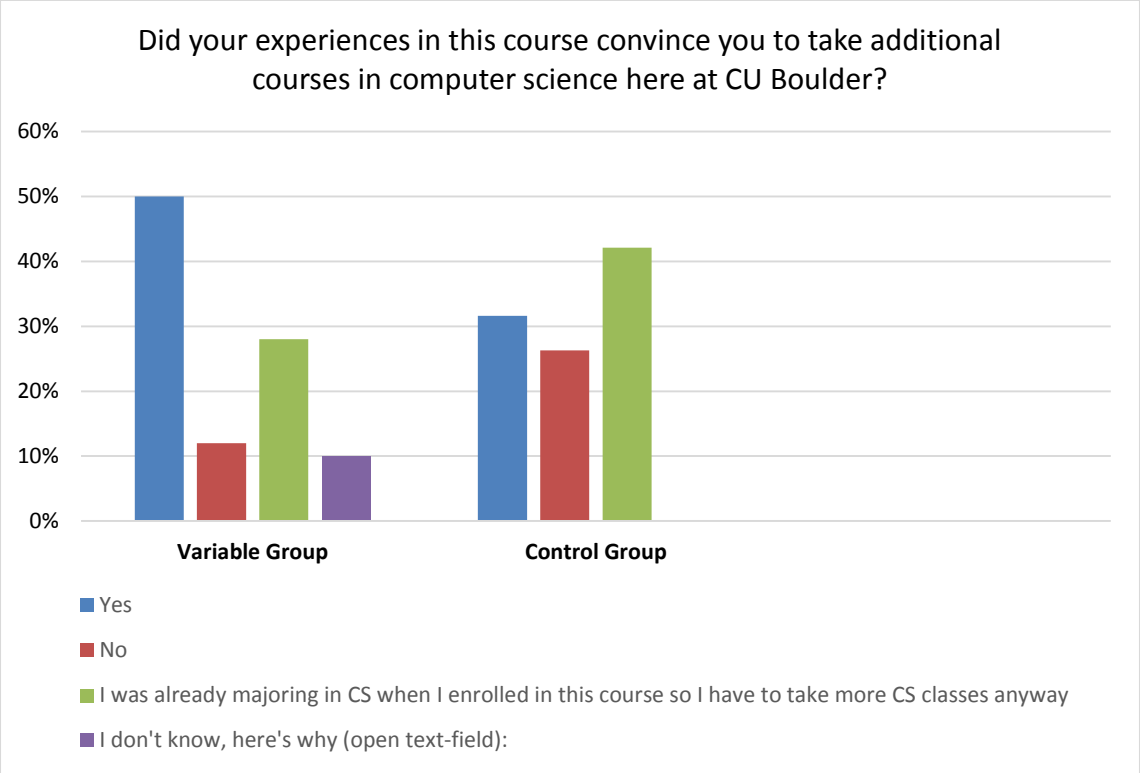


Figure 43. Total Item Responses for Q63 for Variable Group (left) & Control Group (right)

In the variable group, 50% of surveyed participants (25 students) stated that they would take additional courses in CS at the university; 32% of surveyed participants in the control group (6 students) stated they would take additional CS courses. The control group had a higher proportion of engineers represented in the sample. Examples of student comments from the variable group who responded with "I don't know, here's why" include [emphasis added]:

> *"**Graduating** in May. Have **courses picked** out already."*
>
> *"I **dropped my CS major** in the second week."*
>
> *"**I want to**, but I cannot due to **major requirements**."*
>
> *"**If I weren't graduating** in December, **yes**!"*
>
> *"**I probably would have** but I only have **one semester of college left** so I chose classes similar to my major instead."*

Similar to the previous student responses, these students indicated that they chose not to pursue CS because they were anticipating graduation, had already decided to drop the CS major, or had to choose specific courses more related to their major, instead of CS courses. These results suggest students may wish to pursue other CS educational opportunities but are hindered from doing so because of external factors and obligations. Interestingly, no students in the control group responded to the Q63 open-ended question. Students in the control group also had a higher representation of students who explicitly said "No" to taking more CS courses in the future (26%, 5 students total) compared to the variable group (12%, 6 students total).

I also sought to determine whether students would consider additional CS instruction outside of the scope of a traditional classroom or undergraduate course. Item Q64 was intended to identify an inclination to pursue CS instruction in Khan Academy tutorials, hack-a-thons, or other non-traditional venues. Figure 44 shows the results of this inquiry.

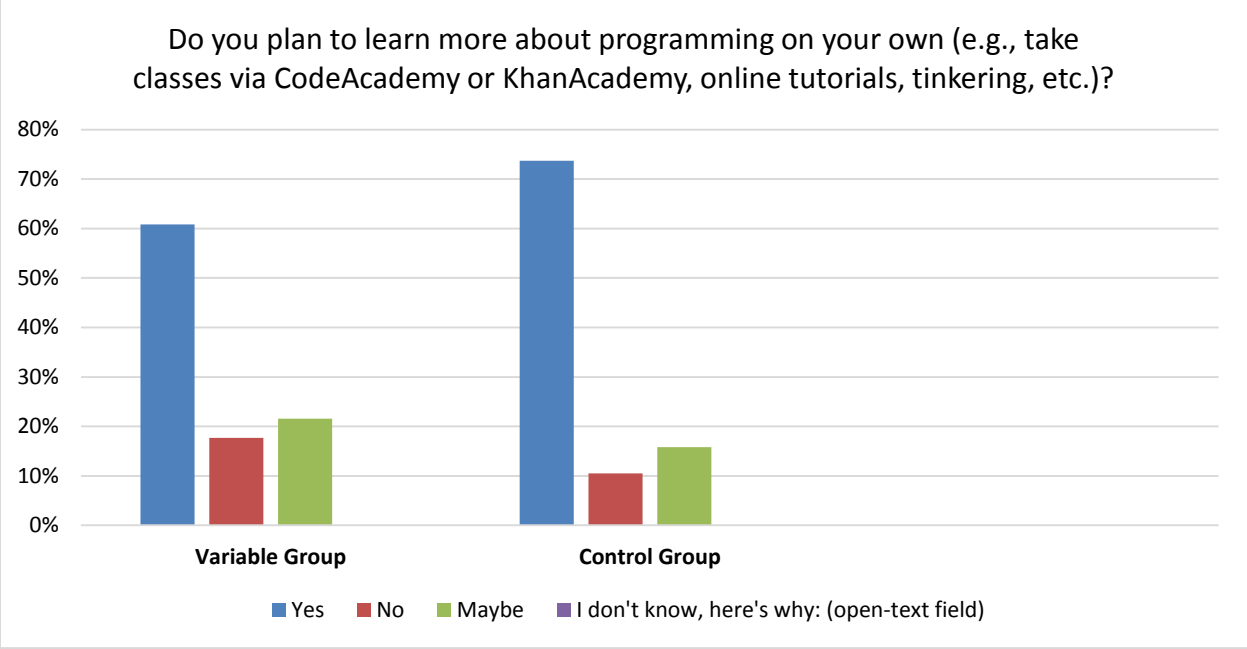


Figure 44. Total Item Responses for Q64 for Variable Group (left) & Control Group (right)

A total of 31 students (61%) in the variable group and 14 students (73%) in the control group said that they would learn more about programming on their own, in online classes, by tinkering, or by other means. Only two students in the control group (11%) and nine students (18%) in the variable group said that they did not intend to learn more about CS on their own. Students who remained unsure (by indicating "maybe") outnumbered the students who explicitly said "no" to this survey item. These results suggest that students may be more likely to pursue follow-on computing instruction in unconventional ways. CS educators should perhaps place more emphasis on introducing these opportunities to facilitate this activity.

A follow-up survey was administered to Virtual Worlds students one year after the course had finished. The full text of this survey is contained in Appendix D.

A total of 25 students participated in the follow-on survey (27% of the total students whom enrolled in ATLS/CSCI-1220 [both sections] during the Fall 2014 semester). Of the survey takers, 12 were female and 13 were male. Students from both course sections were represented in the survey. Seven students (28%) had been enrolled in the control group (section 002), and eighteen students (72%) had been in the variable group (section 001). Of the students in the control group, four had enrolled in ATLS-002 and three in CSCI-002. Of the students in the variable group, four students had enrolled in ATLS-001 and thirteen in CSCI-001.

The first three items (Q1, Q2, Q3) on the survey sought to measure if (1) students had taken any additional CS courses at the university in the last year, (2) if students had participated in any free online courses or programming tutorials such as Khan Academy, Coursera, Code.org, etc., and (3) if students learned more about CS or programming through hands-on experiences, such as participating in a hack-a-thon, summer internship, workshop, etc. Figure 45 summarizes these results.

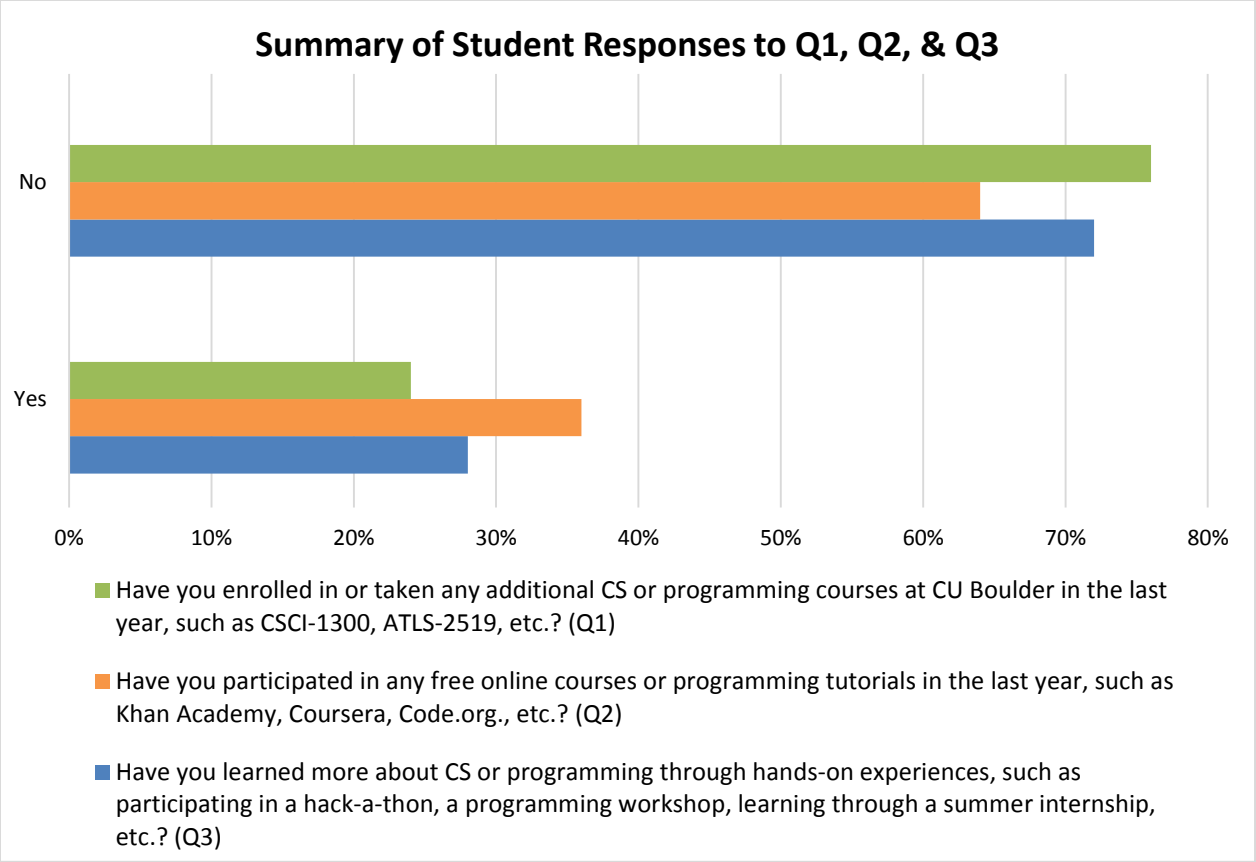


Figure 45. Total Item Responses for Q1-Q3 for One Year Follow-Up Survey

Students had three choices to select for these questions, "yes, "no," and "I don't know." Since no participants selected the third option "I don't know," this option is not shown in Figure 45.

For Q1, six students (24%) reported that they had taking additional CS or programming courses at the university in the past year. For those students that said "yes" in the survey, they were prompted to answer another question in the survey[38] and describe which course they took and their overall experiences of that course. Five students reported that they had taken CSCI-1300 (the required, introductory course for the CS major) as their follow-up course, whereas one student

[38] "Please describe the Computer Science or programming course you enrolled into, when you took the course, and pelase describe your overall experience as a student in this course."

reported they were in the TAM certificate program and only have one class remaining in the program. The following comments represent student responses to this prompt [emphasis added]:

> *"CSCI 1300 Spring 2015. The **course was more understandable after taking 1220**."*
>
> *"CSCI-1300, currently enrolled, been good so far. I have learned a lot of python and **been inspired to look into additional python online courses.**"*
>
> *"**I switched into a cs major because of this class.** I really enjoyed it so took Csci 1300 and it was fairly easy because [of 1220]"*
>
> *"In the spring of 2015 I took CSCI 1300. I enjoyed this course and **found it to be the perfect pace** for an introduction to computer programming."*
>
> *"CSCI 1310 – **Semester after 1220, I was very enthusiastic to be programming** in C++. The course was not at all challenging (I don't think anything introductory should be), but **it was still interesting and entertaining**. CSCI 2270 – This semester. Same as 13010, but with amazing pointer arithmetic!"*

Without being asked, one student reported that they switched into a CS major because of the *Virtual Worlds* course. Overall, these students appear enthusiastic and motivated to continue their study of CS. These comments also suggest that *Virtual Worlds* helped provide a foundation for more rigorous programming classes.

Surprisingly, more students reported that they had taken online courses or website tutorials (Q2) than formal courses at the university. Nine students (36%) reported that they had taken some kind of online learning to continue their pursuit of CS. Similar to Q1, when students indicated "yes" on the survey, they were prompted with a follow-up question to describe their participation and experiences using these tools[39]. The following comments are the direct responses from these students [emphasis added]:

> *"**Code academy**. I finished the first main section on html. **It helped a lot** with digital media"*

[39] "Please describe the online course or website tutorials you used, approximately how long you participated, and please describe your overall experience of these online activities."

*"I am a full time employee at Apple now and **I spend many hours a week on Code Academy**."*

*"I looked into the **Code**[.org] website sometimes. I also used the **Open Street Maps**."*

*"I started learning some html and python through **code codeacademy** and **khanacademy**"*

*"**Code Academy** to learn HTML/CSS I did it for 10 hours. It's a great website, **really taught me how to build a website**."*

*"I have completed the Python course in **khanacademy** and thought **it was really helpful with Csci 1300**"*

*"**Khan Academy, Code Academy** for a few months off and on. I very much enjoyed it and it's **nice to go at your own pace that's supplemental to class.**"*

*"**Khan Acedemy, random blogs and tutorials** for me to learn a little more about how to make my computer better/build my own pc"*

*"**Rosalind**. It's a **Bioinformatics Python** course. I just started it last week but have progressed reasonably far. I really like it, since I am a Biochemistry major and could theoretically **use this application of computer programming in my future career**."*

These students reported the use of the websites prompted in the question (e.g., Code.org, Code Academy, Khan Academy) as well as un-prompted resources, including Rosalind, "random blogs and tutorials," and even open-source applications like "Open Street Maps." These responses reflect that students had varying purposes for using the online resources, whether to learn a specific skill or to reach a specific objective, such as learning how to build a website, learning code to enhance their job skills or future career, or to help supplement their studies for other courses (e.g., Biochemistry or CSCI-1300). One student mentioned how this extracurricular activity helped their "tinkering" practices, such as learning more about hardware so they could build a PC. In summary, more students sought to learn and benefit from programming "at their own pace" than those who chose to take CS courses at the university.

Seven students (28%) reported that they continued their pursuit of CS by participating in hands-on learning activities during a workshop, summer internship, etc. As with the previous two

questions, students were prompted with an open-essay box to describe these experiences[40], as summarized below [emphasis added]:

> *"**Participated in Apple's 'Week of Code.'** I found the information very relevant and helpful."*
>
> *"**Learning more about my computer** and all the cool aspects of it. I use **Bitcoin** more and **can locate virus's on my computer a lot easier now**."*
>
> *"Where I work, the system we use is fuzzy and constantly giving us coding errors. **I read through the error using my knowledge gained from the course** as well as from using other programs to try to **dissect and locate problems in the system**."*
>
> *"**I have begun to program the classroom systems around campus**. When we upgrade a room from analog to digital we have to program the panel in order for the system to work. I have participated or **been a part of the programming step** at least three hours a week."*
>
> *"Worked on coding a website during a **summer internship**, and personally working on **coding my own tumblrfolio**."*
>
> *"I participated in some coding stuff and it was about an hour. It was a great experience and **I was so glad I had a basis of coding before participating**."*
>
> *"I work in a biomedical company and **most of my job is programming in VBA** and I have had to **build a website using HTML/CSS**."*

These comments were coded to indicate that *computational thinking* skills gained from the course had served the students in areas of their lives outside of class. For example, one student said he "read the through the error using my knowledge gained from the course," and abstracted the problem into workable solutions by "dissect[ing] and locat[ing] problems in the system." One participant reported using computational thinking to "[learn] more about [his] computer and all the cool aspects about it," and because of this effort, he "can locate [viruses] on [the] computer a lot easier now."

[40] "Please describe the hands-on learning experience you engaged in, approximately how long you participated, and please describe your overall experience while participating in these activities"

Other students mentioned the benefit of having a background in CS noting that it aided them in other work or career related endeavors, such as working in Visual Basic, building a website for a summer internship, participating in Apple's "Week of Code," or even programming a classroom to create a "smart" classroom. These responses are all positive indications that students used the fundamentals of CS, including computational thinking and abstraction, to solve real-world problems in their lives.

The next question on the survey, Q4, specifically asked participants if they had changed their major to CS because of the *Virtual Worlds* course. Only one student reported to have done so, stating [emphasis added]:

> *"I love being able to work with a computer on this level and* ***really enjoyed the little programs we made in the class****. Computers are such a* ***huge part of our lives today*** *and along with enjoying it, it* ***seems like a very practical major****"*

This student has reified the ambitions behind the CS Principles framework. The course left a lasting impression, in that they still remember enjoying the programs created as part of the course. The student also noted the importance of the social impacts of computing: "computers are such a huge part of our lives today." This student not only enjoyed learning about CS, but continued to pursue it because "it seems like a very practical major."

Figure 46 summarizes participants' response to items Q5, Q6, Q7, and Q8.

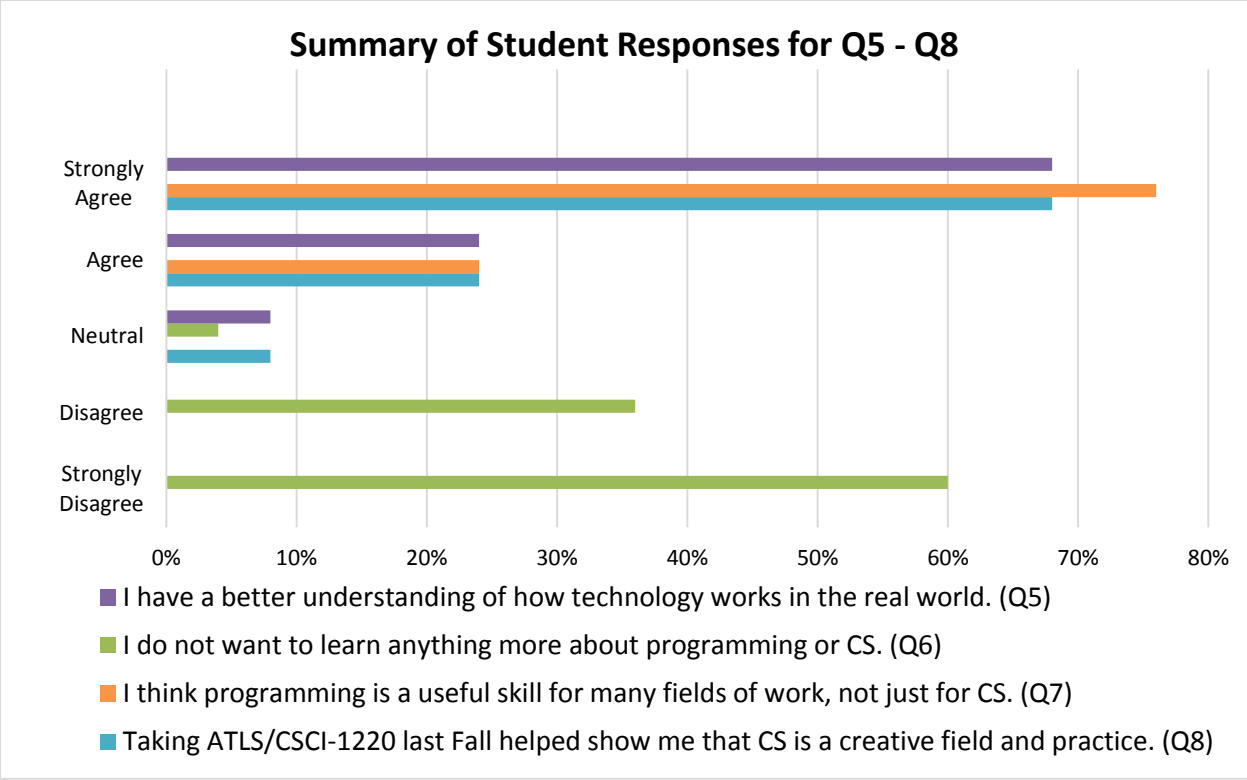


Figure 46. Total Item Responses for Q5-Q8 for One Year Follow-Up Survey

Responses for Q5 indicate that 92% of participants at least "agreed" that they have a better understanding of how technology works in the real world. Note that Q6 was a reverse itemized question, in that "strongly disagree" denotes a positive response and "strongly agree" represents a negative response. These responses suggest that students wanted to learn more about programming or CS in the future. Item Q7 had no neutral or negative responses, indicating that all the students believe that programming is a useful interdisciplinary skill valuable to audiences beyond computer scientists. Finally, 92% of students at least "agree" that CS is a creative field and practice.

Responses to the other survey questions reinforce this conclusion. In general, students held the view that CS Principles helped students (1) have a better understanding of how technology works in the real world, (2) that programming is a useful skill for many fields of work, not just for

computer science, and (3) that CS is a creative field and practice. In addition, when students were asked what was the most memorable aspect of the course (good or bad), most of the students reported positive experiences about creating engaging and fun computer programs, enjoyed the breadth of topics, and thought the lectures were interesting, as can be seen from the following excerpts [emphasis added].

> *"Actually **making programs** and **seeing them work**."*
>
> *"**All the work is fun**, and you help me a lot during your office hour, thank you~"*
>
> *"Attempting to code design a house. **So hard!**"*
>
> *"**I loved the feeling after I succeeded at programming something**."*
>
> *"**I loved the final project**! I had so much fun programming and making the twine story."*
>
> *"**I really enjoyed all the fun activities** we did and **the cool things we made through code.**"*
>
> *"I really **enjoyed learning how to code** and making our own website!"*
>
> *"I **really liked how varied the topics were**."*
>
> *"**Making my own code**."*
>
> *"The **guest talks were great**! And the fun videos at the start of each class."*
>
> *"The jitterbug program. **Pain in the ass but it was cool and rewarding** when i figured it out"*
>
> *"The **speaker event:** online privacy"*
>
> *"**Using the Processing program** to make images"*
>
> *"We **explored some very interesting topics**. I particularly remember the history of computing."*
>
> *"**Completing all the programming** projects for the semester and **being bummed** because I considered **those projects the 'fun projects'**"*
>
> *"I **really enjoyed the actual coding**, even if it was just in Java. It put me in the right mindset to take a 'real' coding class like CSCI-1300 and **made me more interested in the things that computer science could do (like tell a story)**."*
>
> *"**watching your program come to life and work** was definitely the most satisfying part of the class, **making the program was sometimes frustrating but once it works it makes it worth it.** And the professor was easy on the eyes"*

*"**Nothing memorable**, however, I **thought the course was great, very comprehensive.** It was a good introduction enough to enable a student to make an informed decision about whether or not to explore the field further."*

***"The most memorable experiences for me were the lectures in general**-- your wonderful and pleasant attitude coupled with your explanations made the class really exciting. I remember being very fired up after lectures, especially the very first!"*

*"The most memorable experience for me was **getting to make Blinky from Pac-Man out of code** and then changing his colors and making him move around"*

*"**Doing 60% of the work two days before the end of the semester.** While not fun, I knew it would happen; if I had been more familiar with the college process, I would've dropped and signed up for 1310 instead."*

*"Probably when the **guest lecturer** came in to talk about ICT4D, I didn't know anything about that and I was pretty impressed."*

*"The most memorable experience in CSCI 1220 was **learning about Artificial Intelligence and all the future possibilities**."*

Interestingly, none of the surveyed participants mentioned the gaming components of the course. In fact, it was the types of assignments—in particular, the Processing assignments, which focused on creativity and artistic expression—that appeared to leave a more lasting effect on students. This observation further supports the conclusion that gamification of content should be a secondary priority.

Several students retained positive impressions of the instructor, as reflected in the following excerpts [emphasis added]:

*"...I also really like Kara. She was such a **dynamic teacher and brought so much raw enthusiasm to the class and the subject**. I'm really sad I wasn't able to take another class with her before she graduates, but I wish her all the best in her future endeavors!"*

*"The **personality and heart of Kara**. She was a great instructor!"*

*"The night class is an experience all in its own and when the instructor is as good as Kara is, **it makes it easy and super fun to be in class and attentive**! I tried to enroll in another computer science evening course because it was taught by Kara, but was unable to because I am not a computer science major. I recommend Kara to anyone I hear who is interested in taking a computer science course at CU."*

"...Kara's a great instructor."

This issue of whether the "instructor effect" influenced research outcomes is discussed further in the Discussion and Future Work sections of this chapter.

5. Finding 4: CS Principles Instantiates Self-Determination Theory

Overall, the structure and design of the course was successful and appeared to support the choice of Self-Determination Theory as a theoretical framework. Generally speaking, students' attitudes toward the self-paced nature of the course were bimodal, either strongly positive or strongly negative. Students with positive views appeared to benefit from the self-directed learning, showed a higher degree of autonomy, and performed better (had higher final grades in the course). Students with negative views about the self-directed learning aspect of the course offered useful suggestions for making this course better going forward.

The course structure also supported constructionism. Most of the students reported that their "best" experiences in the course happened when they were programming and building computational artifacts. Moreover, students appeared to experience aspects of mastery when overcoming challenges and solving difficult learning problems. Some students reported disliking the reading or "textbook" assignments, and commented that these assignments did not align with their goals and expectations for the course, again pointing toward constructionism as an appropriate model for facilitating introductory computing.

Finally, the variable group exhibited more positive attitudes about connectivism. Although the control group had the same access to forums, forming groups, changing avatars, instant messaging, etc. on the course website, the students in the variable group took greater advantage of these tools, and reported more positive attitudes about working and collaborating with others in the course. This suggests that gamification had more of an impact for connectivist efforts than

stand-alone website features intended to facilitate collaboration and interaction. The conclusion is that connection and collaboration must have a purpose in order to be effective.

5.1. Self-Directed Learning

Self-directed learning is seen as, "…any study form in which individuals have primary responsibility for planning, implementing, and even evaluating the effort" (Knowles, 1975). Most people will claim a preference for assuming such responsibility whenever possible (Hiemstra, 1994). Student comments echo this conclusion, as shown in the following excerpts [emphasis added]:

> *"I liked the **flexible way the course was structured**. It really went well with the **way I prefer to do work and to learn** and I think that it **was one of my better classes** this semester."*
>
> *"I would (and have) recommend this course to my friends **because it is a new and innovative way of learning**."*
>
> *"I thought the **flexibility of the course was the best part about it.** Computer Science seems to be very 'as long as it gets done' culture, and this class embodied it**. I appreciated that most.**"*
>
> *"I love how everything is online. The website, the assignments, etc. Everything being **self paced made things challenging**."*
>
> *"**I liked the deadline freedom** with the quests, too, even though I procrastinated a lot on them a lot."*
>
> *"**I liked the flexibility and the freedom** in the class, making it a less tensed environment to learn, **making feel fun**."*
>
> *"**There is no pressure and no deadlines which I think is perfect for this subject** as you can fit the assignments in gaps in your schedule [and] **they are always fun**."*
>
> *"I **liked the responsibility that the self-paced** class required."*

On the other hand, several students (from both sections of the course) indicated that the lack of deadlines negatively impacted their learning [emphasis added]:

> *"**I really need a deadline** to make me finish assignments on time."*
>
> *"That I **procrastinated** doing some assignments and **didn't get as much out of it** as I could have"*
>
> *"[I disliked] the **ultimate dead lines**."*
>
> *"The only thing I dislike about the course is **the lack of deadlines**. This may just be me, but **deadlines help me stay organized** and on top of the assignments. With how heavy my class load is I constantly **found myself putting this class's work off**, which is **unfortunate**"*
>
> *"...I would assign **more due dates** having only one due date for all of the homework has caused me to fall behind. **I work better with due dates**."*
>
> ***"I did not like the flexible deadlines** because now I have a whole semester's worth of work to do, but this problem was completely created by my lack of time management. **If there was strict deadlines it would have been easier to do homework** over the course of the semester."*
>
> *"**The loose deadlines was a really bad idea**. Especially with the volume and schedule of what was supposed to happen."*
>
> *"I also **didn't like the open ended/ no deadline dates.**"*

There were also students who reported a "love-hate" attitude toward the self-paced structure of the course, noting positive views of the lack of deadlines, but citing the need for deadlines to help structure their work. The following excerpts reflect this attitude [emphasis added]:

> *"The **self paced** was the **best worst thing. I love it but I hate it**."*
>
> *"While I enjoyed not having '**hard deadlines'** it would have **been better to have those deadlines** because it is **easy to fall behind** on the labs."*

> *"**While I liked the self pacing** of the course, I think I would definitely **benefit [from] a broken up deadline requirement** where sets of labs are due at certain times throughout the semester."*
>
> *"I liked having **freedom some of the time**, but **deadlines really do help me**."*

Some students reported that some of the projects were too open-ended [emphasis added]:

> *"My critique of this course would be that **a little more direction** (specifically in relation to the semester projects) **would have been useful**."*
>
> *"The semester project is **too open ended** and I usually **get myself into predicaments** with assignments like that."*

In contrast, other students stated that they enjoyed the open-ended nature of the assignments as an opportunity to explore their creativity [emphasis added]:

> *"**Liked the self paced quests,** creativity and variety of assignments and the **open ended** final project…"*

One student noted the importance of the scaffolding components (a significant component of self-directed learning) to make for a positive learning experience [emphasis added]:

> *"Also the assignments are very well put together and **the instructions are very clear and easy for me** even an international student **[who's] first language is not English**"*

Survey responses from the students also reflect the mixed attitudes about the self-directed learning aspects of the course, as shown in Figure 47.

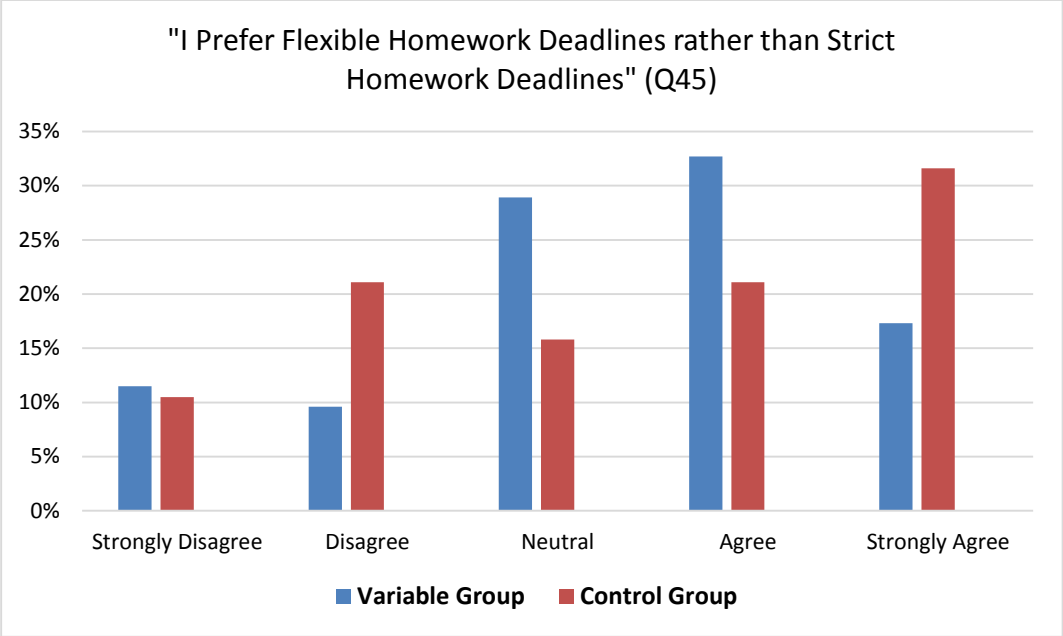


Figure 47. Summary of Q45 Responses for Variable & Control Group

Some students recommended ways to create a compromise between the total lack of deadlines and strict deadlines [emphasis added]:

> *"The class should have some more concrete deadlines. Not all at once* ***but have days where a lot of stuff is due****"*
>
> *"Perhaps the only thing I didn't enjoy this class was* ***the carefree take on assignments…*** *perhaps it could do with* ***at least a few strict deadlines****?"*
>
> *"I think there is a way to* ***compromise*** *[the flexible deadlines] though* ***- deadlines don't have to be strict*** *as in due Mon, late-work only receives -5% for the next week (****so late-work is just really lax****)."*
>
> *"Some assignments* ***could be self paced while others have due dates."***

These comments should be considered in the planning and design of future iterations of this or similar courses.

5.2. Constructionism

Overall, constructionist elements in the course were found to enhance the CS Principles curriculum. Students generally had favorable views of the programming assignments and enjoyed the creative content of the course. The following open-ended comments suggest that constructionist assignments were an effective pedagogical method for some students, and that they viewed challenge in a positive light [emphasis added]:

> *"I actually found the majority of assignments really enjoyable and interesting**. I'm a hands-on learner** and instead of simply watching a lecture, taking notes, and memorizing for a mid-term and/or final exam, **I learned things through a hands-on, self-paced approach**. I love the website and the ability to submit all assignments online."*
>
> *"I really liked all the programming assignments especially how you would have us **start with something and figure out how to change it**."*
>
> *"I **enjoy the difficulty** and amount of assignments."*
>
> *"[What I liked about this course is that] **it challenged me more** than most classes at CU."*
>
> *"**It was a lot of work**, but I think overall it was fine and **I learned something from every quest**."*

However, some students regarded the more open-ended constructionist activities as frustrating and that they needed more direct guidance to construct computational artifacts [emphasis added]:

> *"I like the self guided structure of the homework, though I would be more interested to see more programming taught. **Teaching yourself can be a bit frustrating**."*
>
> *"I did not like the programming aspect simply because I think it **needed to be explained to me in class instead of learning it… on my own** because **it was confusing until it was explained** either in office hours or in class."*
>
> *"The lectures are a necessity for me, **I have trouble learning on my own** and a lecture is where I learn."*

Students appeared to dislike the reading and reflection assignments the most, suggesting that the building of artifacts were viewed as more positive learning experiences [emphasis added]:

> *"Many of the assignments were very similar and **involved lots of reading and reacting**, which **gets tiring and old**."*
>
> *"There was a **lot of reading and writing,** which are particularly long activities for me personally."*
>
> *"**Too much reading** take a lot of my time."*
>
> *"**It was so much writing**. I did more writing in this class than in my writing class. **More programming** please."*
>
> *"There was a **lot of reading and writing,** which are particularly long activities for me personally."*
>
> *"I did not like the assignments on **how we felt about a topic or one that required a lot of reading**."*

Although the self-reflections were intended to aid in the self-directed learning process, and the reading assignments specifically related to computer science topics, these results suggest that the focus should be more about building computational artifacts than reading about them. Seymour Papert would agree.

Other qualitative data indicates constructionist methodologies allowed students the freedom to fail, and demonstrated the importance of feedback, which promoted learning [emphasis added]:

> *"I liked the **status bar,** having no final, and being able to **take quizes more than once**."*
>
> *"I liked the lack of tests a lot. I also liked how the emphasis of the quest **grading was on effort** because it made me feel **more free to experiment with the coding and make mistakes**."*

Web-analytics from the course website show that many students took advantage of redemption. These data suggest that this aspect of the course was used, and viewed positively.

Some students reported that they needed a better way to determine the "level of difficulty" for the individual quests or lab assignments. Several stated they had difficulty assessing the amount of time and the requisite skills needed to complete certain assignments [emphasis added]:

*"I didn't like **so many labs that were too hard to d**o.*

*Some of the quests **are pretty hard**"*

*"I would have liked a length approximation to be given for the written piece so I had an idea of **what was to be expected**."*

Many students complained that the feedback—the green progress bar increasing over time—would have been more useful if grades had been uploaded more promptly. Assessing difficulty level is an important component of effective game design, and in hindsight, this feature should have been added to the course. In addition, in order to enhance proponents of mastery, feedback needs to be given promptly from the instructors and teaching assistants. Prompt feedback and explicit skill-level information thus appear to be important aspects of constructionism.

3. Connectivism

As previously discussed, students generally reported a positive shift in their attitudes about CS. In addition, many students noted the importance and purpose that computer science has on society, as reflected in the following excerpts [emphasis added]:

"*I simply believe that in this point we are at society, **everyone should be learning more about CS**"*

*"I would recommend this course to students because I believe **it can broaden their views** on CS, just like how it has for me"*

*"This course has made me much **more interested in and motivated to continue with CS.** I feel like I already know so much more about technology in general, and how **crucial it is to global development**."*

*"CS is applicable to almost every subject or major! I think e**veryone would benefit from learning this stuff.** They can take what they want from it and **apply it to their majors/lives/careers in their own creative ways.**"*

*"this course it taught me so much about computer and **in this day and age** that is an **important thing to know**."*

> *"I think it's **important to understand basic CS principles**, and this class does that well while being interesting and enjoyable at the same time."*

In general, most students viewed the course content as valuable and meaningful, beyond the context of the course itself. In terms of the more explicit technological connectivist portions of the course—specifically the social media features on the site—the variable group was observed to utilize these options much more than the control group. For example, eight different forums were created by students in the variable group, with 20 separate threads posted by students (not counting the self-reflection assignments). Students had the option to post self-reflection assignments publicly on the forum or privately, by uploading their assignments. In the variable group, there were over 35 topics in the self-reflection forum, totaling 318 posts by students. By contrast, in the control group, only one student posted in the forums *once* during the entire semester. No students posted their self-reflections in the forums. In addition, students in the variable group created seven "groups," averaging eight per group. No students in the control group created "groups" on the course website. Students in the variable group viewed the social aspects of the course positively, as reflected in comments [emphasis added]:

> *"[I liked] taking a **different approach to traditional methods** of teaching a class, self paced homework, **creating an online class community.**"*

> *"I really like the way [the website was] formatted. The **quests are motivating** and the **forums make it interactive**."*

Some students stated they created groups to earn badges and increase their rank on the leaderboard, whereas other students formed study-groups and other student resources not for an extrinsic purpose, but for the sake of helping others.

In summary, both quantitative data (web analytics) and qualitative data (from student comments) suggest that the connectivist aspects of the course were far more successful with the variable group than the control group.

VII. DISCUSSION

1. The Impact of Gamification on Student Motivation & Learning Outcomes

Overall, these findings support a conclusion that gamification supported some students' desire to complete the course material, by creating both extrinsic and intrinsic motivations. However, although gamification seems to have provided a meaningful motivation for some students to complete the course learning objectives, the gamification elements did not contribute to student learning in a statistically significant way. These findings suggest that gamification can be used as a "lens" for the zone of proximal development for students who are already intrinsically and extrinsically motivated to learn the content. This finding is supported by both quantitative and qualitative data.

One significant finding of this work is that extrinsic motivations were observed to be an important component of undergraduate student experience and expectations. One of the criticisms of gamification is that the focus on extrinsic motivation can be detrimental or harmful, particularly if the goal is to increase intrinsic motivation. Prior work refers to "exploitationware" or being "punished by rewards" (Kohn, 1993). However, many students in this study indicated that they prefer—or even require—the use of extrinsic motivation in order to be successful in their learning. Most often this extrinsic motivation is the course grades, but in this instance, extrinsic motivation included deadlines. Thus, extrinsic motivations are not universally harmful, and intrinsic motivations are not universally positive. A balanced combination of both intrinsic and extrinsic motivations likely needs to be part of the course design for gamification to have positive effect.

2. Gaming is Culturally Pervasive

Our results also demonstrate that one does not have to be a "gamer" in order to benefit from gamification. In fact, some of the most successful students in the course, who appeared to benefit most from gamification, were non-gamers. A key result of this work is that people who do not actively play or identify with gaming culture can benefit from gaming in their classes. Interestingly, some of the most avid "gamers" felt that the gamification present in the course did not go far enough. As with many components of user-centered design and human-computer interaction research, design is situationally dependent upon the context into which it is embedded—culturally, socially, theoretically, and practically.

3. Impact of CS Principles on Student Motivation & Learning Outcomes

Results from this case study show that CS Principles is an effective framework for introducing students to the fundamentals of computer science. Results from the post-test and the one year follow-up survey show that many students went on to pursue CS in some capacity. In addition, CS Principles positively influenced student perception of computer science and attitudes about the field as a whole. Both quantitative and qualitative data showed that students perceive CS as a creative, socially relevant, and important field of practice. Our findings reinforce the idea that students' previously conceived perceptions of the field may be one reason why so many students do not believe programming is a creative practice nor even useful skill outside of the scope computer science. However, when the content is presented in an engaging way, this can help change previous conceptions about the field can be altered. In particular, many non-majority students in the course self-reported to having positive changes and attitudes about CS explicitly because of the course. This finding suggests that CS Principles is a step in the right direction for

creating more engaging and compelling curricula to diverse groups of students, especially those with minimal experience and exposure in the field.

Another important finding of this work is that students went on to pursue computer science in a variety ways. Where some students decided to change their major to CS, others pursued the study of CS in different ways, such as engaging in non-formal learning environments like online courses, and utilizing computational thinking skills in other creative and career efforts. Overall, many Virtual Worlds students now consider CS as creative field of practice, fostering more positive attitudes about the field, and increased the levels of interest among these students.

4. Issues Pertaining to Internal Validity

There are several issues regarding internal validity with this research. Although case studies anticipate that design iterations and research findings are bound to the time and context of the event, there are important aspects to consider. First, as both the researcher and teacher of this work, my biases and experiences affect the research in important ways. For example, several students reported or identified the instructor's enthusiasm for the course as having an impact on their learning and the classroom environment. Although this is not a bad outcome, it complicates the research evaluation.

I had also initially anticipated teaching this course again during the Spring or Fall 2015 semester. This course of action turned out to be infeasible. This research will be strengthened in the future if other instructors teach the same course using similar structure and materials.

5. Summary of Findings

In summary, the introduction game design elements were not found to have a strong impact on student motivation and learning. Although many students responded positively to the

gamification, some indifferent. Most significantly, the effect of the CS Principles curriculum itself was the strongest contributor to positive learning of introductory computer science. This was not so much whether there were game elements, but rather how the course overall was structured that was most beneficial to the students. Students who are more extrinsically motivated need more strict deadlines and rubric evaluations for their work, whereas students who are intrinsically motivated responded positively to the game design elements. The conclusion is that the design of both the gamification and the course structure need to align with the intrinsic and extrinsic goals of the students in order to be effective. When done well, this approach can enhance the "zone of proximal development," where gamification can act as the lens that magnifies the learning experience, rather than sustaining it all on its own.

VIII. CONCLUSIONS AND FUTURE WORK

There are many opportunities for future work related to both gamification and CS Principles. First, the gamification of a course website seemed to work well in an undergraduate classroom environment. Gamification needs to align with situated-design and user-centered design best practices in order to be effective for students. Our results showed that faster feedback cycles, more clarity on the use and purpose of game scoring like badges and points, and more appropriate use of extrinsic motivations could potentially create a more positive user-experience for students in the course. For example, the use of pop-ups on the website to denote that achievements were unlocked or that badges were awarded could prove to be more useful than a static list of badges and all possible rewards. In addition, a "skill tree" that is mapped using more conventional gaming constructs could also prove useful in helping students navigate and complete their coursework. In addition, the application of "real" tangible rewards—such as extra credit, additional learning experiences, and other extrinsic motivating factors could also be a useful

follow-on, since some students seemed to have benefitted more from extrinsic factors than to intrinsic motivations. Further, the application of gamification outside of the realm of computer science education into other STEM topics is also worthy of consideration.

In addition, this course has opportunity to be distributed as an online course or used in a flipped-classroom methodology. For example, some students greatly benefited from the traditional lectures, whereas other students thought the lectures were disconnected with what they were actually doing in the homework assignments. This shows that there needs to be more consideration in how to apply situated learning techniques into the context of an online course, particularly in the realm of constructionism and connectivism. For example, since all of the course interactions of students happened on the website as opposed to a course recitation or even during lectures, applying this work into a "distance learning" context is a logical follow-up of this research. Distance learning literature and research, whether it is Massively Online Open Courses (MOOCs) or an online undergraduate course, are often concerned about motivating and sustaining students to complete all of the online learning objectives. Gamification may be an effective way to help students engage in these courses, particularly since the data shows the variable group interacted more with forums, peer-to-peer connections, and connectivist techniques than in the control group. Therefore, gamification research should be extended into the realm of distance learning in order to evaluate if these effects are similar to the student experiences in this course.

CHAPTER VI
CASE STUDY TWO: TECHNOVATION

The second case study considers the benefit of gamification as a scaffolding support system for minority students in a CS extracurricular high school program. Although previous research has explored students' attitudes toward CS in the context of K-12 education (Margolis & Fisher, 2002; Ashcraft & Blithe, 2009) and higher education (Guzdial et al., 2010), extracurricular CS programs have received limited attention.], particularly among non-majority groups. This case study examined gamification as a mechanism for increasing the engagement of teenaged Latinas in an extracurricular high school program incorporating significant CS content.

The extracurricular program, called Technovation, is a large and long running non-profit technology program for girls and young women. Technovation is an international program, developed by Iridescent, a 501c3 nonprofit science education organization in the United States. In Technovation, teams of young women identify a problem, create a mobile app to solve it, program the app, build a company to launch the app in the market, and pitch their plan to experts—all within 3 months. This study examines how gamification may supported the intended learning outcomes for Technovation. This case study specifically focuses on how "at-risk"[41] Latina teenagers respond to gamification in the program. Although the Technovation curriculum was not modeled or designed from CS Principles, or any other CSTA Standards, the core objectives Technovation are complementary to these standards. As in the first case study, I acted as the teacher-researcher,

[41] The term "at-risk" is often used to describe students or groups of students who are considered to have a higher probability of failing academically because of circumstances that could jeopardize their ability to complete school, such as homelessness, incarceration, teenage pregnancy, domestic violence, transiency (migrant-worker families), English as a second language, among other socio-cultural factors.

which provided me the flexibility and control to iteratively design and develop the gamification intervention and to facilitate data-collection.

I. SITE SELECTION

The Technovation program was conducted at a high school campus in Denver during Spring 2015. I received granted permission to conduct research on the Technovation program by the Technovation leadership, the high school principal, the Denver Public School District Research Review Board (RRB), and the University of Colorado Boulder's Institutional Review Board (IRB).

1. Duration of Study

The program ran for thirteen weeks between January and April, 2015. Data-collection occurred during regularly scheduled Technovation sessions at the high school campus. Technovation program sessions occurred after-school at the high school campus on Mondays from 4:15-6:15PM, and after-school on Wednesdays from 2:15-4:15PM.

2. Description of Student Population

Participants were female high school students who enrolled into the 2015 Technovation program via the non-profit center located on the high school campus. Students who enrolled in the Technovation program at the high school were conveniently sampled to participate in the research. All participants are female, and 14 to 18-years-old (freshmen to senior high school students). The urban high school is a predominately Latino high school; all participating subjects were considered to be part of a vulnerable population.

3. Participation Benefits

To encourage students to enroll and participate in Technovation, the non-profit community center leading the program provided each student a stipend of $150 upon completion of the twelve-week program. Students had to demonstrate an "80% attendance and participation rate" in order to qualify for this stipend. Incorporating an extrinsic reward, such as money, into the study created a unique opportunity to test a potentially "harmful" effect, since extrinsic factors such as monetary gain have been shown to negatively impact intrinsic motivation in prior SDT research (Ryan, Connell & Grolnic, 1993). I had no control over the implementation of this extrinsic reward, so it was considered as another independent variable when assessing student motivation and its impact on the intended learning outcomes of the program.

II. RESEARCH DESIGN

1. Theoretical Framework

The Technovation curriculum follows a recommended course structure and judging rubric for students and coaches to use throughout the duration of the program. This curriculum offers limited flexibility to modify lesson objectives. However, there was sufficient flexibility to incorporate CS Principles concepts, and to incorporate three key aspects of gamification: self-directed learning, constructionism, and connectivism.

1.1. Self-Directed Autonomy

The girls in Technovation lead the creative-direction of the program. The girls choose what community problem they want to investigate, brainstorm their solutions to that problem, and develop a mobile app that contributes to these solutions. The "coaches" and "mentors" provide

feedback, but do not direct student creative activity. Technovation encourages the coaches and mentors to step-out of a leading instructing role and to encourage the girls to self-direct their learning experience.

1.2. Constructing Mastery

Technovation students must actively construct and evaluate their artifacts throughout the program. For example, consider the Technovation Lesson 4 tasks and objectives:

- Revise market size calculations based on feedback
- User interface and product design – paper prototyping
- Product Description
- Get feedback on paper prototype from peers/potential users
- Get feedback on product description from Teachers/Mentors

First, students take their initial app ideas and construct them into tangible paper prototypes. Next, they get feedback on their constructed prototype from classmates, potential users, teachers, and mentors. In this example, the girls construct their ideas into tangible artifacts and learn from that construction by gaining immediate feedback on their designs from others.

1.3. Connectivist Purpose

The Technovation program encourages coaches to seek and recruit mentors for each team engaged in the program. Their reasoning behind this is that:

> "Mentors act as guides and role models during Technovation's 12-lesson curriculum… Participants look to their mentors for guidance on how to overcome obstacles and solve problems… After young women observe how their mentor

works through challenges, they will be more confident in overcoming similar challenges by themselves in the future." (TechnovationChallenge.org, 2015)

Technovation mentors help build active, connectivist community by fostering a sense of purpose and relatedness. Even the ways in which coaches search and recruit mentors facilitates connectivism. For example, coaches can use TechnovationChallenge.org website to find a "virtual mentor" for the students, since a local mentor may be hard to identify. After scrolling through a list of volunteers and scanning through their credentials and brief biographical descriptions, coaches can select a mentor that is connected with the community: in my case, a first-generation Latina who worked her way through college and became a successful software engineer. The importance of role models is well known; successful Technovation program require that the participants be exposed to female computer scientists and engineers.

III. IMPLEMENTATION

1. Curriculum Structure

According to the CSTA Standards, Level Two (see Appendix A) is the appropriate guideline for high school students in demonstrating the five strands of content for this age group. The Technovation curriculum facilitates all five content strands call in the CSTA Standards, as summarized in Table 23.

CSTA Strands	Technovation Program Goals
Community, Global, & Ethical Impacts	Create a business and mobile application that solves a real problem in your local community; learn how to communicate to others about the issue and solving that issue

Collaboration	Collaborate with others and form a team (5 members total) to build an app and create a company for the app
Computational Thinking	Learn how to brainstorm solutions, use iteration, user-center design, and prototyping to design and test the app
Computers & Communication Devices	Conduct market research and competitive analysis to learn about other computer solutions
Computing Practice & Programming	Design, prototype, build, and implement a mobile application for your business

Table 23. Comparison of Technovation Objectives & CSTA Strands

The Technovation program offers a "coach/mentor lesson guide," which provides a structured curriculum that coaches may follow for the duration of the program. The Technovation curriculum is modular and flexible, and is designed to accommodate different learning environments and strategies. The curriculum provides general guidelines to help coaches and mentors facilitate the program. The recommended curriculum spans twelve weeks, with each week focusing on a particular lesson or topic. The table below lists the "official" Technovation curriculum structure:

Lesson	**Topics**
1	Introduction to Technovation Career Exploration—Get to know your mentor! "Talk to Me" App Inventor tutorial
2	Ideation—Brainstorming community issues Lean Startup Design a survey—get feedback on the issues "Collection of Games" App Inventor tutorial
3	Ideation—Brainstorming solutions Design a survey—get feedback on solutions **Potential Market Size** Get feedback on Market Size from Teachers/Mentors "Maps" App Inventor tutorial
4	Revise Market Size calculations based on feedback User Interface and Product Design – Paper Prototyping **Product Description**

	Get feedback on paper prototype from peers/potential users Get feedback on product description from Teachers/Mentors
5	Revise Product Description based on feedback Revise paper prototypes based on feedback and transfer to AI Plan what you want to accomplish each week **Competitive Analysis** Get feedback on Competitive Analysis from Teachers/Mentors
6	Revise Competitive Analysis from Pricing based on feedback **Branding and Promotion** Get feedback on Marketing plan from Teachers/Mentors Continue working on Prototype
7	**Potential Revenue** Revise branding and promotion plan based on feedback Continue working on prototype
8	Revise potential revenue based on feedback **Pitch Guidelines—**Plan pitch video, write out script Get feedback on pitch video plan Continue working on prototype
9	**Demo Video Guidelines**—plan demo video Begin filming pitch video Continue working on prototype Get feedback from peers/potential users for your app so far
10	Begin filming demo video Continue working on prototype Continue working on pitch video
11	Edit videos Put together business plan Review deliverables Continue working on prototype
12	Make any last edits and revisions Reflection Submission Post-Surveys

Table 24. Technovation Course Curriculum and Weekly Lessons

The bold-face phrases in Table 26—Potential Market Size, Product Description, Competitive Analysis, Branding & Promotion, Potential Revenue, Pitch Guidelines, and Demo Video Guidelines—identify the learning objectives and expected program deliverables, that each team of students is required to complete to be considered eligible for judging during the first round of the international competition.

Technovation permits the development of mobile apps using any programming or API environment, including Objective C and Swift (iOS), C# (Windows Phone), and Java (Android OS). The Technovation curriculum recommends that students use MIT's App Inventor to build apps (see *Figure 48*), since most Technovation students are learning introductory computer science for the first time. The recommended CS tutorials in the official curriculum are all App Inventor tutorials. App Inventor is similar to Scratch (Malan & Leitner, 2007); students "snap" together blocks to write code.

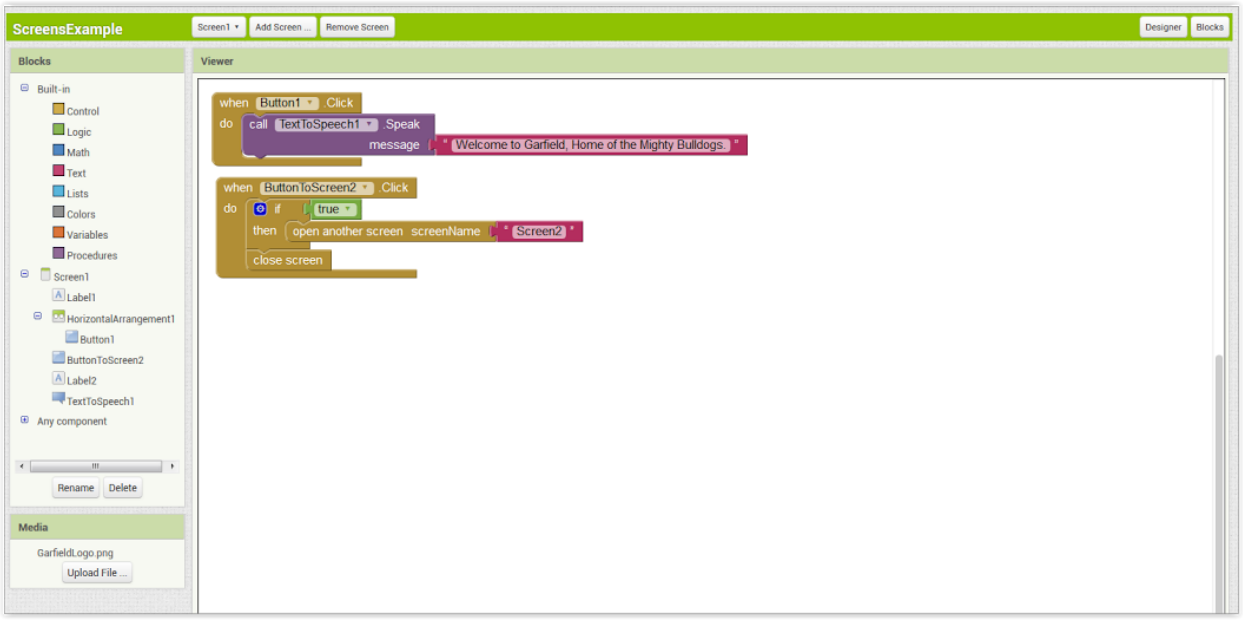


Figure 48. App Inventor's Block-Programming Environment

Students can also design the interface of the app through the "Designer" display of App Inventor. Similar to the Blocks screen, the Designer screen allows users to drag-and-drop buttons, upload images, implement text fields, and create other types of user-interface components.

The Technovation curriculum provides coaches/teachers PowerPoint slides, beginner programming tutorials, and other instructional content. The curriculum also provides "Lesson

Summaries," which include "modules" that outline the topics and activities for the lesson, "objectives" for the lesson, and recommended "content" taught for that lesson. Figure 49 depicts the recommended curriculum structure for Lesson 3:

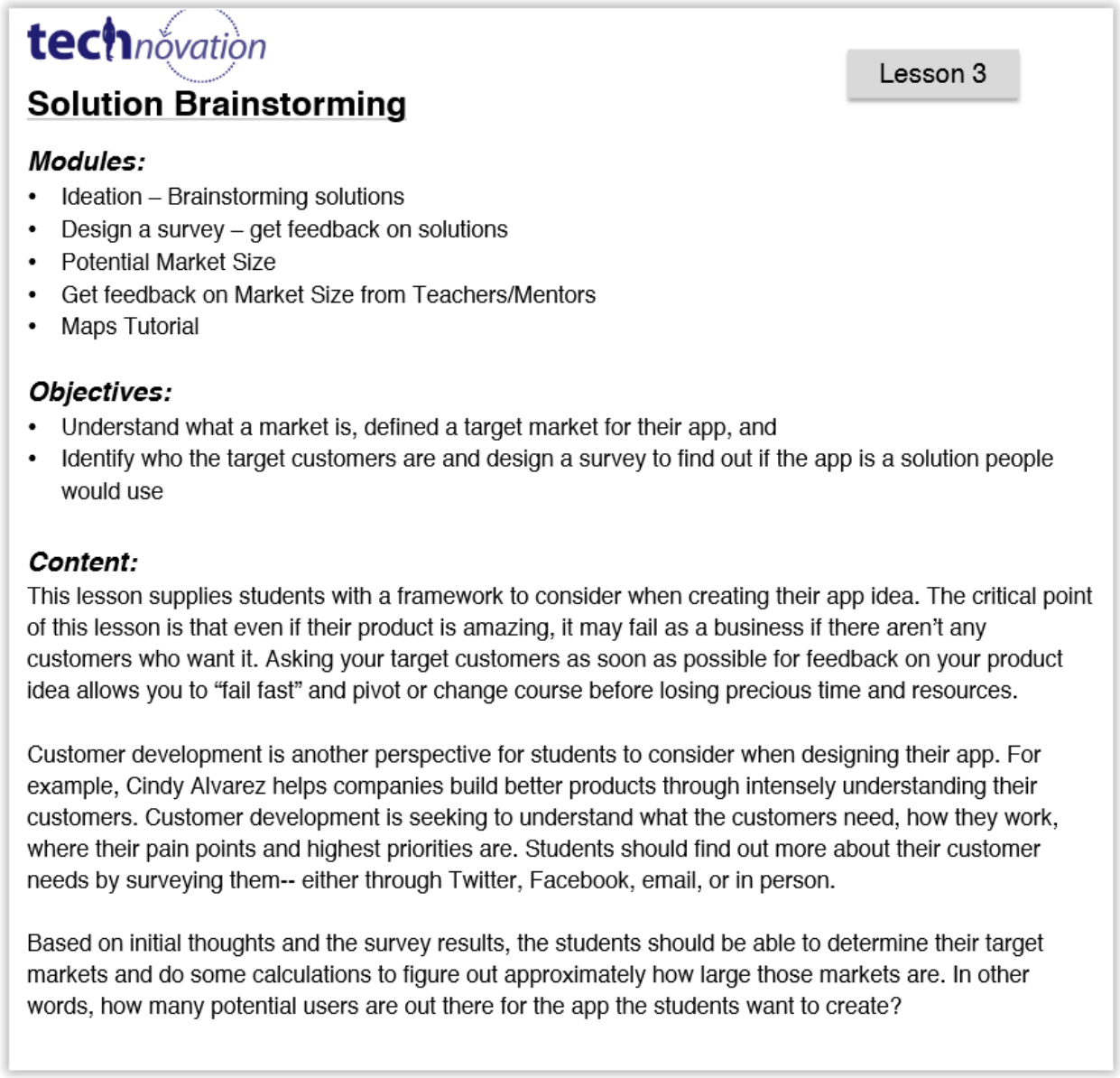


Figure 49. Example Lesson Strategy for Technovation

The non-profit community program supporting the class required the use of the recommended Technovation course structure for the duration of the program. This requirement was intended to

allow standardized assessment by the Technovation leadership. Working within these requirements, I introduced gamification into the program, as described below.

2. Learning Objectives

The Technovation program requires students to submit six deliverables in order to compete in the international competition. The "final deliverables" for Technovation are:

1) 100-word app description
2) App prototype source code
3) 3-5 screenshots of the app prototype
4) Pitch video on YouTube under 4 minutes (+/- a few seconds)
5) App demo video on YouTube under 2 minutes (+/- a few seconds)
6) Business plan—typed and in PDF format

Technovation also provides the official Judging Rubric that the judges will use to evaluate students' work. The judging rubric for Technovation contains both objective and subjective scoring criteria. The objective score is an aggregate of the *Ideation*, *Technical*, and *Entrepreneurial* scores. The complete judging rubric is included in Appendix G. As the judging rubric demonstrates, the learning outcomes for the program are explicitly provided to both coaches and students. I structured the gamified course website around the recommended course structure and lesson objectives in an effort to make the program as successful for the students as possible.

3. Course Website

I used a modified Virtual Worlds WordPress and BadgeOS LMS for the Technovation website. I registered the website under its own domain name, so that I could implement appropriate privacy protection for the site, since the participants were considered to be vulnerable population.

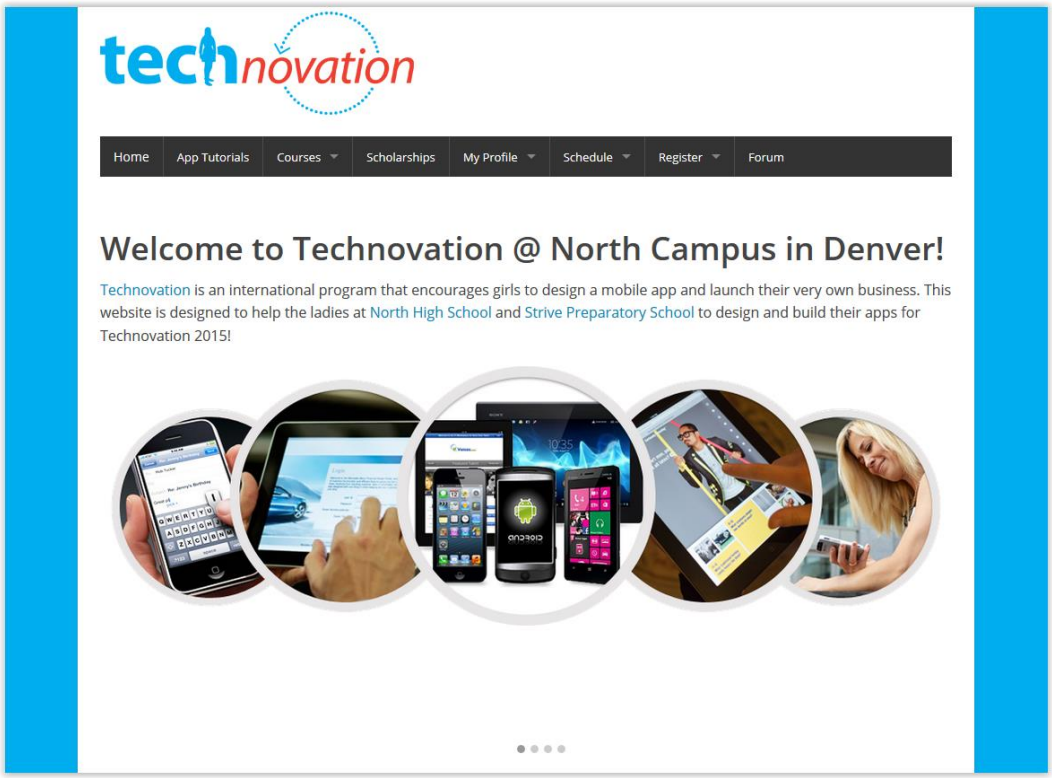


Figure 50. Homepage of Gamified Technovation Website

Similar to the first case study, the website provided a way for students to recieve rapid feedback on their progress. This website uses the same structure as the first case study, including a progress bar, list of completed objectives, and awarded badges.

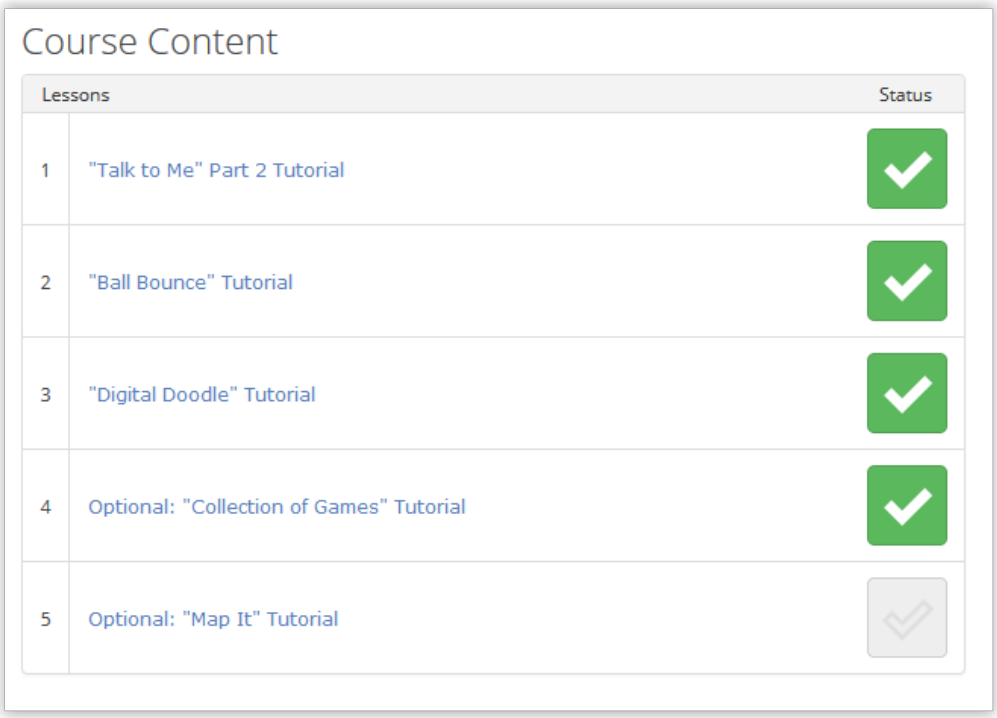


Figure 51. Feedback for Completing Unit Lessons

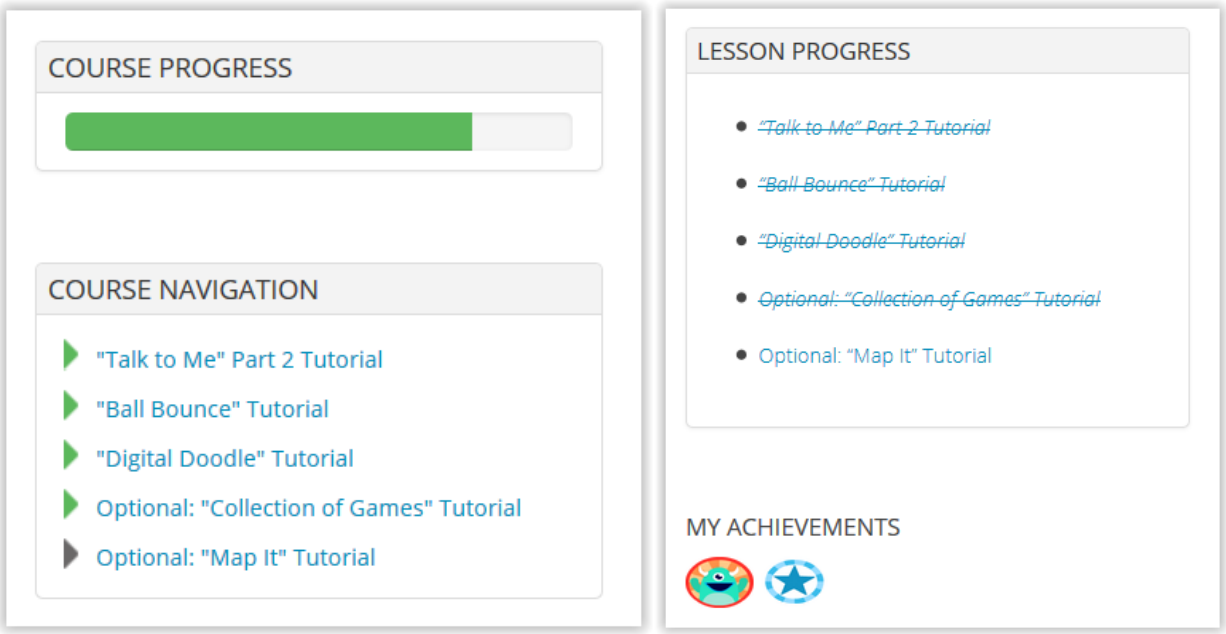


Figure 52. Secondary Feedback via the Website

However, rather than creating a leaderboard, the Technovation website incorporated a "member list" in an effort to support connectivist community engagement.

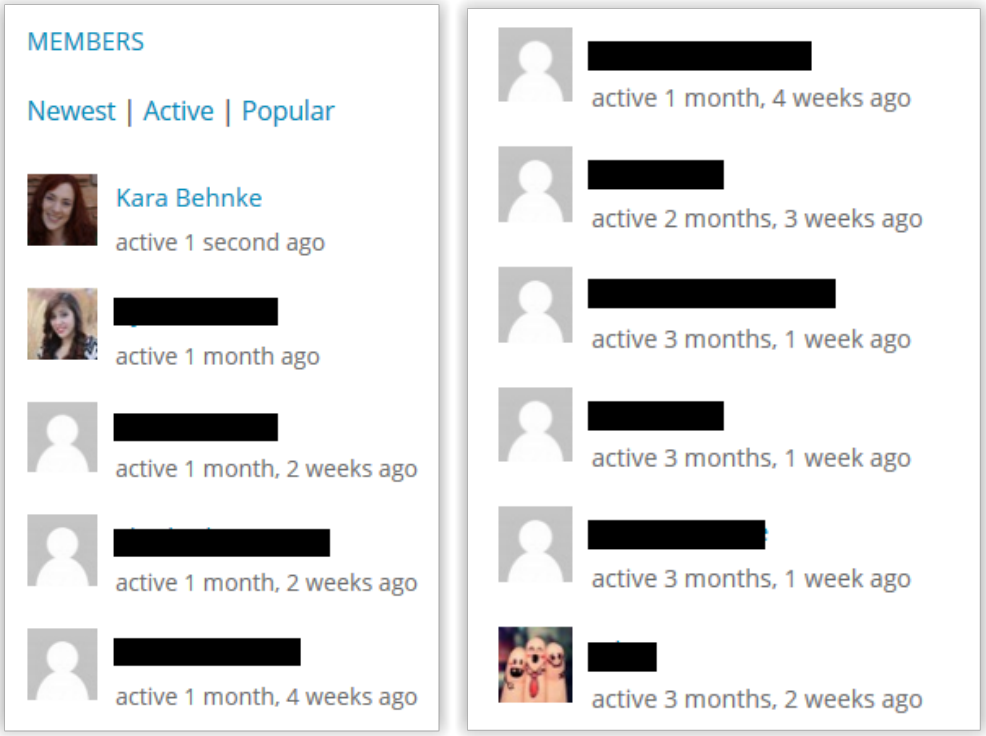


Figure 53. Technovation Members List

Since Technovation explicitly requires students to form teams and complete group-work, the use of a leaderboard was not consistent with the course learning objectives.

Technovation also provides a "Student Workbook" that the girls use to structure program tasks. The student workbook also provides students with learning exercises and materials (see *Figure 54*).

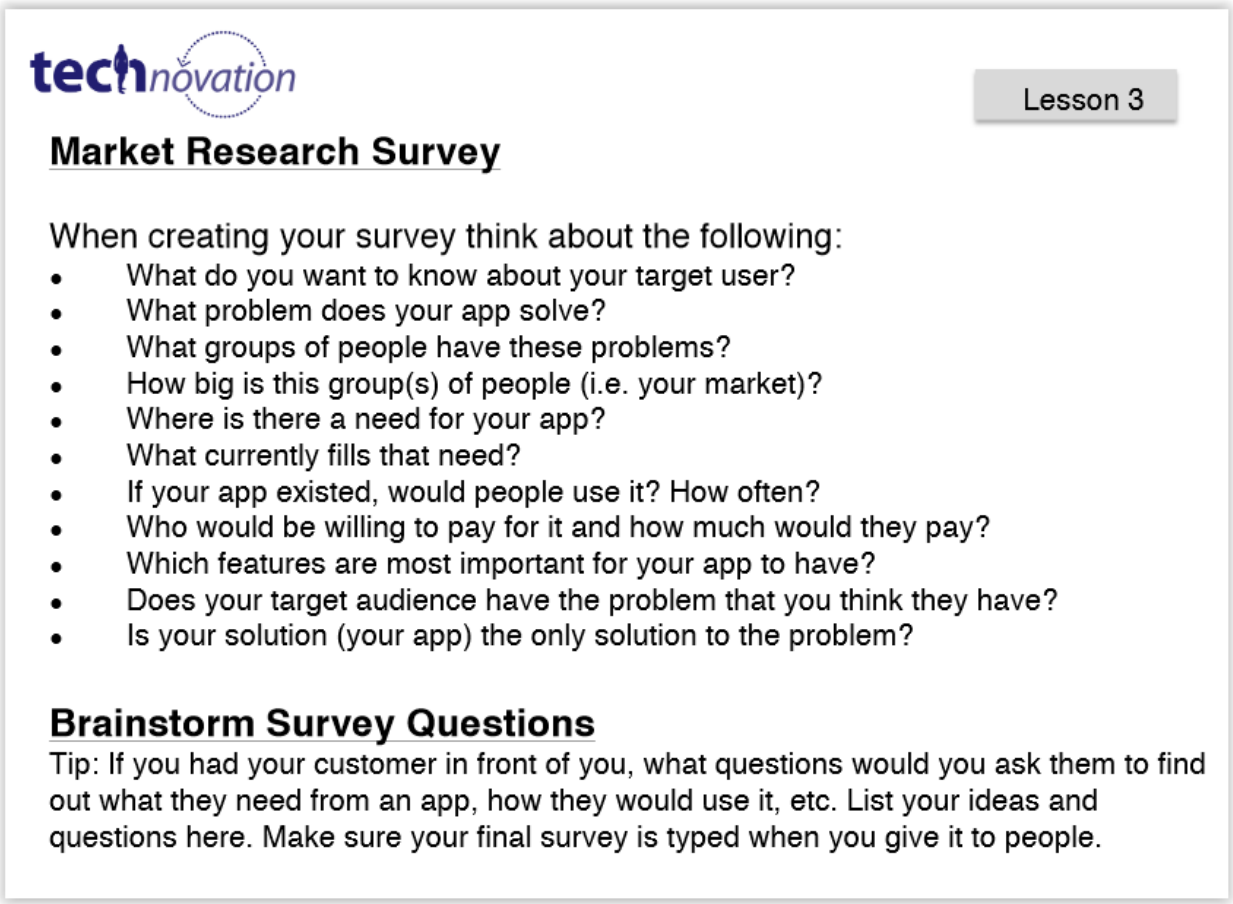


Figure 54. Example of Student Workbook Unit (Hardcopy)

I converted the provided card copy course materials into a digital format on the course website, thereby allowing gamification features to be incorporated into the student's primary course online content document. Examples of hardcopy and website materials are shown in Figure 54 and 55.

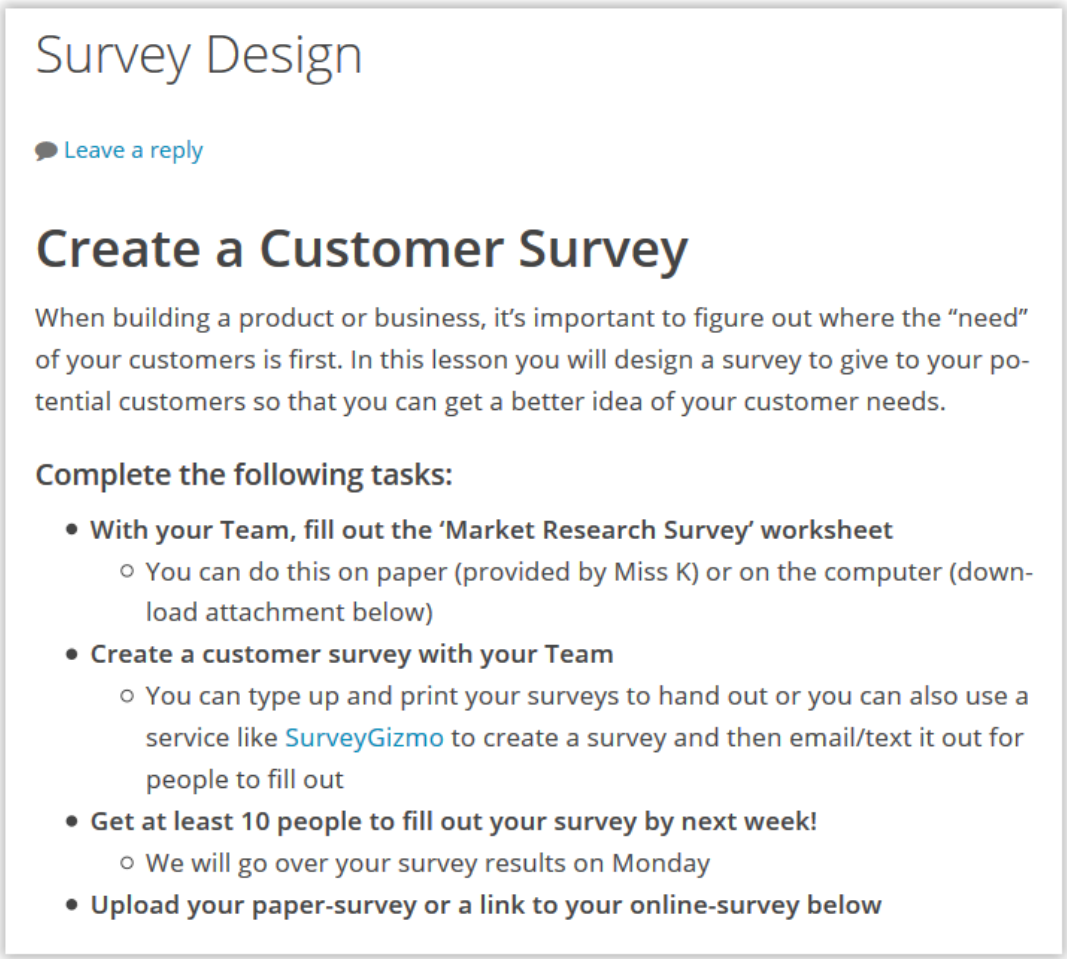


Figure 55. Example of Student Workbook Unit via Website

All course learning exercises and materials were transferred to the course website in this manner.

IV. EVALUATION

1. Procedures

Primary data for this case study was collected in the form of quantitative pre-test and post-test surveys, focus group interviews, and website metadata.

1.1. Pre-test & Post-test Surveys

The pre-test and post-test surveys incorporate IMI items in addition to questions that measure baseline knowledge. The surveys employ a five-point Likert scale to compute "student motivation" and "learning outcomes" into ordinal variables. The full pre-test/post-test survey can be viewed in Appendix H. The items specifically related to IMI items or student motivation are Q1, Q4, Q10, Q11, Q13, and Q14. The items related to learning outcomes are Q2, Q3, Q5, Q6, Q7, Q8, Q9, Q12, Q15, and Q17. I used *t*-tests to evaluate these data.

1.2. Experience Sampling Method Survey

An Experience Sampling Method (ESM) survey was randomly distributed at the end of a session activity in the middle of the program. ESM surveys seek to capture an individual's level of engagement and affect in an activity. This survey was administered in an effort to measure student's perceived flow during one of the afterschool activities. Questions were conceptualized and itemized to measure student interest, enjoyment, concentration, immersion, challenge, skills, and importance. Previous work has shown that interest, enjoyment, and concentration are prerequisites for flow (Csikszentmihalyi, 2001). Immersion, a central aspect of flow experiences, is related to learning; challenge and skill. Importance and meaningfulness are also theorized to be key conditions for flow (Shernoff, Hamari, & Rowe 2012). Appendix I contains the full text of the ESM survey.

1.3. Focus Group Interviews

Two focus groups were conducted during the study. The first focus group session occurred in the middle of the Technovation program; the second occurred at the conclusion of the program. The focus groups averaged approximately twenty-five minutes of a regularly scheduled lesson

time on the high school campus. I used semi-structured interview questions to investigate key research questions during the discussion, and allowed emergent questions to arise when appropriate in order facilitate the discussion. For example, I asked direct questions to the students regarding their level of motivation to attend the program whether due to personal interest, or external influences (e.g., the money stipend). All focus groups were video-recorded and transcribed for qualitative analysis. Because I incorporated research questions and hypothesis into the study design, inductive methods (Strauss & Corbin, 1990) such as open-coding and axial coding procedures (Lindlof & Taylor 2011) were used to create categories for content analysis. Appendix J contains the focus group protocol that was used for both sessions.

1.1. Web Analytics

Similar to the first case study, web analytics were collected as a primary measure of quantitative observation. The gamification intervention consistent of a digital website, so any information provided by participants on the website was collected as primary data, including user names, profile avatars, number of visits, number of comments, lesson submissions and time-stamps. Descriptive statistics of this data were analyzed to interpret user attitudes and behavior (Ochoa & Duval, 2011; Ducheneaut et al., 2007).

V. FINDINGS

1. Summary of Findings

In general, the Technovation program did not fulfill its learning objectives, in large part because it did not meet the expectations of participants. Most of the girls thought that the Technovation program would have more focus on app development and programming than business and entrepreneurship. This issue was discussed at length during the focus group

interviews. Although the girls demonstrated learning in some areas, there was little participant interest in learning about business skills. Moreover, the gamification intervention was not successful. Part-way through the program, a second gamification intervention was designed and implemented, which was somewhat more successful than the website. The desire to utilize the CSTA Standards as a framework also met with limited success. Although the CSTA offers guidelines for "levels" of curriculum and learning outcomes according to grade and age, this model did not prove successful in practice. Because some students had previous experience in programming, and other students had no experience whatsoever, the CSTA Standard Levels were not necessarily useful for guiding the design and implementation of curriculum.

2. Limited Success in Learning Outcomes

A total of eleven girls initially enrolled into the program; seven girls (64%) completed the program with 80% participation and attendance rate. Two of these girls had previous experience using App Inventor; one had participated in the 2014 Technovation season. Only one of the two teams was successful in submitting all of the Technovation Deliverables by the submission deadline. The other team was unsuccessful in part because several team members dropped out of the program after teams had already been organized and selected. This resulted in only two girls participating on one team, and five girls participating in the other. The team with two girls struggled to complete two of the six deliverables (i.e., create the app source code and business plan).

The pre-test and post-test was used to measure any affect in student learning outcomes. Results show that the girls demonstrated an increase in knowledge about (1) understanding the design process used to create technology products (Q3); (2) understanding what a prototype is (Q5); (3) incorporating knowledge of the consumer into the products that they develop (Q6); (4)

understanding how technology products are marketed (Q8); (5) understanding user-interface design (Q9); and (6) how to develop code for a mobile application (Q12). The table below summarizes these results.

	Pre Q3	Post Q3	Pre Q5	Post Q5	Pre Q6	Post Q6	Pre Q8	Post Q8	Pre Q9	Post Q9	Pre Q12	Post Q12
Mean	2.57	4.0	3.29	5	3.14	5	2.85	4.71	2.43	4.43	2.42	4
Mean Differ-ence	0.14		1.71		1.86		1.85		2		1.57	
***p*-value (two-tailed) ≤ 0.05**	0.025		0.03		0.0015		0.0067		0.0096		0.017	

Table 25. Summary of Successful Pre-Test & Post-Test Learning Outcomes

The learning outcomes that most directly relate to CS or CSTA Standards were Q5, Q9 and Q12. Approximately half of the learning objectives were successfully achieved.

2.1. Technovation Did Not Meet Student Expectations

One of the biggest complaints or criticisms of the Technovation program was the perceived "false advertising" of Technovation as a program about app development. Many of the girls expressed more interest in learning about computer science and programming than entrepreneurship, but the curriculum objectives had them spending most of their time developing the business-oriented components, such as developing market surveys, competitive analysis, paper-prototyping, and developing a business plan, pitch video, etc. Only three of the twelve lessons mentioned computer programming, and these materials were simply references to MIT App Inventor tutorials. Clearly, there needs to be more emphasis on the computer science

curriculum. Although the Technovation curriculum is designed to be flexible and can be modified as coaches see fit, as a stand-alone curriculum, it is not an effective introductory CS program.

2.2. Positive Learning Experiences with App Design

During the second focus group, when students were prompted to explain what they enjoyed most about Technovation, the majority expressed that the app design was the most positive learning experience. Several of the students enjoyed the computational thinking processes involved with the design and building of the app [emphasis added]:

> *"**I *love* building the app**… I think it was just what I was expecting of the program… learning how to build an app, **how to make the blocks work, how to make the screens connect**, how to make an app!"*

> *"The process of making an app [was really interesting]… like, **how you have to combine the blocks.**"*

Although some students mentioned that they did not necessarily like the paper prototyping process because that lesson took "too long," one student valued its importance in the design process [emphasis added]:

> *"[The prototype] was too long…maybe **it is necessary** because I definitely think that having all those papers that I was creating…I was like, 'what am I missing? What am I missing?' …cuz **I couldn't remember what I had to put in the app, so it was useful**."*

Overall, the app design process had the most positive impact on students throughout the program.

3. Gamification Not Successful

Findings show that the gamification intervention—the website—was not useful, nor did it help motivate them to complete course objectives. Focus-group interviews point to technical difficulties experienced at the beginning of the program, "[the website] didn't work at the beginning so it was hard to use," as detracting from a desire to use it, even after issues were resolved. In addition, the girls indicated that they "didn't care" about the gamification systems—

badges, points, etc. This is not to say that they did not understand the mechanics behind such artifacts, they simply did not value them: "[we would] rather communicate how we normally do like through our phones and texting." I also struggled to communicate with the girls via the website and e-mail—text messaging them directly was a far more efficient and effective way of communicating. This observation demonstrates the importance of *situated design* (Simonsen et al., 2014) when it comes to designing and implementing gamification, as explained below.

3.1. The Importance of Situated Design

Figure 56 shows the orientation of the site classroom setup during a typical day for Technovation.

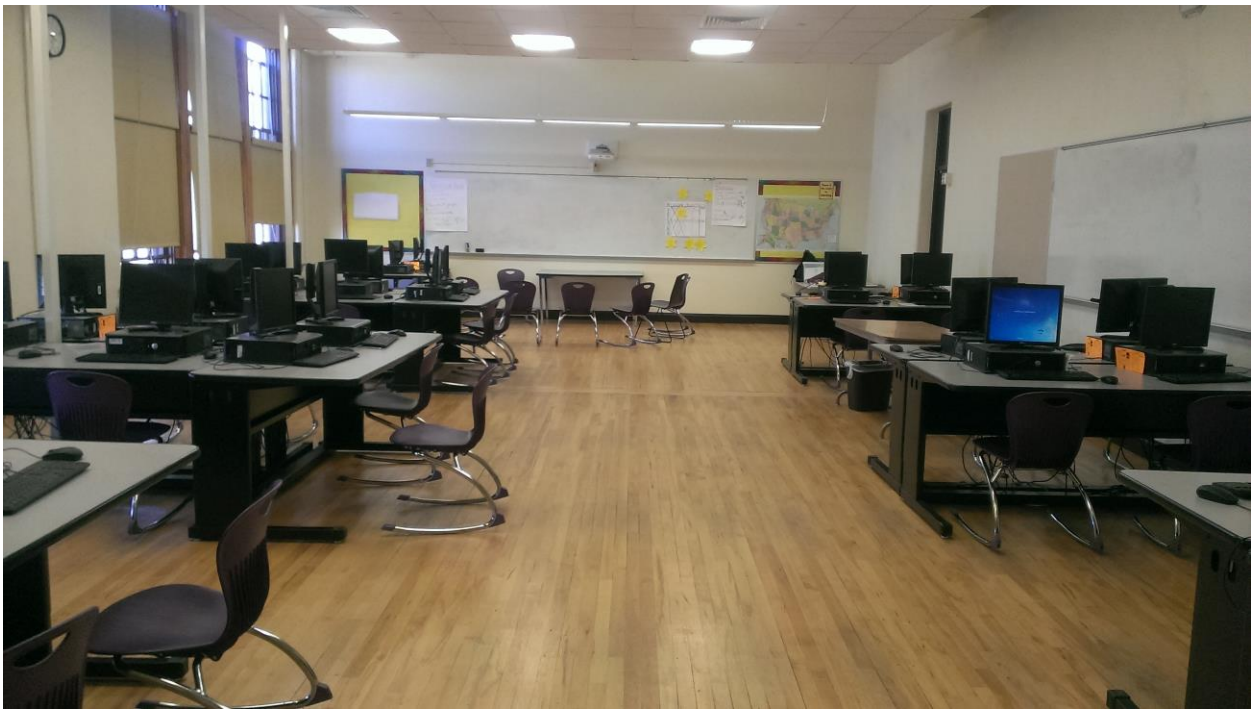


Figure 56. The Technovation Classroom

During the afterschool session, we would begin the day's activities by sitting together in a circle at the front of the room. This is where the day's goals and objectives would be discussed, in addition to performing "ice-breaker" games and encouraging the girls to collaborate through fun

and play. Afterwards, the girls would break into their teams and work at the computer "pods" (computers are organized in groups throughout the classroom).

After the first round of qualitative analysis from web analytics and taking observational notes, one of the emergent themes was that the website was not "situated" properly into how the program was actually being conducted on a daily basis. For example, the classroom itself offered for more important collaboration space than the website. Therefore, the gamification website did not support individualized student goals and learning processes because much of the learning was happening in the physical space than the digital space. Therefore, the next gamification system was constructed of paper, which explicitly stated program goals and objectives, and was physically hung in the room where the students met twice each week. Figure 57 depicts the second gamification system in use.

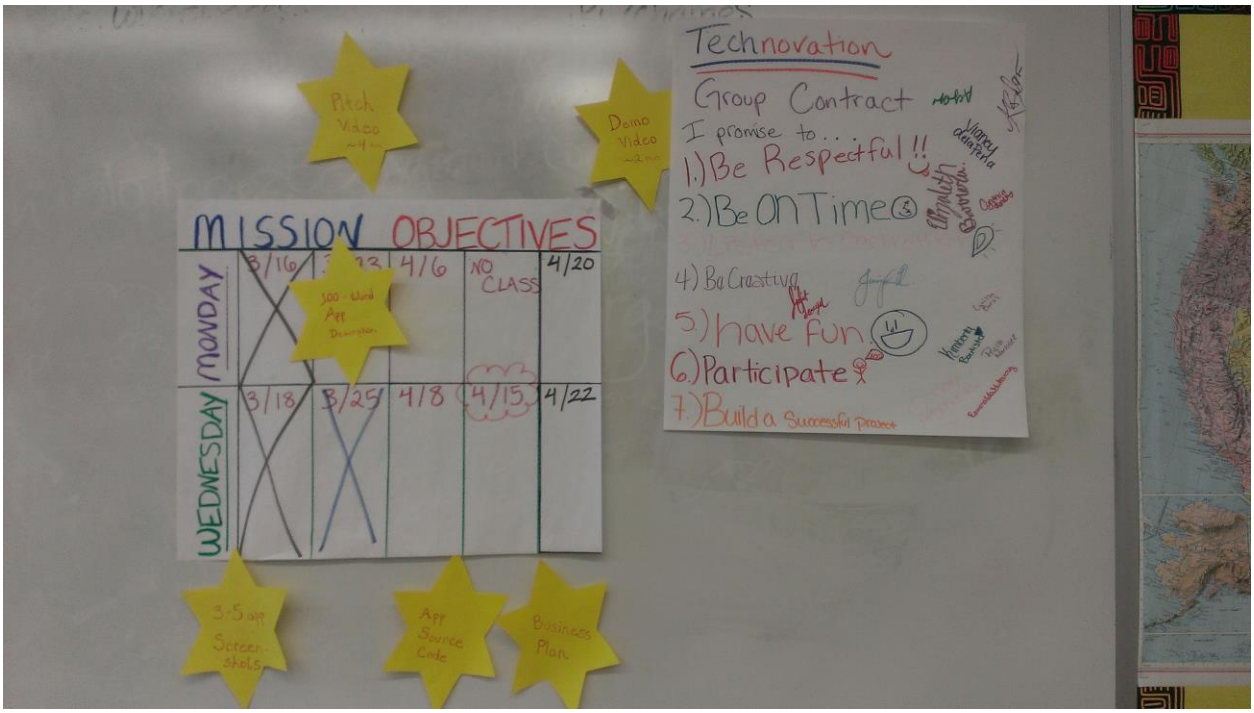


Figure 57. Photograph of the 2nd Gamification System

During the second focus group session, one of the students observed that, "I think we should have this from the beginning." The obvious conclusion is that the gamification has to be situated in a context that is useful and makes sense.

Although the gamification website was successful for the first case study, a similar implementation was not effective for the second case study. This is likely because students in the first focus were required to conduct all of their work on the website, so they were more likely to see and use the available tools. For the second case study, the website was not used because the classroom itself was a more important learning environment than the website. In addition, the intrinsic and extrinsic goals of the students from the first and second case studies were fundamentally different. In the first case study, students were paying to receive course credit in a formal learning environment. In the second case study, students were not receiving course credit were participating in an informal learning environment, and were being paid to do so.

4. Extrinsic Motivations Just as Important as Intrinsic Motivations

The focus-group interviews suggest that the extrinsic reward—the monetary stipend—was an important reason why students participated in the program. When explicitly asked if they would have participated in the program if they were not being paid, two of the seven girls said they would not have participated. One of the students stated, "the money is forcing me be here." On the other hand, other girls described intrinsic reasons for why they were participating in the program [emphasis added]:

> *"[I chose to enroll in Technovation] to **learn new things**."*
>
> *"…to **learn about technology**."*
>
> *"…when I was a freshmen, I didn't do anything, like… a program [or a club sport], so I was like… okay, **I'll just join, whatever program**…"*
>
> *"…because I wanted to **learn about technology and coding**"*

*"…to learn **how to build an app.**"*

*"I think I still would have liked to join [without the stipend] because like… I dunno, **there's nothing else to do so might as well.**"*

These girls were intrinsically motivated to join the program because they wanted to learn more about technology, because they wanted to participate in an afterschool program, and perhaps simply because they wanted to learn new things. However, all of the girls asserted that the stipend was what helped them come back and continue to participate in the program. Where some of the girls were more extrinsically motivated than others, overall, all of the participants demonstrated both intrinsic and extrinsic reasons for participating. Even the student who stated that "the money is forcing me to be here" discussed at length how she wanted to learn how to build an app and understand how the technology worked. However, all of the girls indicated that they were grateful for the stipend and that it helped them continue their participation in the program.

4.1. More "Work" Than "Play"

Several girls wanted the program to be harder. When specifically asked if they wanted the Technovation program to be more work or more fun, four of the seven girls said that Technovation should be "more work [oriented]" or be offered as a "class in school"—this would have helped motivate them to work harder on the curriculum tasks. Two other girls said the program should "balance work and fun," because if it became too work-oriented, it "will feel too much like school" and that having fun while working was a priority. However, some students stated that they would have achieved more and been more productive if was structured more like a class [emphasis added]:

*"I think **if it was a class I would have done better**…because of…grades…trying to pass the class."*

*"I think that having that pressure of **passing the class is what drives you to do good**. And sometimes, you learn. Because you have to learn stuff to get a good grade, right?"*

*"Sometimes **pressure allows us to get work done**…"*

*"I would have liked **doing it during school**."*

*"… I think pressure [can be good] because **I think we get off-task frequently** and we still have a lot of work to do."*

*"…**if it would have been a class**, there would have been more… a different schedule. Like, there would have been a specific set of time that we had to spend on things, and they would have to get done. **It would be for a grade**, and **people would be more like…pushed to get that grade**."*

One of the girls said that the program was not really worth her time because she had other obligations after school, like studying for AP exams; however, the same student observed that [emphasis added]:

*"I think it would be a good idea… [to] **actually get credit for doing [the program]… then it would be worth my time.**"*

On the other hand, some girls stated that if the program had been a formal class, their participation would not have increased [emphasis added]:

*"Yeah because for a class I think I **would have been pressured** and stuff and **I don't like that.**"*

*"I think I **get more out if right now**…for me, like, in class, I get like, more off-task on it too, like **I get bored because I know it's pressure**…"*

One student noted that, if a technology program was offered in school, it would have to be offered as a core class [emphasis added]:

*"If it was an elective, **an elective are something you can choose**. If you didn't want to be in that class, then you can just get out of the elective and get another elective… if it's like a core class, it would be kinda like… uuugh, why do I even HAVE to take this?" [student emphasis]*

5. The Importance of Girls-Only

The girls were unanimous that it was important to have an all-girls environment for them to participate in the program. This offered them a safe environment to explore, work productively in teams, and be creative without fear of worrying about peer-pressure. The comments below exemplify this issue [emphasis added]:

> *"I like working with others…with girls, cuz…with guys, with like,* ***the pressure of****…****them thinking we're dumb or something****…****with girls, we work together****."*

> *"…sometimes, guys wouldn't put in much effort as a girl would…* ***sometimes they think they're too cool****… they're like, 'I don't need to do this'… or sometimes they're like, 'oh* ***I'm too smart, you don't know what you're doing…****'"*

One student mentioned that working with others, sharing ideas to build an app, and working together on the project was a positive experience for her.

VI. DISCUSSION

In summary, findings from the pre-test and post-test surveys indicate that the girls experienced positive growth on "career skills." However, this case study demonstrates some of the problems and institutional challenges associated with offering computer science courses in high school curriculum. Students that were more extrinsically motivated wanted to receive course credit. They also indicated that students were more likely to work hard to get a good grade. Despite these assertions, two of the girls said they were "glad [we] did [Technovation]" because "otherwise I would just sit around at home." The most positive response from one of the students was that "[she] loved doing the app development, it helped make me think and understand how computers work." The girls also responded positively to working in teams, in that "it was cool to see everyone working together to get stuff done, like the [pitch] video."

1. Implementation Challenges

One of the challenges in facilitating CS curriculum in K-12 education is the apparent lack of education materials and support for teachers. For example, the non-profit community center stated that they used the Technovation platform explicitly because it had an established curriculum that would make it easier for teachers to follow and implement into a classroom or afterschool program. Therefore, scaffolding the teachers themselves is considered an important priority for the administration. As the teacher and facilitator of the program, it was useful to have those materials already established, which made it easy to implement in the curriculum. However, these materials did not align with the goals or expectations of the students. Focusing too much on the teacher-support may lead to losing sight of the goals of the students.

1.1. CSTA Curriculum Standards Are Mixed, Not "Leveled"

This project attempted to implement Level 2 of the CSTA Standards into the curriculum. However, levels of experience are not clear-cut or well defined in real teaching scenarios. For example, some students in the program had previous experience using App Inventor and programming, therefore, "level two" was the appropriate skill level for them to engage in. However, most of the participants had no previous exposure to CS or programming concepts, thus an elementary skill-level or "level one" would have been more appropriate for them. Guidance for teachers facing similar mixed experience in classes is likely to be an on-going need.

2. Gamification & Situated Design

The gamification intervention from the first case study was not successful for Technovation. This failure reinforces the need for the design of the gamification to align with the intrinsic and extrinsic motivations of the students. Second, this failure demonstrates that the design and implementation of the gamification must be appropriately embedded into the "situated

context" of the learning. The first case study implemented gamification in a for-credit course, in Case Study Two, the informal learning environment resulted in different student priorities. This outcome further emphasizes that the design of the gamification must be situationally aligned with the goals of the students and the curriculum.

The results of Case Study Two did highlight the importance of extrinsic motivation. The girls were explicit that the extrinsic rewards associated with the program helped them participate in and graduate from the program. This is especially important since there were few other extrinsic incentives for them to participate. The focus groups illustrated that having purely intrinsic motivations is challenging. Students often got off-ask, side-tracked, and did not accomplish as much as they hoped. Although they appeared to appreciate a "playful-driven" program that used games and "having fun" with friends, this also hurt work productivity. I conclude that the learning outcomes must align with the intrinsic and extrinsic goals of the students. If a program is to be more "fun-oriented," rigorous deadlines and grading rubrics, as was the case with the Technovation program, are likely to be misplaced. On the other hand, if students are expected to work productively, then the program should help motivate them both intrinsically and extrinsically, such as receiving an elective credit or a stipend for participating.

IX. FUTURE WORK

Possible follow-up work for this case study would include implementing a mobile app design program that explicitly focused on the designing, prototyping, and building of mobile apps, instead of using Technovation as the curriculum model. The girls repeatedly emphasized that they wanted to focus on app design from the beginning and would have liked to learn more about this process throughout the program.

Such a course could be designed as a mobile app design program offered as an elective course for high school credit and as an afterschool program. Results from the two implementations could be compared to determine which of the two were more successful in terms of student motivation and learning outcomes. Offering an app design program for elective credit may also help integrate CS curriculum into high school education. Students seem to want the opportunity to learn computer science at school. However, the CS courses in high schools usually depend upon the availability of school and district resources, including funding, curriculum, and teachers. Therefore, afterschool CS programs may be a viable option for some schools.

CHAPTER VII
CASE STUDY THREE: CODERDOJO

The third case study investigates whether gamification impacts children's motivation to participate in and learn computational thinking practices. The study sought to evaluate the effectiveness of gamification as a scaffolding tool for children over a four-day workshop series at an urban public library. The gamification intervention is structurally different from the previous two case studies. Participants are much younger, requiring different pedagogical and gamification design considerations. However, the theoretical underpinnings of the gamification design and the broad learning outcomes, are consistent with the previous studies. All attempt to incorporate self-directed learning, constructionism, and connectivism and the gamification intervention was developed in concert with the learning context.

I. RESEARCH SITE

1. Site Selection

This study came about as a result of the collaboration of three entities. Modular Robotics is a Boulder-based manufacturer for Cubelets, robotic blocks constructed by magnetically linking blocks with one another. CoderDojo is a global network of free computer programming clubs for young people. Hack4Colorado is a volunteer-based group in Colorado that organizes hackathons to promote creative engineering solutions for communities. Collaboration with Modular Robotics, CoderDojo Denver, and Hack4Colorado resulted in the creation of a program at the Denver Public Library (DPL). The DPL digital media lab and "makerspace" is known as the ideaLAB. The ideaLAB is open and free to the public. People of all ages can make videos, games, music, art, crafts, and more. The coordination between these partners resulted in the offering of a "Cubelets

CoderDojo" workshop event, hosted by the ideaLAB in the DPL. Children aged seven to eleven were the target age group, with the intent to create a subsequent workshop later in the year for teenagers aged twelve to fifteen.

2. Duration of Study

This case study took place over the course of a four-day workshop during the first Saturday and third Sunday of the month between March and April, 2015. Each workshop session was two-hours long and hosted by the ideaLAB in the Denver Public Library. Data-collection for the study began on the first day of the workshop series and concluded on the fourth day in April. The ideaLAB administered parent-child registration for the workshop. CoderDojo Denver and Hack4Colorado marketed the event through their network of community partners and volunteers. The Education Director of Modular Robotics was the lead instructor/facilitator of the workshop. Several other adult mentors and volunteers were recruited by CoderDojo Denver, Modular Robotics, and the ideaLAB to help facilitate workshop activities. I acted solely as a researcher (rather than the teacher-researcher) for this case study.

3. Description of Student Population

Participants in this study were seven- to eleven-year-old children. Children registering for the event were invited to participate in the study (through their parents). I discussed the research procedures with parents through email prior to the start of the workshop, and also upon the parent and child's arrival to the event. The children of parents who did not sign informed consent forms were not included in the study.

Fifty-nine children attend at least one session of the workshop series; thirty-six of these participants were considered active participants in the study when parents or guardians signed and returned the consent forms. Male and female children were represented, and six girls (16%)

attended at least one of the four sessions. The racial and ethnic distribution was predominately White but other non-majority groups, including Latino, African American, and Asian children, were also represented in the study.

II. RESEARCH DESIGN

1. Curriculum Structure

The workshop series offers a hands-on opportunity for students to play, hypothesize, and test the fundamentals of computer science. The curriculum uses Cubelets to engage students in exploring the computational thinking practices and content strands of the CSTA Standards. During each day of the workshop, students participated in inquiry-based and constructionist-based activities designed to help them expand their knowledge of computer science and engineering, while connecting new concepts and vocabulary to activities and ideas into their everyday life. They practice and conceptualize these computational thinking practices by building Cubelet robots as solutions to questions framed by the curriculum.

2. Modular Robotics: Cubelets

Cubelets are modular robot components organized into three categories—sense, think, and act[42]. The Black cubes "sense" an input, such as the amount of light in the room, the distance to an edge or wall, etc. Clear "action" cubes react to senses and produce an output, such as a flashlight, sound, wheel movement, order of magnitude, etc. Colored "think" blocks, such as the green passive cube or red inverter cube, change the input/output flow. Gray "battery" cubes provide power.

[42] The *sense-act-think paradigm* is one of the operational definitions of a robot and is often used as a broad roadmap for robotics research (Siegel, 2003).

Figure 58. The Cubelets Six Kit

Every Cubelet has a microcontroller, which employ a distributed programming system. A Sense block produces a number, which flows into and is interpreted by an Action cube. For example, the Brightness Sense block produces a high integer value when it senses a lot of light, and sends that integer to the Flashlight Action block to produce a response. There are no CPU blocks, variables, functions, or other use of procedural logic. Instead, the Cubelets robot itself is the program—the way blocks are put together determines the way integers flow from Sense to Action blocks, thus determining the robot's behavior. Independently, each Cubelet serves a specific function; when you put them together, they do something. This lays the foundation for the curriculum structure of the workshop.

The CoderDojo event is designed as a workshop series, so the curriculum builds upon the themes and topics from the previous days. *Table 29* illustrates the topics and skills associated with to each day of the workshop. Each high-level curriculum task corresponds to a CSTA Standard, as shown in the Table 27.

Workshop Topics & Skills	CSTA Strands	Schedule
• Understand robots are devices that sense-think-act • Play, manipulate, and understand a sense-act, input-output relationship • Cubelets can be "reprogrammed" by re-ordering and re-orienting • Learn about *prototypes*, *3D models*, and *hypotheses*	Computers & Communications Devices Computational Thinking Computing Practice & Programming Community, Global & Ethical Impacts	Day 1 (March 7th)
• Learn about what an *engineer* is and about *design thinking* • Demonstrate problem-solving with constraints and criteria • Understand conditionals (if-then statements)	Community, Global & Ethical Impacts Computational Thinking Computing Practice & Programming	Day 2 (March 22nd)
• Prototype robots that meet specific constraints, criteria • Demonstrate *prototypes*, *3D models*, *hypothesis*, and *design thinking* skills	Collaboration Computational Thinking Computing Practice & Programming	Day 3 (April 4th)
• Work in a team to design, prototype, and build a robot that meets specific constraints, criteria, conditionals (final maze activity)	Collaboration Computational Thinking Computing Practice & Programming	Day 4 (April 19th)

Table 26. CoderDojo Curriculum Goals & Associating Computational Thinking Practices

Children demonstrate computational thinking practices and explore other CSTA strands by actively designing, prototyping, and building Cubelets robots. The curriculum emphasizes specific learning outcomes; I examined how Cubelets enabled this understanding. Like the other case studies, the learning objectives incorporated gamification in an effort to promote student

motivation and learning. The role of gamification and role of in facilitating the learning objectives were both examined.

3. Learning Objectives

The primary goal of the Cubelets CoderDojo is to help children learn computational thinking practices as a skill to explore scientific challenges and create solutions to those challenges. Building and modeling with Cubelets facilitates these computational thinking practices. The goal of this research study was to understand how gamification might support children's motivation while undertaking these endeavors.

The learning objectives for the CoderDojo workshop were developed by the Modular Robotics educational director, based upon her experience with Cubelets in classroom learning environments. Similar to the second case study, I gamified an existing curriculum, rather than developing a curriculum from scratch. During our pre-event meetings, the workshop facilitator provided me with a list of the "essential understandings" that students were expected to learn and demonstrate by building/playing with Cubelets. I treated these essential understandings as the primary learning outcomes for the workshop. *Table 28* lists these core learning objectives, which were used directly as a part of the pre-test and post-test assessment (details on these assessments are discussed later in this chapter). By the end of the workshop series, students were expected to be able to demonstrate an understanding of the concepts listed.

Measurable Learning Outcomes
Understand how **Robots** work
Rapid **Prototyping**
Build a **3D Model**
Define and Test **Hypotheses**
Understand what an **Engineer** does
Facilitate **Design Thinking**

Table 27. Workshop Learning Objectives & Associating Big Ideas

Data and information facilitate the creation of knowledge, without which, robots would not function. Prototyping is an algorithmic tool to develop and express solutions to computational problems. The design and development of 3D models enable problem solving, human expression, and the creation of knowledge. Forming and testing hypotheses is a process of abstraction that reduces information to facilitate focus on relevant concepts. Understanding what an engineer does creates opportunity for students to understand the role of engineers and their impact on the world. And finally, creativity is a prerequisite for effective design thinking (NRC, 2003; Brown, 2008; Ferrara, 2012).

4. Robot Investigator Workbook

To help structure the curriculum and facilitate the intended learning outcomes, I designed a "Robot Investigator Mission Log" as the structural and content gamification mechanism for the CoderDojo workshop. The Robot Investigator Mission Log (see *Figure 59*) is a workbook that functions as a scaffolding guide for students.

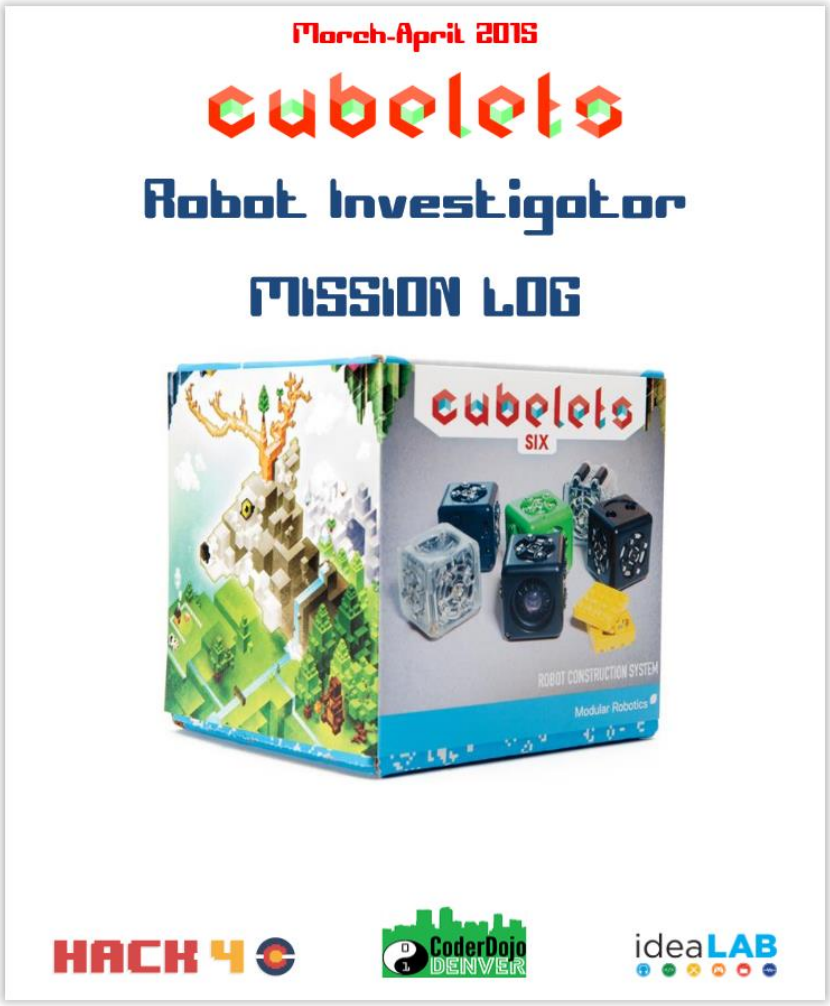


Figure 59. The Robot Investigator Mission Log

The workbook functions as a scaffolding tool by partitioning curriculum into daily objectives and modular lessons. Each day of the workshop (see *Figure 60*) is labeled as "Day 1 Mission," "Day 2 Mission," and so on.

Figure 60. Learning Objectives via the Robot Investigator Workbook

The Missions represent the broad curriculum tasks of that day. The "mission target" foreshadows the themes of the day's lessons. Each Mission is subdivided into smaller "Mission Objectives," which are modular lessons or tasks that students follow throughout the workshop activities.

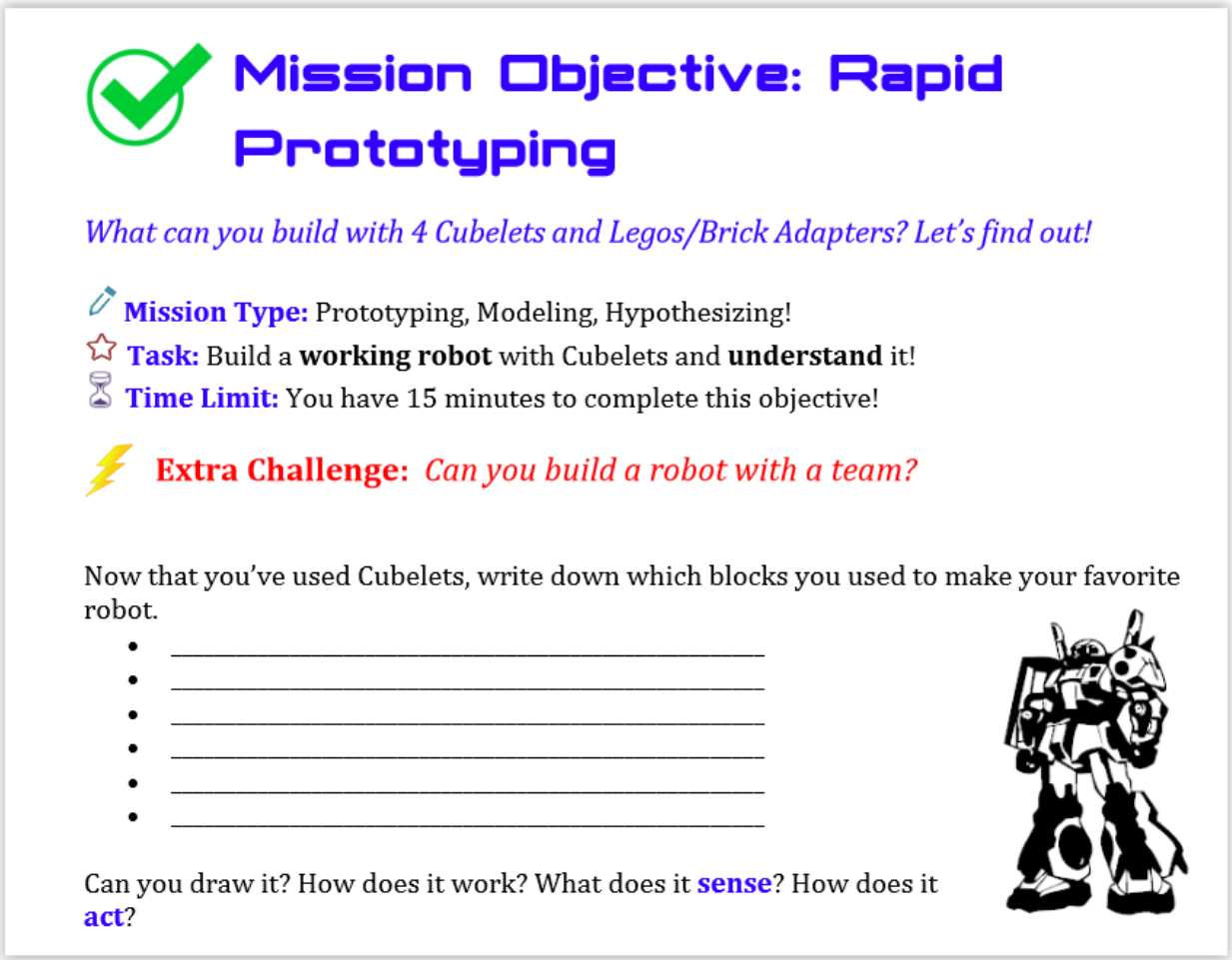


Figure 61. Mission Objective for Day 1

Each mission objective begins with a statement that summarizes the "Task" for that exercise. The "Mission Type" indicates the learning objective associated with this lesson. The objective also implements a "Time Limit" or a game constraint[43] that may encourage user-engagement while also helping to keep the curriculum on-task and on-time[44].

[43] Time limits, such as a countdown clock, are an effective structural game element that is frequently used in video games, game shows, sports, etc. It is most commonly used to create a sense of "pressure" for the player, which can enhance performance and motivation to complete a certain task (Kapp, Blair & Mesch, 2014).

[44] Each day of the workshop is two-hours long. Time had to be managed wisely in order to get through the intended curriculum content of that day. Therefore, the time limit was designed to help both the student and the facilitator stay on track in a timely manner.

The Robot Investigator workbook helps scaffold the intended learning objectives for the student. The workbook is designed to work in tandem with building Cubelets robots. Each mission objective required the student to perform some kind of task that directly relates to the robot they were modeling or prototyping. As shown in *Figure 62*, students are tasked to create robots according to specific criteria and constraints. The workbook provided instructions and details about the objective. For example, students were given a task to build robots that drive straight, move sideways like a crab, drive in circles, drive away, and drive towards you. The workbook then prompted the student with questions that could help in their design-thinking process. In this case, the Cubelets did need to be arranged in a particular sequence in order to create a robot that drives away, drives toward you, and drives in circles. In this fashion, the Robot Investigator workbook supported the constructionist process.

Mission Objective: Complete the Drive-Bot Challenge

*Engineers are people who **solve problems** by **making** or building something. Can you go through the **design process** while meeting the **criteria** and **constraints**?*

Mission Type: Prototyping, Modeling, Hypothesizing, Design Thinking!
Task: Build as many robots as you can that do **different things**!
Time Limit: You have 35 minutes to complete this objective!

Extra Challenge: *Can you complete **all** of the following tasks?*

Can you make a robot that will:

1. Drive straight
2. Move sideways like a crab
3. Drive in circles
4. Drive away, like being controlled by the Force
5. Drive towards you

Drive straight

Which Cubelets will you need for your robot? _____
What job could it do? _____
Do these robot pieces need to be in a special order? _____
What is that order? _____

Figure 62. Daily Missions & Objectives are Color-Coded

In general, the Robot Investigator workbook supports students and the facilitator throughout the CoderDojo curriculum. Students can use the workbooks as a space to brainstorm ideas, draw sketches of their robot prototypes, and answer questions to reflect on their learning, among other creative exercises.

5. Badges & Extrinsic Rewards

An important consideration of this work is whether extrinsic rewards can positively promote student motivation when learning. Prior research has shown that extrinsic rewards can be harmful towards personal motivation (Deci & Ryan, 1991; Ryan & Deci, 2000), but other work has shown that extrinsic rewards can help sustain motivation (Nicholson, 2012). Extrinsic rewards were implemented directly into the workshop curriculum to help explore this tension.

On the first day of the workshop, the Cubelets facilitator announced that the children had the possibility of winning rewards by earning badges for the work they completed in the workshop. Students could earn badges in two ways: first, by completing the designated task as specified in the workbook, and second, by completing extra sets of challenges. Badge-earning opportunities were always highlighted in red text, as shown in Figure 63.

Extra Challenge: *What do you think the robot is **sensing**? How can you **drive** the robot across the table? Write your solution below.*

_____

_____

If your robot has 2 senses (like the picture), what do you think is happening?

_____

_____

Is it easier or harder to control? _____

Can you make it drive faster or slower? _____

Are there ways to build your robot to make it easier? _____

With both the **Distance** and the **Brightness** blocks, the robot **drives slowly** without anything close to it because it's getting **input** from the _____

It **drives fastest** when something is **close** to the **Distance** sense because then

_____

What does the **secret block** do? _____

Complete this page and show it to a mentor to earn a badge!

Figure 63. Red Text Signifies Opportunities to Earn Badges

Students were awarded a badge (i.e., a Lego sticker or Cubelets stamp) for every listed challenge they completed in the Robot Investigator workbook. The "Extra Challenge," which was always highlighted in red text and marked by a lightning bolt, was used to support self-directed autonomy and extrinsic motivation. The children were not required to complete these objectives, but they were encouraged to do so that they could potentially win prizes (and perhaps learn along the way).

Over the span of the four workshop sessions, students had opportunities to complete learning objectives and receive badges for their efforts. The Cubelets facilitator explained that, if the children receive a target number of badges, their name would be entered into a "raffle" so that they could win prizes. Children were not told what these prizes were. Previous work in motivation

has shown that when participants know that they will receive a specific prize or reward for completing some kind of task (before they have done so), this knowledge is likely to adversely affect motivation to complete that task (Kohn, 1999; Pink, 2009). However, *contingent rewards* (i.e., the participant has knowledge of the reward but not its nature) have been found to be effective (Pink, 2009). In this case, students were told on the first day of the workshop that they could potentially win prizes if they received enough badges, but they were not told what those prizes were. The "grand prize" for the workshop has a Cubelets Six Kit[45]; other prizes included Modular Robotics t-shirts, stickers, Cubelets beverage glasses, and donated or sponsored activities.

When students earned a badge for completing an extra challenge, they display that badge on the first page of their Robot Investigator workbook (see *Figure 63*). Badges were put on the first page of the workbook in order to promote the earning of badges and to make counting badges and administering the prizes at the end of the workshop easier for mentors and volunteers.

[45] These kits sell for $159.95 on the Modular Robotics website.

Figure 64. The "Badges Plaque" to Display Earned Badges

Mission objectives in the workbook are color-coordinated. The badges plaque uses the same color-coordination scheme to organize the badges according to the day they were earned or received. In summary, the children were encouraged to engage and participate in the gamification (earn badges and complete workbook activities), but they were not required to do so. The Robot Investigator Mission Log was designed to support the core affordances of Self-Determination Theory.

III. EVALUATION

1. Theoretical Framework

Like the first and second case studies, the curriculum incorporates self-directed autonomy, constructionist mastery, and connectivist purpose into its pedagogy, as discussed below.

1.1. Self-Directed Autonomy

Although the CoderDojo workshop used a structured curriculum focused on specific learning outcomes, many of these tasks were self-directed. For example, Day 3 and Day 4 only focus on one mission objective, whereas Day 1 includes five mission objectives and Day 2 has four mission objectives. The decreasing number of objectives was a way of supporting self-directed autonomy by scaffolding the more structured tasks, leading to more open-ended, autonomously-driven tasks. This also supports autonomy—Cubelets are somewhat complex to understand at first, but once someone "gets the hang of things," he/she is more likely to explore and prototype more alternative implementations. As children gain competency in using and building with Cubelets, the structured tasks yielded to more open-ended "free time" exploration.

1.2. Constructing Mastery

It requires constructionist learning and competency in computational thinking practices to build functional Cubelets robots. Cubelets are explicitly designed to enable constructionist learning and implicitly designed to support expanding competency of computational thinking practices. As the workshop series progresses, each day builds upon the lessons and activities from the previous day, as students demonstrate competency and mastery of the material. When children (parents) registered for the CoderDojo workshop at the ideaLAB, they were not informed that the workshop would use Cubelets as the main learning tool. By Day 4 of the workshop, groups of

children worked in teams to prototype, test, and deploy robots that could successfully autonomously maneuver through a maze without human interaction.

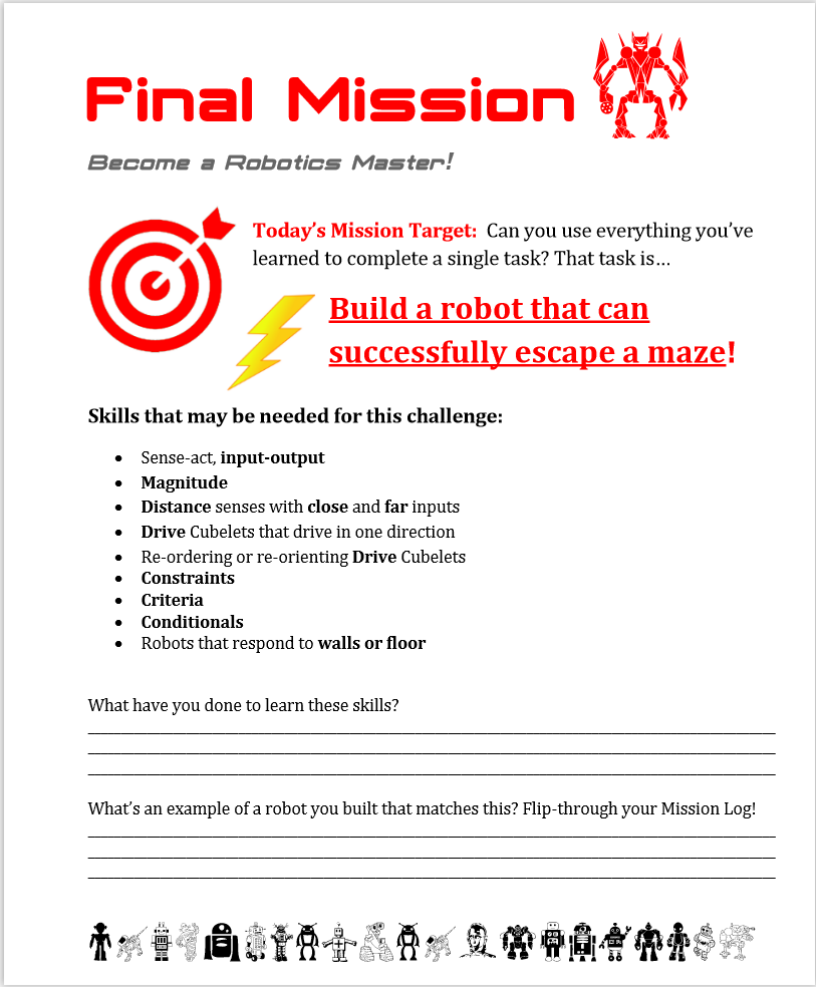


Figure 65. Day 4: Build a Robot to Escape the Maze

On the 4th day, the children were tasked to "become a robotics master!" by incorporating all of the knowledge they learned from the previous three days into one single task. This was considered the "Boss Fight" of the entire workshop, and if students succeeded, they "beat the level" and obtain mastery of the material.

1.3. Connectivist Purpose

Students were required to work in groups for the entire duration of the CoderDojo workshop. This curriculum decision was made for several reasons. First, providing a Cubelets Six Kit or Twenty Kit for each registered child (approximately thirty to thirty-five children per session) was not feasible. The ideaLAB space could not support number of individuals working alone (not enough tables, chairs, etc.), nor was it feasible for Modular Robotics to provide so many materials for a two-hour lesson. In addition, the workshop promoted student group-work because engineers and computer scientists often work in teams; children were likely to come up with more solutions and more diverse and creative ways to solve problems; and group-work promoted community and connectivist purpose.

2. Procedures

This study used a mixed-methodology approach. Quantitative and qualitative data analysis began after the workshops' conclusion in April. Primary data for this research was collected in the form of pre/post-test survey and IMI questionnaires, video-recorded ethnographic observations, and content analysis of the Robot Investigator workbooks.

2.1. Survey Instruments

Surveys were administered on each day of the workshop. On Day 1 and Day 4, I administered pre-test and post-test surveys. Table 29 shows the baseline assessment used to measure learning outcomes for the children. Appendix K contains the full text of the pre-test and post-test surveys used for these assessments.

Q2	I know what a *robot* does. (1=Totally Disagree, 5=Totally Agree)
Q4	I know what a *prototype* does. (1=Totally Disagree, 5=Totally Agree)
Q6	I know what an *engineer* does. (1=Totally Disagree, 5=Totally Agree)
Q8	I know what a *hypothesis* is. (1=Totally Disagree, 5=Totally Agree)
Q10	I know what a *3D model* is. (1=Totally Disagree, 5=Totally Agree)
Q12	I know what *design thinking* is. (1=Totally Disagree, 5=Totally Agree)

Table 28. Itemized Learning Outcomes for Pre- and Post-Tests

In order to make the surveying instrumentation more appropriate for young children, a "smileyometer" was used to indicate levels of agreement on the Likert scale as shown in Figure 65. This procedure has been used in previous work to measure fun and enjoyment when surveying children (Sluis, Dijik, & Perloy, 2012).

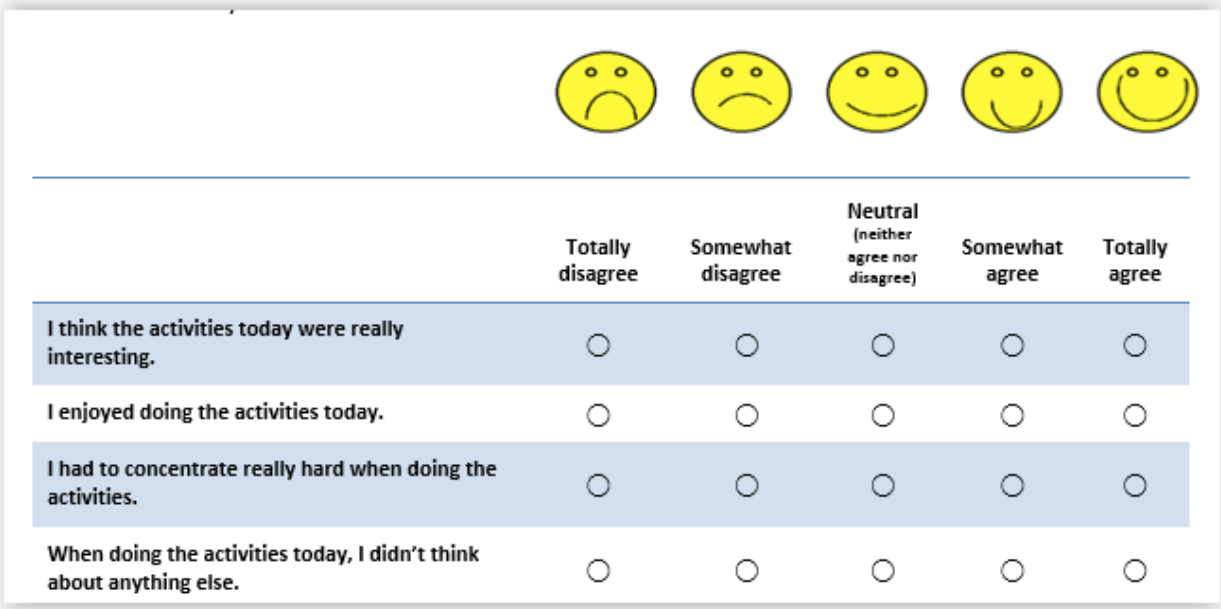


Figure 66. Smileyometer (Sluis, Dijik, & Perloy, 2012) as a Visual Aid for Likert Scaling

The Experience Sampling Method (ESM) surveys were administered at the end of Day 2 and Day 3 in an effort to measure the levels of interest/enjoyment, perceived competency, and

relatedness immediately after the activities were completed. This testing was implemented in order to assess whether Cubelets and the gamification increased student enjoyment, mastery, and connectivism. I use the analysis procedures described in the other two case studies to analyze and interpret the results of this quantitative data.

2.2. Ethnographic Video-Recordings & Content Analysis

Qualitative-data was collected by video-recording the children while they actively participated in the learning activities. I used content analysis procedures to open-code and axial-code observed inferences, especially when the children were engaging with the constructionist practices and the gamification mechanisms. I also transcribed the audio as another potential indicator of motivation and learning outcomes fostered by the learning environment.

2.3. Robot Investigator Workbook

The gamification prototype of this research is the Robot Investigator Mission Log. All information that the children provided in their workbooks, including completing activity tasks, sketches, the use and implementation of badges, etc. was examined.

IV. FINDINGS

Our results suggest that both student motivation and learning outcomes were positively impacted by the introduction of game elements. In addition, Cubelets and the Robot Investigator workbook had a statistically significant impact on students' learning outcomes for introductory computer science topics. Cubelets as an artifact also showed high levels of engagement that facilitate self-directed learning, constructionism, and connectivist learning.

A total of fifty-nine children participated in at least one day of the workshop series; five children attended all four days of the workshop. A total of 73 "badges" (Tetris stickers) were earned and collected by the children, with an average of 9 stickers per child participating in the workshop.

1. Workshop Facilitates Successful Learning Outcomes

1.1. Successful Learning Outcomes

Of all the children who participated in the workshop, thirteen children completed both the pre-test and post-test survey for the workshop. Participants that only completed one of the pre- or post-tests were not considered as part of the analysis. Three of the thirteen children (23%) who completed both pre-test and post-test were girls.

The pre- and post-test questions that reflected learning outcomes were Q2 (robots), Q4 (prototypes), Q6 (engineer), Q8 (hypothesis), Q10 (3D models), and Q10 (design thinking) (see Table 30 and Appendix K). A mean score was calculated for each question in the pre- and post-test items. Table 30 shows that, on average, participants' knowledge increased significantly for items Q2, Q4, Q10, and Q12. Items Q6 (engineers) and Q8 (hypothesis) did not have significant gains—a high ceiling effect could explain this result. For example, one could infer that "hypotheses" are taught at approximately this age and grade level in school, thus significant gains may not have been obtained because of this. Another possibility is that these students already knew about what engineering is or what an engineer does because of family history or parental job occupations. Regardless, learning outcome items saw significant gains between the pre- and post-test.

	Pre Q2	Post Q2	Pre Q4	Post Q4	Pre Q6	Post Q6	Pre Q8	Post Q8	Pre Q10	Post Q10	Pre Q12	Post Q12
Mean Score	3.23	4.15	2.31	4.23	4.23	4.46	3.85	4.00	3.69	4.46	2.00	3.46
Difference	0.92		1.92		0.23		0.15		0.77		1.46	
(Student's t-Test) ***p* value**	0.06		0.00058		0.43		0.34		0.08		0.02	

Table 29. Pre-Test & Post-Test *t*-Test of Mean Scores where $p \leq 0.05$

A Student *t*-Test confirmed that the change on items for Q2, Q4, Q8, Q10 and Q12 was statistically significant at a $p \leq 0.10$ level, whereas items Q4 and Q12 are statistically significant at a $p \leq 0.05$ level. These results indicate that the CoderDojo workshop was successful in reaching the intended outcomes for children learning about computational thinking concepts.

2. Cubelets Facilitate Constructionism

Children demonstrated their understanding of the learning objectives by actively building and constructing robots. Most of the children would begin their robot design processes by first drawing pictures of models and prototypes in the Robot Investigator workbook. Next, they would test their models by building and constructing their designs with Cubelets.

Figure 67. Photograph of Children Engaging in Constructionism via Cubelets

The final day of the workshop required the children to build a robot prototype as a team and then deploy that robot to navigate through a maze autonomously. Qualitative observations (i.e., facial expressions, gestures, verbal statements, etc.) reflected that students experienced immense satisfaction and joy when their robot successfully navigated the maze. Moreover, throughout this process, students demonstrated proponents of mastery—overcoming failure and a difficult challenge through continued effort. Not all of the teams successfully completed the maze challenge; nor did any team successfully get their robot through the maze on their first try. Regardless, students were generally supportive of other teams' efforts, and satisfied about their work, whether they built a successful maze-navigating robot or not. However, attitudinal issues were present; at least one child on each day showed a type of upset emotion (i.e., crying or showing anger).

3. Gamification Supports Extrinsic & Intrinsic Motivation

Findings also show that the gamification prototype was successful in promoting student engagement and the intended learning outcomes. For example, one of the girls—the grand-prize winner—stated on multiple occasions (prior to winning the prize) that "[she] got 15 stickers! I feel so accomplished…I am proud!" Another female participant showed signs of pride in completing the workbook activities—so much so that, on one day of the workshop, she was upset that "[she] didn't have more time" to complete the workbook activities. She (and her father) also requested that the Robot Investigator workbook be mailed or sent to them after the data-analysis process was complete. Eight parents/children indicated that they wanted to receive their workbooks after data-analysis process was complete. This is another indicator that the gamification structure was well received. However, there were children who simply wanted to engage in the construcionist play in the Cubelets activities; badges were not an important factor for these children. Finally, the badges—particularly the Tetris stickers—were "cool" enough that students wanted to collect them to mark their achievements in the workbook. These interactions indicate various levels of extrinsic and intrinsic motivation to engage in the activities. Most importantly, the children were continuously engaged throughout the duration of the workshop, whether they were playing with Cubelets or working in the Robot Investigator Mission Log.

V. DISCUSSION

Overall, the children were actively engaged in constructionist play using Cubelets. They adapted computational thinking skills, applied methodologies to design and test their prototypes, and successfully collaborated in teams that build a robot to successful and autonomously navigated through a maze. In addition, the Robot Investigator Workbooks appeared to be a useful learning

aid for students to conceptualize and draw their prototypes, demonstrate computational thinking concepts such as conditionals, and even present mark of pride for the work and lessons learned throughout these endeavors. There was a positive relationship between the number of awarded badges and the attendance rate of students who attended the workshop. In addition, many children demonstrated through qualitative observation that they were motivated to engage in the activities because Cubelets are fun, badges offered an incentive to pursue through tough challenges or to revisit work or learning material that wasn't completed before, which also worked as a mark of pride and accomplishment, and external rewards offered an extra incentive to attend multiple days of the work shop series and participate in the activities.

1. Issues for Internal Validity

One issue for this study is the range of cognitive ability of the participating children. For example, there were students that could not engage in some of the workbook activities because their reading level was insufficient. Older students were better able to complete the workbook activities and survey instruments.

CHAPTER VIII
CASE STUDY FOUR: ONLINE TUTORIALS

Online courses are becoming increasingly popular to teach introductory computing topics to people for free. The first case study showed that many students engage in online tutorials to pursue their learning of CS and programming. These tutorial websites include modular training programs such as *Code Academy* (2013) *Khan Academy* (2006), and *Grok Learning* (2013), educational game-based learning environments such as *Gidget* (2014) and *LightBot* (2008), as well as open-ended creative environments such as *Scratch* and *Alice* (Utting et al., 2010). Users of these tools report that they enjoy these informal resources because they allow for flexibility in how they learn (Boustedt et al., 2011), they provide a better sense of retention of the material, and that they are more motivating, engaging, and interesting than traditional classroom courses (Cross, 2006). However, few of these online resources report anything beyond the number of users that have signed up for their services. Further, if online website tutorials are going to be used as a legitimate tool for learning introductory CS and programming, more research is needed to assess evaluate what users are learning (if anything) when interacting with these resources.

This case study was conducted to follow-up on the findings in the findings of the *Virtual Worlds* project, where several students used online training tutorials to continue their pursuit of computer science. The purpose of this study is to gain insight into whether these websites facilitate useful learning outcomes for introductory computing concepts. This work considers how three instructional approaches to learning introductory programming—game-based learning, gamification, and command prompt programming—effect student motivation and learning.

I. RESEARCH SITE

1. Site Selection

The research sites of this investigation were introductory CS tutorial websites. Three websites were selected because of the different instructional approaches they incorporate. Each of these sites claim to teach introductory computing concepts and programming. The first site, *Gidget*, is a game-based learning environment that uses an interactive game experience to teach computing concepts. The figure below is a screenshot of the Gidget user interface. This is considered a "game-based learning" environment because the learning is designed to occur within the experience of playing the game.

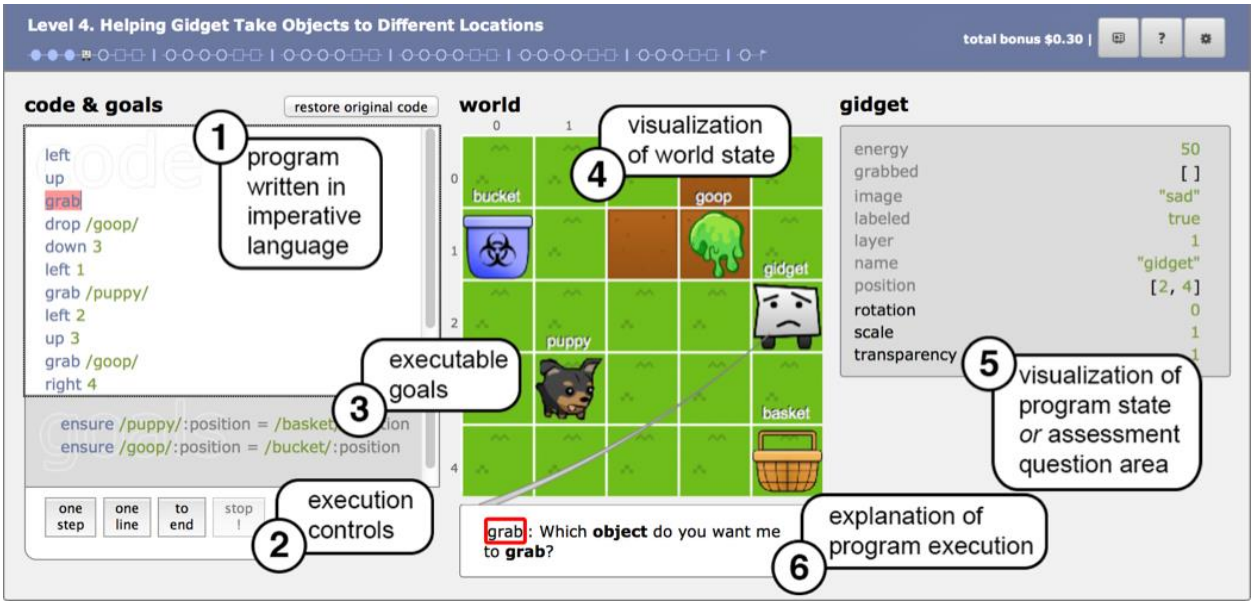


Figure 68. Screenshot of the *Gidget* (2014) Game-Based Learning Environment

Gidget is a web application similar to the "grid-world" that is used in introductory programming for students[46]. Players help a robotic character to fix its code by writing scripts to complete missions. The game was designed to appeal to both youth and adult novices by presenting debugging tasks as puzzles. The game uses an imperative Python-like programming language (Lee et al., 2014). Gidget has a total of seven modules totaling thirty-seven levels, where each module contains a set of levels focusing on a related set of programming keywords and concepts. Players can receive help from tutorials that highlight and explain different parts of the interface and the sequence of steps. The game also features an in-game reference guide, providing explanations and examples of each command in the language.

The second site used for this study was *Khan Academy*'s module on JavaScript. Khan Academy has a variety of training tutorials for various programming languages—JavaScript was chosen because it teaches introductory computer science concepts discussed later in this chapter.

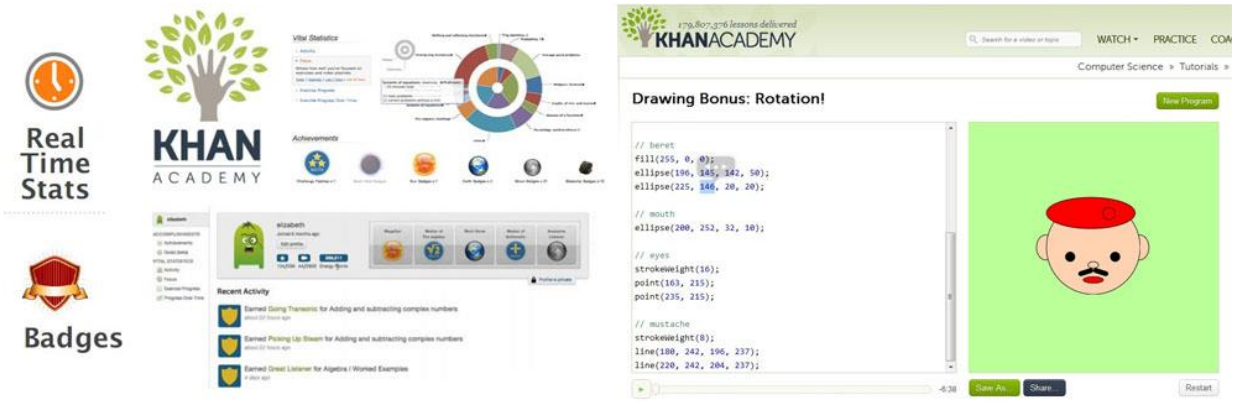


Figure 69. Screenshots of Khan Academy's Gamification (left) & Programming Editor (right)

[46] Grid World is a case study that was used as part of the AP CS A examination. As of the year 2014-2015, Grid World is no longer required as part of the exam.

Khan Academy was also chosen because it employs typical gamification techniques, such as the use of badges, achievements, levels, points, leaderboards, etc., in an effort to motivate learners. The CS modules involve users watching a video explanation of the topic followed by writing code in a programming editor and execution window. The output of the code changes immediately to reflect the new code without the need to compile the script. Users may also "drag-and-drop" features to alter values. The JavaScript lessons primarily use drawing and animation exercises.

The third website used for this study was the "Introduction to Programming 1" course in *Grok Learning*. Similar to Khan Academy, Grok Learning is an online interactive platform that offers coding courses in Python. Grok Learning focuses on educating high school students and professionals who want to increase their programming skills.

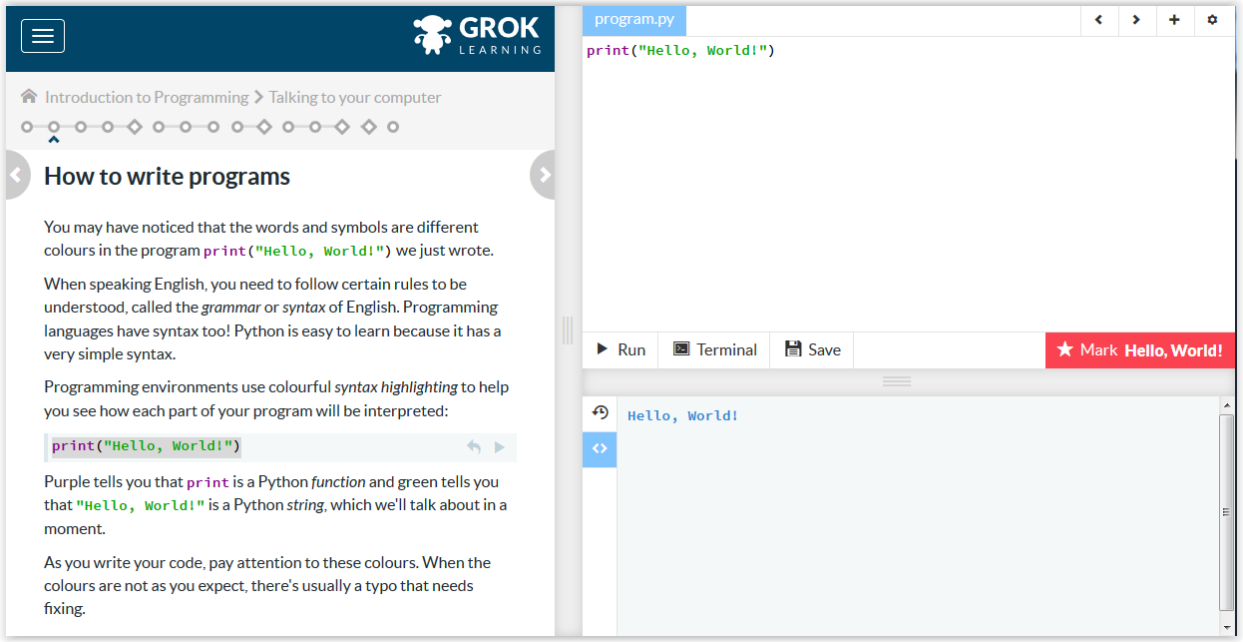


Figure 70. Screenshots of Grok Learning Programming Console

Grok Learning also provides users with step-by-step tutorials and instructions to learn and write code. However, unlike the previous two sites, Grok Learning does not incorporate any gaming

components to the platform. It was specifically chosen for this reason, in that it offers a more "traditional" approach to programming by offering users the ability to use the terminal or command prompt to code. The instrumental approaches and intended learning outcomes for each website is listed in the Table 31.

Website Name & User Features	"CS1" Pedagogical Content

• **Gidget**
  ○ Game-Based Learning
  ○ Narrative, story characters
  ○ Play structure
  ○ Python-like
  ○ "Cute"
  ○ Incremental Levels
	• Input / output
	• Syntax
	• Variables
	• Mathematical & logical operators
	• Strings
	• Conditionals
	• Functions
	• Lists and arrays
	• Classes

• **Khan Academy**
  ○ Gamification
  ○ JavaScript
  ○ Creativity focus (drawing & animation)
  ○ Immediate code editor
  ○ Video tutorials
	• Input / output
	• Syntax
	• Variables
	• Mathematical & logical operators
	• Draw & animation
	• Strings
	• Functions
	• Conditionals
	• Looping
	• Lists and arrays
	• Objects, types, inheritance

• **Grok Learning**
  ○ No-gaming
  ○ Python
  ○ Terminal, command prompt
  ○ Compiler to run code
  ○ Incremental tutorials
  ○ Partially free
	• Input / output
	• Syntax
	• Variables
	• Mathematical & logical operators
	• Draw & animation
	• Strings
	• Functions
	• Conditionals
	• Looping
	• Lists and arrays
	• Objects, types, inheritance

Table 30. List of Online Tutorial Websites and Instructional Approaches

The websites were selected because of their specific instructional approaches (game-based learning, gamification, no-gaming) and the CS1 learning outcomes associated with each website.

2. Participant Recruitment

Purposive sampling was used for this study. This is a non-probability sampling technique where the units investigated are based on the judgement of the researcher. Introductory CS students with little to no previous experience in programming were the preferred subjects for this study. Therefore, I asked two instructors and one professor at the university to recruit students into the study on my behalf.

Students were sampled from three separate introductory computing courses at the University of Colorado Boulder. These courses were CSCI-1300 "Introduction to Computer Science," ATLS-2000 "Meaning of Information Technology," and ATLS-2519 "Code." Introductory courses were selected in an effort to recruit students from a variety backgrounds, majors, etc., who potentially had some interest in learning about computer technology. ATLS-2000 does not cover any programming or explicit CS topics in the course, but CSCI-1300 and ATLS-2519 explicitly teach introductory computer science.

I left it up to the discretion of the instructors how they wanted to proceed and recruit students for the study. Two instructors used the study as an opportunity for students to receive extra credit in the course, the other offered it as an "alternative track" or supplemental material to be used in the course (i.e., not extra-credit but additional learning resources). Students were not given any other incentives to participate in the study.

3. Duration of Study

The duration of this study was approximately six weeks. Data-collection began during the second or third week of the Fall 2015 semester (depending when the study was introduced by the course instructor). Data-collection ended on October 25th, 2015. This gave students approximately eight weeks to participate in the study.

VI. RESEARCH DESIGN

This study seeks to gain insight into whether gaming techniques help motivate students to pursue online learning for CS. A learning assessment was used to test whether the selected sites were helpful in teaching foundational introductory CS.

Students first participated in pre-test survey that measured previous programming experience. A knowledge test on foundational programming concepts that are often taught in introductory CS courses was also administered. After students took the pre-survey, they were asked to engage with an online tutorial website of their choosing to learn about programming. Students were not asked to complete the website tutorials in any given amount of time, for several reasons. First, it was up to the instructor of the course how they wanted to incorporate this study into their class; one professor used the website as a semester-long initiative to supplement homework assignments, whereas the other instructors offered the study participation as a form of extra credit. Second, I wanted to preserve student autonomy. I also wanted to observe how long students would likely engage in this endeavor specified external time pressure. For example, previous work indicates that many online learners stop engaging in the free online course after a certain amount of time (Lee & Ko, 2015). Therefore, I examined how long it took for the students to complete the work—or if they ever completed the work. Highly-motivated students were likely

to be sampled for the study because of the nature of the research design (i.e., students did not receive direct benefit or compensation for participating in the study).

1. Procedures

1.1. Pre-Test & Post-Test Design

The pre-test survey (Appendix M) examined student programming experience. The independent variable in this research was the instructional approach—using a game-based learning environment, using gamification, and using a non-gaming oriented website to learn introductory programming.

1.1.1. Knowledge Test for CS1 Concepts

In order to measure how much participants leaned and what they learned, I created a "CS1 knowledge test" designed to be taken before and after the learning intervention (the chosen tutorial website). First, concepts to test were determined by comparing topics taught in introductory programming courses to the sets of concepts that were covered in the Gidget, Khan Academy, and Grok Learning activities. A total of eight concepts were chosen: (1) basics (variables, mathematical operators, relational operators, Booleans), (2) logical operators, (3) selection statements (conditionals), (4) arrays, (5) while loops, (6) for loops, (7) function parameters, and (8) function returns.

These questions were modeled after Allison Tew's dissertation work on FCS1, a programming language-independent test using pseudo-code. In her studies, Tew showed that the testing results of entry-level CS students in the classroom with their "native" programming language and in the pseudo-code were strongly correlated (Tew, 2010). I generated pseudo-code questions using examples, descriptions, and the two-page pseudo-code guide provided in her

dissertation. To minimize confounding factors in syntax design, I followed the prior work in syntax learnability, which excludes semi-colons and curly braces, indenting code blocks, upper-casing reserved words, and closing program blocks with explicit keywords (Sime, Greene, & Guest, 1976). Figure 69 is a screenshot of two questions of the knowledge assessment survey.

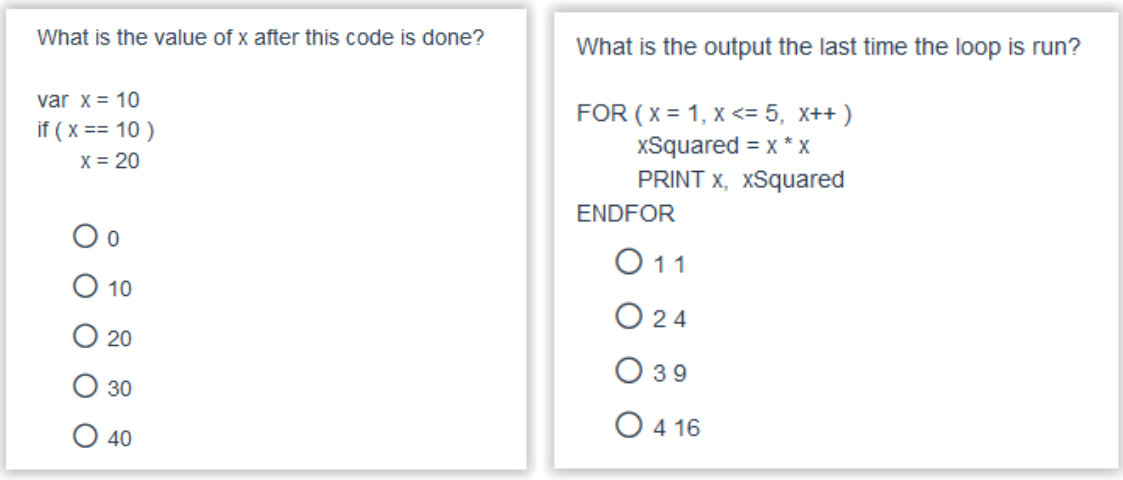


Figure 71. Screenshots of CS1 Knowledge Assessment Items

I also designed "distractors" to deliberately test for common programming misconceptions, such as pre-test items Q11 (e.g., concatenate strings) and Q15 (e.g., for loops) in the pre-test survey (Bonar & Soloway, 1985; McNamara, Jackson & Graesser, 2009). Items Q34-Q51 on the post-test were identical to the knowledge assessment questions in the pre-test in an effort to measure knowledge gained.

1.1.2. SIMS Survey

The Situational Motivation Scale (SIMS) was used to measure different constructs of intrinsic and extrinsic motivation. SIMS measures the following motivational concepts: (1) intrinsic motivation, (2) identified regulation, (3) external regulation and (4) amotivation (Guay, Vallerand, & Blanchard, 2000). *Identified regulation* refers to the act of performing an activity

because it is a means to an end. *External regulation* measures the performance of an activity for the purpose of receiving external rewards, and *amotivation* identifies activities that are neither intrinsically nor extrinsically motivating. Items Q18-Q33 asked participants about intrinsic motivation, identified regulation, external regulation, and amotivation regarding their interest in learning programming and introductory CS. For questions specifically asking about the use of the websites and motivation itemization, a 7-point Likert scale was used.

VII. EVALUATION

During the first stage of this research design, four CS learning websites were selected for students to choose to use. Students were offered to choose between Gidget, Khan Academy, Grok Learning, and Scratch. However, Scratch was dropped from the data analysis procedures because no students engaged in Scratch during the study.

A total of fifty-seven students participated in the pre-test survey. Only eleven of these students also participated in the post-test survey. Students who did not complete both the pre-test and post-test survey were excluded from the study.

VIII. FINDINGS

Nine of the eleven students (82%) that participated in both the pre-test and post-test were women. One of these students was actively enrolled in CSCI-1300 during the time of the experiment. All of the other students were enrolled in the non-programming course, ATLS-2000. Three of the eleven students indicated that they had taken a computer science course before (Q2)—one student had taken the AP CS A Java in high school, another was a student in the *Virtual Worlds*

class from the first case study, and the third student had completed the introductory course CSCI-1300.

Four students chose to use Gidget, six students used Khan Academy, and one student used Grok Learning exclusively. Two of the other students used Grok Learning (totaling three students of the sample) in addition to one of the other website interventions.

1. Game-Based Learning is Engaging

Overall, the students indicated more positive responses about the game-based learning environment than the gamification mechanics. Although the sample size is small, 50% of participants explicitly mentioned the gaming portion of Gidget as producing a positive experience for their learning. The Gidget users averaged 5 hours of activity engagement when using the website. Figure 71 summarizes these users' overall perceptions of using Gidget.

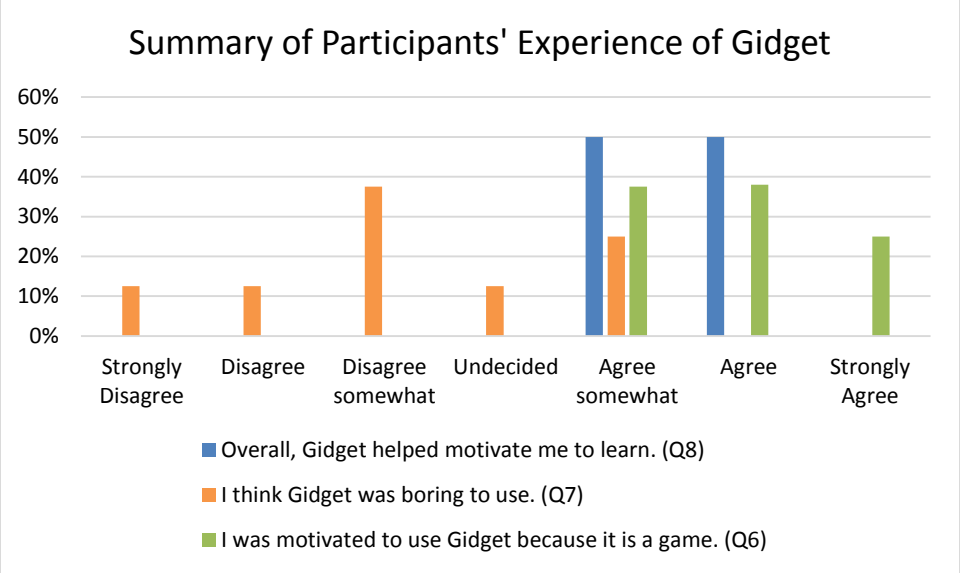


Figure 72. Responses to Post-Test Q6-Q89 for Gidget

Item Q7 sought to triangulate whether participants thought Gidget was engaging to use (Q6) and helped motivate them to learn (Q8). Qualitative data supports these responses. When prompted about what they liked about using Gidget (Q4), students responded [emphasis added]:

> *"It was a **fun** and **interactive game**."*
>
> *"It was **easy to follow**"*
>
> *"It was **like a game** and had **game like goals**!"*
>
> *"It was very slow paced and **easy to catch on**"*
>
> *"since it was a **game like setting** and fairly simple format made it **easy and approachable**"*
>
> *"I like **the 'game' feel** of it and the different **challenges were cool and difficult**, but solvable."*
>
> *"I liked the simple interface, and easy to understand objectives for each level. The program **felt a lot like a game**, which made learning different parts of code really **fun and entertaining**. It provided a **good challenge for each level**, especially in debugging the code."*
>
> *"I found that it was very easy to use and **made learning** [the] basics of coding more **interesting and entertaining**."*

Overall, the open responses suggest that students enjoyed and benefited from the game-based learning environment. Participants were not explicitly informed that Gidget was a "game-based learning" environment, although some responses specifically mentioned that it was "a fun and interactive game." Other responses noted that Gidget was "like a game" and had "game like goals" and a "game like setting." Nevertheless, most of the students observed that they liked the game feel of the experience. One student observed that the "challenges were cool and difficult, but solvable" and that it was a "good challenge for each level," suggesting that flow criterion of challenge and difficulty balance was present.

2. Mixed Engagement with Gamification

Students had mixed responses about Khan Academy's gamification elements as shown in Figure 72.

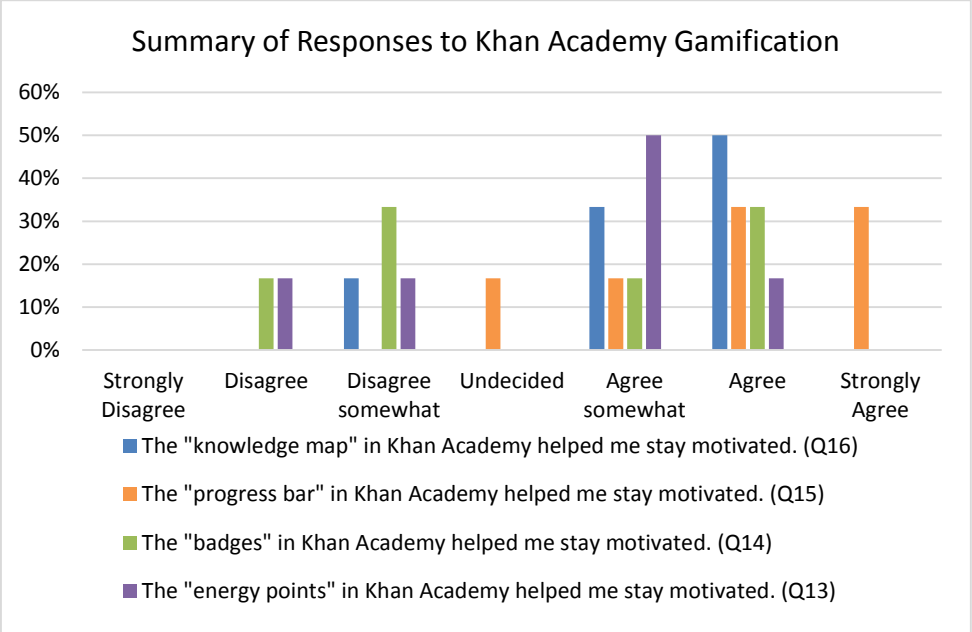


Figure 73. Responses to Post-Test Q13-Q16 for Khan Academy

Students had a more favorable about the progress bar and knowledge map than the energy points and badges. This is likely because the progress bar and knowledge map served a more specific purpose in the learning process, in that they providing direct feedback about learning progress. However, the responses to the energy points and badges were still mixed, although nobody "strongly agreed" nor "strongly disagreed" that these helped them stay motivated.

One student explicitly mentioned that the gamification components were not that encouraging, yet could be useful for a beginner [emphasis added]:

> *"The little things like **'energy points'** and **'badges'** etc. to track progress **weren't that encouraging.** I guess **they are not detrimental, but I could still do without them**. This*

might have just been my perspective, having done some other CS classes, but sometimes it felt a little bit slow at times. However, looking at it from a beginner, ***I can see where it would be helpful****."*

Therefore, gamification may better suited for students that have no previous experience in programming or where more feedback is useful during the learning process.

3. Scaffolding Support is Important for Skill Mastery

Participants also made note of the struggles they faced when the challenges were beyond their skill level. When asked what they did not like about Gidget, most of the students responded that there was not enough support for them when they needed help [emphasis added]:

> *"some of the instructions were very* ***vague****"*
>
> *"I think there should have been* ***more exercises repeating the same thing*** *instead of building on lessons every time.* ***It got difficult to remember all I had learned*** *from past coding lessons."*
>
> *"It* ***wasn't always clear*** *how to achieve a certain task, and there* ***are no definitive tutorials*** *on the site in how to accomplish each of the tasks."*
>
> *"When you are stuck you are stuck. There* ***was not too much help*** *to get you through problems"*
>
> *"I thought there were* ***very little help buttons*** *and* ***little online resources*** *to find about what the commands were. The fact that it is* ***not a real code*** *that controls gidget means that the learning experience* ***should have more description*** *on how to use the software, and another system to [learn] what the commands will do the gidget and how set them up."*
>
> *"…once it got into deeper material* ***not a lot of help was offered.*** *I have been stuck on problem number 20 for a little over an hour and I can't seem to figure it out. I found that* ***even though I took a coding class I didn't do nearly as well as I thought I was going to****."*

Students responded similarly about the support resources of using Grok Learning [emphasis added]:

> *"When you're stuck,* ***there's really nothing to help you*** *other than past lessons, but those past lessons never have the answer because the quiz* ***forces you to up your thinking****."*

These responses suggest that the learning experience could be enhanced with better tutorials and better use of incremental levels. Effective game mechanics often use previously learned skills applied those skills in new ways. However, one participant noted that she lost sight of what was being learned from each level, and struggled "even though [she] took a coding class [and] didn't do nearly as well as [she] thought." This suggests that the repetition and feedback to conduct *mastery* in constructionism is an important consideration for game-based CS learning.

In contrast, participants were generally positive about Khan Academy's use of tutorials. Overall, the combination of having video tutorials and hands-on constructionist activities worked well for the students' learning experience [emphasis added.

> *"I liked how it **walked you through the coding** and it was **not as confusing** as the other websites I tried."*
>
> *"They **explained everything really** well and simplified it enough for me to understand it"*
>
> *"learning along with the **voice in combination with seeing it done** was super helpful"*
>
> *"I have taken programming classes at CU and I think that Khan **Academy did a good job at explaining things effectively and clearly**."*
>
> *"The user interface, the **dual video design was very helpful**. I also liked the [challenge] and how **they guided you through the process** while still giving you hands on experience."*
>
> *"I liked how they would have a real person record an **interactive video** and take you through the piece of information they are trying to teach you. **Afterwards they give you a practice problem so you can remember** the things you just learned about better."*
>
> *"I liked how it **walked you through the coding** and it was not as confusing as the other websites I tried."*

It seems that if both Gidget and Grok Learning had utilized better use of tutorials and support, students would have had a more positive experience in using those tools to learn. These findings show that both useful feedback and support is important for the learning process to be successful.

4. Students Intrinsically & Extrinsically Motivated to Learn

As initially predicted when participants were recruited for this study, students appeared to be intrinsically motivated about their overall attitudes when it comes to programming. Figure 73 summarizes the results of all SIMS items used to measure intrinsic motivation.

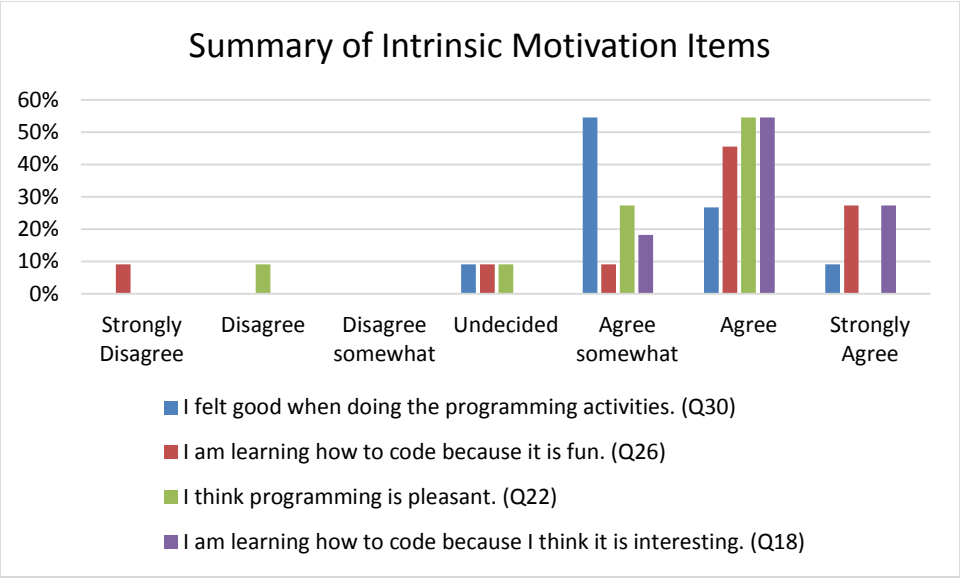


Figure 74. Summary of Intrinsic Motivation Responses of Students from SIMS

Overall, responses were positively skewed toward item responses indicated high intrinsic motivation.

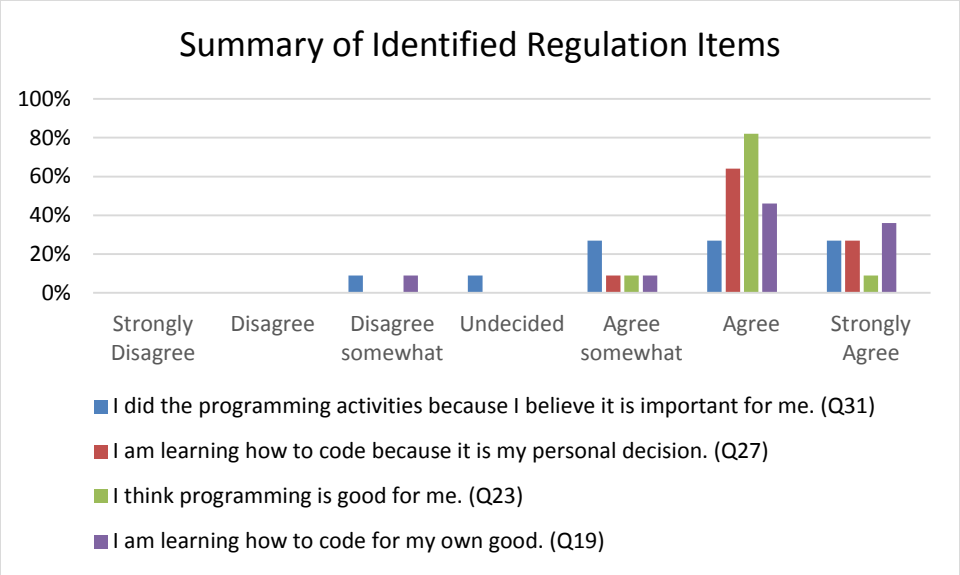


Figure 75. Summary of Identified Regulation Responses of Students from SIMS

Students also indicated that they were autonomously choosing to learn more about programming (see Figure 74). Identified regulation occurs when a behavior is valued and perceived as being chosen by oneself (Guay, Vallerand, & Blanchard, 2000). Although some students indicated that they "disagreed somewhat" that programming is an important to them (Q31) and others were "undecided" that learning how to code is for his own good (Q19), most of the students responded positively.

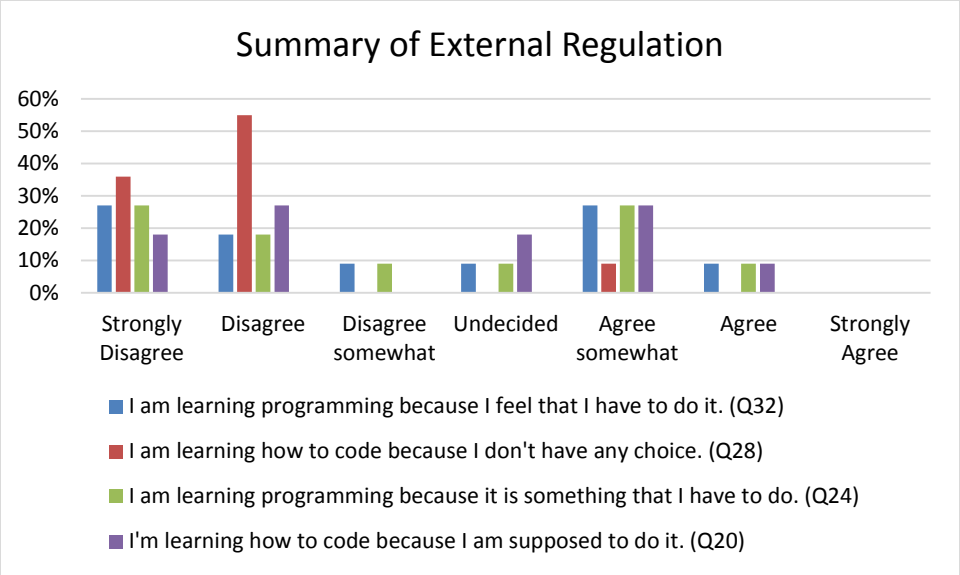


Figure 76. Summary of External Regulation Responses of Students from SIMS

Figure 75 suggests that extrinsic motivation also played a factor in motivating students to engage in programming activities. For example, most students were intrinsically motivated, and indicated item responses to the external regulation measures, as shown in Figure 76.

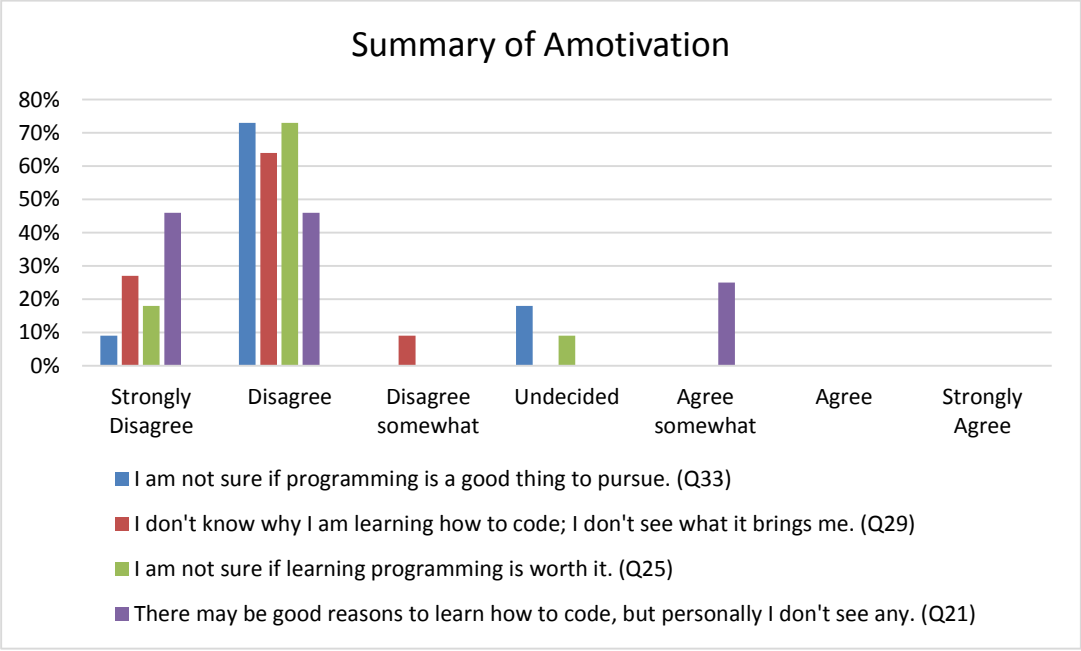


Figure 77. Summary of Amotivation Responses of Students from SIMS

Finally, students appeared to be engaging in the online training tutorials because of both intrinsic and extrinsic motivation. Most students demonstrated a purpose to pursue programming by having a negative-skew toward the reverse-item responses.

5. Pre- and Post-Learning Outcomes Not Significant

The pre-test and post-test results for the knowledge assessment were mixed. A paired *t*-test was implemented to determine if there was any significant change between the pre and post-test. Seven of eighteen knowledge questions showed an increase in the number of students answering those questions correctly. Only two items of eighteen showed a statistically significant increase $p \leq 0.05$, Q11/Q38, and Q23/Q50 (see Table 32). Four sets of questions (Q7/Q34, Q17/Q44, Q19/Q46, and Q20/Q47) saw a decrease in the number of students answering the question correctly, and seven questions reflected "no change" in the number of questions answered correctly on the post-test.

	Pre Q8	**Post Q35**	**Pre Q9**	**Post Q36**	**Pre Q10**	**Post Q37**	**Pre Q11**	**Post Q38**	**Pre Q12**	**Post Q39**	**Pre Q18**	**Post Q45**	**Pre Q23**	**Post Q50**
(Paired t-Test two-tail) ***p* value**	0.08		0.58		0.58		**0.04**		0.19		0.28		**0.04**	

Table 31. Pre-Test & Post-Test *t*-Test of Mean Scores where $p \leq 0.05$

In summary, 39% of questions were answered correctly, 39% of questions had no change in the number of responses answered correctly, and 22% of questions were answered more incorrectly in the post-test than the pre-test.

X. DISCUSSION

Although this sample size is not large enough to generalize, the data suggest that game-based learning had more of an impact on student motivation than gamification. Although

gamification did not adversely impact students in general, some students seemed to have benefited more from the gamification than others. In addition, students appeared to understand their individual intrinsic and extrinsic motivations to pursue learning introductory programming. Follow-up research could determine whether students who are extrinsically motivated to learn respond more positively to gamification than purely intrinsically motivated students. Perhaps intrinsically motivated students are more likely to engage within game-based learning, or that truly novice programmers would respond more positively to the gamification. However, more research is needed.

Pre-test and post-test questions about specific learning outcomes generally did not indicate a positive change in students' fundamental understanding of introductory programming. For example, even though students reported to have enjoyed learning from the game-based learning environment, they did not demonstrate that understanding. This was the case with all interventions, suggesting that knowledge may not be immediately transferrable after a short interaction with an instructional website.

1. Threats to Internal Validity

This study has several limitations that prevent generalizability. First, students were recruited from a single institution. Since the online learning websites generally appeal to mass audiences around the world, this is a limitation for scalability. In addition, the sample recruited was small. In addition, the CS1 knowledge assessment questions have not been validated. Again, the small sample population makes it difficult to validate this assessment. Even though fifty-seven participants completed the pre-test survey, the number of students who completed the post-test survey is a fifth of this number. Finally, students were learning introductory programming

concepts in the courses in which they were enrolled. Since these students were recruited from computer science classes, it is not feasible to tell what was learned (or unlearned) as a result of the website or the course they were taking, or if the design of the knowledge assessment itself was incompatible in some way.

XI. FUTURE WORK

Future work in this area could perhaps recruit people using Mechanical Turk. Mechanical Turk is an online marketplace where individuals aged eighteen and over can receive micro-payments for doing small tasks. Individuals could be recruited to participate in a similar pre-test and post-test survey assessments. In order to control for people who would be externally motivated to participate in this work (i.e., getting paid to take the surveys and learn on the online websites) other types of people could be recruited that have similar novice experience but without the extrinsic use of rewards, for example, from social media or from student populations not actively enrolled in any technology or programming courses at the time.

CHAPTER IX
CONCLUSIONS & FUTURE WORK

I. RESEARCH SUMMARY

Previous work on gamification does not consider the importance of both intrinsic and extrinsic motivations for learners. The U.S. educational system is predominately driven by extrinsic factors such as grades, deadlines, standardized test scores, and funding. Few studies have examined the motivating factors—both intrinsic and extrinsic—that encourage students to learn more about the fundamental and creative practices of computer science. In this dissertation, I have employed a mixed-method approach to investigate these issues. I first conducted a case study within a traditional undergraduate classroom using a quasi-experimental design research approach. The variable group utilized gamification, whereas the control group used typical educational constructs. The second case study utilized gamification in an extracurricular high school program for at-risk students. The third case study explored ways to use toys and game elements to facilitate learning for elementary and middle school children in a library workshop. The fourth case study was a preliminary exploration of the effectiveness of online training tools that employ gaming constructs to teach introductory computer science topics. In all cases, I evaluated the results using both quantitative and qualitative procedures.

In education research, there is a trend toward design-based research because design-based research focuses on activity in real classrooms. As Mark Guzdial (2011) pointedly notes:

> "You rarely come out with definitive claims supported by statistical significant that researchers expect to generalize. Instead, you end up with statements like, "Under these conditions, we can show that these interventions lead to significant learning

gains." Those kinds of statements can guide future design, and help the teacher, but avoid defining A One, Ture Way." (p. 9)

This research shows just how complex the learning process can be—what works for one student in one educational context many not necessarily work for another student.

II. SUMMARY OF FINDINGS

1. Gamification & CS Education

The findings of this work show that students appreciate *playful design* (Ferrara, 2012) when the play and the game design aligns both intrinsically and extrinsically with their motivations. However, in order for game design techniques to positively impact and influence student learning outcomes, both the internal and external requirements must be present. Overall, students seem to respond positively to game design techniques when they are used in introductory CS learning initiatives. Results in case study one, three, and four suggest that game design techniques offer motivating opportunities for students to engage with the material. In addition, these techniques appear to support the proponents of Self-Determination Theory. When students are autonomously choosing their learning endeavors—whether they are intrinsically or extrinsically motivated to do so—they can benefit from the constructionist and connectivist forms of game design in education. In addition, because of the pervasiveness of gaming culture in the U.S., game design techniques appear to appeal to a broad audience.

2. CSTA Standards

This research used the CSTA Standards as a design guideline and rubric to create actionable, design-based instructional methodologies for use in formal and non-formal learning

environments. The CSTA Standards offer classroom and learning methodology according to "levels" of student grades and ages. Each content strand—computational thinking, collaboration, computing practice and programming, computers and communications devices, and community, global, and ethical impacts—has an association with each level of CSTA Standards. After conducting several case studies that sought to leverage these design guidelines and rubrics, it appears that the CSTA Standards do not offer teachers and educational practitioners' a sufficient level of guidance of flexibility. These Standards could be improved by aligning them more closely with the CS Principles effort.

CSTA Standards need to take into consideration that learning CS does not happen at precise age or grade-levels. If CS curriculum is indeed an important part of liberal arts education, as most of us believe it is, then the educational standards that support that education must reflect the unique character of computer science. CS is not reading, or even math. In the meantime, teachers need lesson plans and activities based upon *skill levels*, as opposed to age or grade levels. Students can vary dramatically in levels of expertise, whether they self-identify as "novices" or "experts" in the field. Therefore, we need to create to ways to identify and assess levels of competency for CS fundamentals.

III. FUTURE WORK

Future work should continue to examine the connections between intrinsic and extrinsic motivations, particularly through the lens of Self-Determination Theory, as they apply to introductory computer science education. This research will likely continue to use game-based learning and gamification techniques as an educational framework. This work will likely involve faculty and researchers who teach traditional courses, online courses, and hybrid courses, in

addition to informal learning environments that occur outside of the classroom environment, such as maker-spaces, hack-a-thons, workshops, and other innovative "third spaces." This future work can focus on the way in which gaming and learning interact in the increasingly blurry lines between the social, cultural, and personal identities of students learning and engaging with the material.

Gaming is here to stay. Play is an important tool for learning in both nature and society. Understanding the nature of play, particularly game play is a critical step toward the successful adoption of computer science in educational venues of all kinds.

REFERENCES

ACM & IEEE. (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. Retrieved from http://www.acm.org/education/CS2013-final-report.pdf

Ally, M. (2008). Foundations of educational theory for online learning. In T. Anderson (Ed.), *The Theory and Practice of Online Learning* (Fifth Edit.). AU Press, Athabasca University.

Amabile, T. M. (1996). *Creativity in Context*. Boulder, CO: Westview Press.

American Association of University Women (AAUW). (2000). *Tech-Savvy: Educating girls in the new computer age*. Washington, DC: Author.

Andersen, F. O. (2005). *International trends in primary school education: An overview based on case studies in Finland, Denmark & Japan*. Bilund, Denmark: Lego Learning Institute.

Ashcraft, C., Eger, E., & Friend, M. (2012). Girls in IT: The facts. *National Center for Women and Information Technology*. Boulder: University Press of Colorado. Retrieved from NCWIT: https://www.ncwit.org/sites/default/files/resources/girlsinit_thefacts_fullreport2012.pdf

Astrachan, O., & Briggs, A. (2012). The CS principles project. *ACM Inroads*, *3*(June), 38–42.

Arpaci-Dusseau, A., Astrachan, O., Barnett, D., Bauer, M., Carrell, M., Dovi, R., … Uche, C. (2013). Computer science principles: analysis of a proposed advanced placement course. In *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13)* (pp. 251–256). Denver, CO: ACM New York. Retrieved from http://dl.acm.org/citation.cfm?id=2445273

Barab, S. A., Gresalfi, M., Arici, A., Pettyjohn, P., & Ingram-Goble, A. (2010). Transformative play: games as 21st century curriculum. In *ICLS '10: Proceedings of the 9th International Conference of the Learning Sciences - Volume 2* (Vol. 2, pp. 93–100). International Society of the Learning Sciences. Retrieved from ACM: http://dl.acm.org/citation.cfm?id=1854548

Barnes, T., Richter, H., & Powell, E. (2007). Game2Learn: Building CS1 learning games for retention. In *ITiCSE '07: Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education* (pp. 121–125). New York, NY: ACM.

Barr, V., & Stephenson, C. (2011, March). Bringing computational thinking to K-12: What is Involved and what is the role of the computer science education community? *ACM Inroads*, *2*(1), 48–54.

Basawapatna, A. R., Koh, K. H., & Repenning, A. (2010). Using scalable game design to teach computer science from middle school to graduate school. In *ITiCSE '10: Proceedings of the fifteenth annual conference on Innovation and technology in computer science education* (p. 224). New York, NY: ACM.

Bayliss, J. D., & Strout, S. (2006). Games as a "flavor" of CS1. In *SIGCSE '06 Proceedings of the 37th SIGCSE technical symposium on Computer science education* (Vol. 38, pp. 500–504). New York, NY: ACM. doi:10.1145/1124706.1121498

Behnke, K. A. (2012). Ladies of *Warcraft*: Changing perceptions of women and technology through productive play. In *FDG'12: Proceedings of the International Conference on the Foundations of Digital Games* (pp. 288-289). New York: ACM.

Behrens, A., Atorf, L., Schneider, D., & Aach, T. (2011). Key factors for freshmen education using MATLAB and LEGO mindstorms. In *ICIRA'11 Proceedings of the 4th international conference on Intelligent Robotics and Applications - Volume Part I* (pp. 553–562). Springer-Verlag Berlin, Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-25486-4_55

Bogost, I. (2007). *Persuasive games: The expressive power of videogames*. Cambridge, MA: MIT Press.

Bogost, I. (2011). Persuasive Games: Exploitationware. *Gamasutra*, May 3. Retrieved from http://www.gamasutra.com/view/feature/134735/persuasive_games_exploitationware.php. Accessed April 2015.

Bonar, J., & Soloway, E. (1985). Preprogramming knowledge: A major source of misconceptions in novice programmers. Human–Computer Interaction, 1(2), 133-161.

Boustedt, J., Eckerdal, A., McCartney, R., Sanders, K., Thomas, L., & Zander C. (2011). Students' perceptions of the differences between formal and informal learning. ACM ICER, 61–68.

Boyer, N. R., Langevin, S., & Gaspar, A. (2008). Self direction & constructivism in programming education. In *SIGITE '08: Proceedings of the 9th ACM SIGITE conference on Information technology education* (p. 89). New York: ACM Press.

Briggs, A., & Snyder, L. (2012). Computer Science Principles and the CS 10K Initiative. *ACM Inroads*, *3*(2), 30–31.

Brockett, R. G. & Hiemstra, R. (1991). *Self-direction in adult learning: Perspectives on theory, research, and practice*. London and New York: Routledge.

Brown, T. (2008). Design thinking. *Harvard Business Review*, *86*(6), 84–92, 141. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18605031

Buechley, L., Eisenberg, M., & Catchen, J., & Crockett, A. (2008). The LilyPad Arduino: Using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In *CHI '08 Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (pp. 423–432). New York: ACM.

Castronova, E. (2005). *Synthetic worlds: The business and culture of online games*. Chicago, IL: University of Chicago Press.

College Board. (2013). *AP Computer Science Principles Draft Curriculum Framework*. Retrieved from College Board: http://secure-media.collegeboard.org/digitalServices/pdf/ap/ap-computer-science-principles-curriculum-framework.pdf

Collins, A. & Halverson, R. (2009). *Rethinking Education in the Age of Technology: The Digital Revolution and Schooling in America*. New York: Teachers College Press.

Cooper, S., Dann, W., & Pausch, R. (2003). Teaching objects-first in introductory computer science. In *SIGCSE '09: Proceedings of the 34th ACM technical symposium on Computer science education*. New York: ACM Press.

Corno, L. & Mandinach, E. B. (1983). The role of cognitive engagement in classroom learning and motivation. *Educational Psychologist*, *18*(2), 88–108.

Cowley, B., Charles, D., Black, M., & Hickey, R. (2008). Toward an understanding of flow in video games. *Computers in Entertainment (CIE) - Theoretical and Practical Computer Applications in Entertainment*, *6*(2), 1–27.

Charsky, D. (2010). From edutainment to serious games: A change in the use of game characteristics. *Games and Culture*, *5*(2), 177–198.

Creswell, J. W. (2003). Chapter One, "A Framework for Design." *Research Design Qualitative Quantitative and Mixed Methods Approaches*, 3–26.

Cross, J. (2006). Informal learning: rediscovering the natural pathways that inspire innovation and performance. San Francisco, CA: Pfeiffer

Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience*. New York: Harper and Row.

Csikszentmihalyi, M. (1997). *Finding Flow: The Psychology of Engagement with Everyday Life*. New York: Basic Books.

Csikszentmihalyi, M. (2003). *Good Business: Leadership, Flow, and the Making of Meaning*. New York, NY: Viking Adult.

Csikszentmihalyi, M., Rathunde, K., & Whalen, S. (1993). *Talented teenagers: The roots of success and failure*. New York: Cambridge University Press.

Deci, E. L., & Ryan, R. M. (1991). A motivational approach to self: Integration in personality. In R. Dienstbier (Ed.), *Nebraska symposium on motivation*, *38*(1), 237-288. Lincoln, NE: University of Nebraska Press.

Decker, A., & Lawley, E. L. (2013). Life's a game and the game of life: How making a game out of it can change student behavior. In *SIGCSE '13: Proceeding of the 44th ACM technical symposium on Computer science education* (pp. 233–237). New York, NY: ACM.

Deterding, S. (2011). Gamification by design: Response to O'Reilly. *Gamification Research Network*. Retrieved from http://gamification- research.org/2011/09/gamification-by-design-response-to- oreilly/

Deterding, S., & Dixon D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification." In *MindTrek'11: Proceedings of the 15th International Academic MindTrek Conference: Envision Future Media Environments, September, Tampere, Finland*. New York: ACM. doi: 10.1145/2181037.2181040

Deterding, S., & Dixon, D. (2011). Gamification: Toward a definition. In *CHI '11: Proceedings of the 2011 annual conference on Human factors in computing systems* (pp. 2425–2428). New York: ACM . Retrieved from http://gamification-research.org/wp-content/uploads/2011/04/02-Deterding-Khaled-Nacke-Dixon.pdf

Dickey, M. D. (2005). Engaging by design: How engagement strategies in popular computer and video games can inform instructional design. *Educational Technology Research & Development*, *53*(2), 67–83.

Dignan, A. (2011). *Game Frame: Using games as a strategy for success*. New York, NY: Free Press.

Downes, S. (2006). *An Introduction to Connective Knowledge*. Creative Commons License. Retrieved from http://www.downes.ca/post/33034

Downes, S. (2012). *Connectivism and Connective Knowledge: Essays on Meaning and Learning Networks*. Creative Commons License. Retrieved from http://www.downes.ca/files/Connective_Knowledge-19May2012.pdf

Duckworth, A. L., Peterson, C., Matthews, M. D., & Kelly, D. R. (2007). Grit: perseverance and passion for long-term goals. *Journal of Personality and Social Psychology*, *92*, 1087–1101.

Ducheneaut, N., Yee, N., & Nickell, E. (2006). "Alone together?": Exploring the social dynamics of massively multiplayer online games. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems* (pp. 407–416). New York, NY: ACM. doi:10.1145/1124772.1124834

Ducheneaut, N., Yee, N., Nickell, E., & Moore, R. J. (2007). The life and death of online gaming communities: A look at guilds in *World of Warcraft*. In *CHI '07: Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 839–848). ACM Press.

Dunn, E. W., Ankin, L. B., & Norton, M. I. (2008). Spending money on others promotes happiness. *Science, 21*(March).

Dweck, C. (2006a). *Mindset: The New Psychology of Success*. New York: Ballantine Books.

Eagle, M., & Barnes, T. (2009). Experimental evaluation of an educational game for improved learning in introductory computing. In *SIGCSE '09: Proceedings of the 40th ACM technical symposium on Computer science education* (p. 321). New York, NY: ACM Press.

Entertainment Software Association (ESA). (2013). *2013 essential facts about the computer and video game industry*. Washington DC: ESA. Retrieved from ESA: http://www.theesa.com/wp-content/uploads/2014/10/ESA_EF_2014.pdf

Ericson, B., & Guzdial, M. (2014). Measuring demographics and performance in computer science education at a nationwide scale using AP CS data. *Proceedings of the 45th ACM Technical Symposium on Computer Science Education (SIGCSE '14)*, 217–222. http://doi.org/10.1145/2538862.2538918

Ericson, B., Guzdial, M., & Biggers, M. (2007). Improving secondary CS education: Progress and problems. In *SIGCSE '07 Proceedings of the 38th SIGCSE technical symposium on Computer science education* (pp. 298–301). New York, NY: ACM.

Falkner, N., & Falkner, K. (2014). Whither, badges? or wither, badges!: A metastudy of badges in computer science education to clarify effects, significance and influence. In Koli Calling '14: *Proceedings of the 14th Koli Calling International Conference on Computing Education Research* (pp. 127–135). New York: ACM.

Ferrara, J. (2012). *Playful Design: Creating Game Experiences in Everyday Interfaces*. New York, NY: Rosenfeld Media.

Fincher, S., Lister, R., Clear, T., Robins, A., Tenenberg, J., & Petre, M. (2005). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. In *ICER '05 Proceedings of the first international workshop on Computing education research* (pp. 111–121). New York: ACM Press.

Fuchs, M., Fizek, S., Ruffino, P., & Schrape, N. (2014). *Rethinking Gamification*. Hybrid Publishing Lab.

Furst, M., Isbell, C., & Guzdial, M. (2007). Threads™: How to restructure a computer science curriculum for a flat world. In *SIGCSE '07 Proceedings of the 38th SIGCSE technical symposium on Computer science education* (pp. 420–424). New York: ACM.

Gaggioli, A., Bassi, M., & Fave, A. (2003). Quality of experience in virtual environments. In G. Riva, F. Davide, & W. A. Jsselsteijn (Eds.), *Being There: Concepts, Effects and Measurement of User Presence in Synthetic Environments*. Amsterdam, the Netherlands: Ios Press.

Gee, J. P. (2003). *What Video Games Have to Teach Us About Learning and Literacy*. New York: Palgrave Macmillan Press.

Gee, J. P. (2006). Why game studies now? Video games: A new art form. *Games and Culture*, *1*(1), pp. 58–61. Thousand Oaks, CA: SAGE Publications.

Gee, J. P. (2008). Learning and games. In K. Salen (Ed.) *The ecology of games: Connecting youth, games, and learning*. Cambridge, MA: The MIT Press.

Gidget. http://www.helpgidget.org. Accessed October 2015.

Guay, F., Vallerand, R.J. and Blanchard, C. (2000). On the Assessment of Situational Intrinsic and Extrinsic Motivation: The Situational Motivation Scale (SIMS). Motivation and Emotion. 24, 3 (Sep. 2000), 175–213.

Greenbaum, J. and Kyng, M. (1992). Introduction: situated design. In *Design at work*, Joan Greenbaum and Morten Kyng (Eds.). L. Erlbaum Associates Inc., Hillsdale, NJ, USA 1-24.

Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does Gamification Work? -- A Literature Review of Empirical Studies on Gamification. *Proceedings of the 47th Hawaii International Conference on System Sciences*, 3025–3034. http://doi.org/10.1109/HICSS.2014.377

Harlow, H. F., Harlow M. K., & Meyer, D. R. (1950). Learning motivated by a manipulation drive. *Journal of Experimental Psychology, 40*(1950), 231-234.

Henderson, B., Meier, D. R., Perry, G., & Stremmel, A. J. (2012). The nature of teacher research. *Voices of Practioners*. National Association for the Education of Young Children. Retrieved at https://www.naeyc.org/files/naeyc/file/vop/Nature%20of%20Teacher%20Research.pdf accessed October 2015.

Hu, C. (2011). Computational Thinking – What it might mean and what we might do about it. In *ITiCSE '11: Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* (pp. 223–227). New York, NY: ACM.

Huizinga, J. (1944). *Homo Ludens: A study of the play-element in culture* (English 19.). Switzerland: Routledge & Kegan Paul Ltd.

Huotari, K., & Hamari, J. (2012). Defining gamification: a service marketing perspective. In *MindTrek '12: Proceedings of the 16th International Academic MindTrek Conference*. *Retrieved from* http://dl.acm.org/citation.cfm?id=2393137

Hsieh, H. F. & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research, 15*(9).

Jewell, P. A. & Loizos, C. (1966). Play, exploration and territoriality in mammals. *Symposia of the Zoological Society*, London, 18.

Kafai, Y. & Resnick, M., Eds. (1996). *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Kafai, Y. B. (2006). Playing and Making Games. *Games and Culture*, (October 2001), 36–40.

Kafai, Y. B., Heeter, C., Denner, J., & Sun, J. Y. (2008). Eds., *Beyond* Barbie *&* Mortal Kombat*: New perspectives on gender and gaming*. Cambridge, MA: MIT Press.

Kapp, K. M., Blair, L. & Mesch, R. (2014) *The Gamification of Learning and Instruction Fieldbook: Theory into Practice*. New York: John Wiley & Sons.

Khaled, R., Barr, P., Biddle, R., Fischer, R., & Noble, J. (2009). Game design strategies for collectivist persuasion. In *SIGGRAPH '09: Proceedings of the 2009 ACM Special Interest Group on Computer Graphics and Interactive Techniques, Sandbox '09, New Orleans, LA* (pp. 31–38). New York: ACM.

Kim, J. T., & Lee, W. H. (2013). Dynamical model for gamification of learning (DMGL). *Multimedia Tools and Applications*, *74*(219).

Kim, E. S., & Willson, V. L. (2010). Evaluating Pre-test Effects in Pre-Post Studies. *Educational and Psychological Measurement*, *70*(5), 744–759.

Knowles, M. (1975). *Self-directed Learning: A Guide for Teachers and Learners*. New York: Association Press.

Kondracki, N. L., Wellman, N. S., & Amundson, D. R. (2002). Content Analysis: Review of Methods and Their Applications in Nutrition Education. *Journal of Nutrition Education and Behavior*. Retrieved from http://doi.org/10.1016/S1499-4046(06)60097-3

Kop, R. (2011). The challenges to connectivist learning on open online networks: Learning experiences during a massive open online course. *International Review of Research in Open Distance Learning*, *12*(3).

Koster, R. (2005). *A Theory of Fun*. Scottsdale, AZ: Paraglyph Press.

Lee, J. J., & Hammer, J. (2011). Gamification in education: What, how, why bother? What. *Academic Exchange Quarterly*, *15*(2), 1-5.

Lee, M.J, Bahmani, F., Kwan, I., Laferte, J., Charters, P., Horvath, A., Luor, F., Cao, J., Law, C., Beswetherick, M., Long, S., Burnett, M., & Ko, A.J. (2014). Principles of a Debugging-First Puzzle Game for Computing Education. IEEE VL/HCC, 57-64.

Lieberoth, A. (2014). Shallow gamification: Testing psychological effects of framing an activity as a game. *Games and Culture*, *10*(3), 229–248.

Lightbot. http://www.lightbot.com. Accessed: October 2015.

Linhoff, J., & Settle, A. (2009). Motivating and evaluating game development capstone projects. In *FDG '09: Proceedings of the 4th International Conference on Foundations of Digital Games (Orlando, Flordia, April 26–30, 2009)* (pp. 121–128). New York: ACM.

Lindlof, T. R., & Taylor, B. C. (2002). *Qualitative Communication Research Methods: Third Edition*. Thousand Oaks, CA: SAGE Publications.

Locke, E. a., & Latham, G. P. (2002). Building a practically useful theory of goal setting and task motivation: A 35-year odyssey. *American Psychologist*, *57*(9), 705–717. doi:10.1037//0003-066X.57.9.705

Malaby, T. M. (2007). Beyond play: A new approach to games. *Games and Culture*, *2*(2), 95–113. doi:10.1177/1555412007299434

Malan, D. J., & Leitner, H. H. (2007). Scratch for budding computer scientists. In *SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education* (Vol. 39, pp. 223–227). New York, NY: ACM. doi:10.1145/1227504.1227388

Malone, T. W. (1980a). What makes things fun to learn? A study of intrinsically motivating computer games. *Cognitive and Instructional Sciences Series*, *CIS-7*(SSL-80-11).

Malone, T. W. (1980b). What makes things fun to learn? Heuristics for Designing Instructional Computer Games. In *SIGSMALL '80: Proceedings of the 3rd ACM SIGSMALL symposium and the first SIGPC symposium on Small systems* (Vol. 162, pp. 162–169). New York, NY: ACM.

Malone, T. W., & Lepper, M. R. (1987). Making learning fun: A taxonomy of intrinsic motivations for learning. *Aptitude, Learning, and Instruction: Cognitive and Affective Process Analyses*, *3*, 223–253.

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. In *SIGCSE '08: Proceedings of the 39th SIGCSE technical symposium on Computer science education* (pp. 367–371). New York, NY: ACM.

Maloney, J., Resnick, M., & Rusk, N. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), 1–15.

Margolis, J., & Fisher, A. (2002). *Unlocking the computer clubhouse: Women in computing*. Cambridge, MA: MIT Press.

Mattern, K., Shaw, E., & M. Ewing. (2011). Advanced Placement exam participation: Is AP exam participation and performance related to choice of college major? Retrieved from College Board: https://research.collegeboard.org/sites/default/files/info2go/2012/8/infotogo-2011-6-ap-participation-performance-major-choice.pdf

McGonigal, J. (2011). *Reality is Broken: Why Games Make Us Better and How They Can Change the World*. New York, NY: The Penguin Press.

McNamara, D., Jackson, G., Graesser, A. (2009) Intelligent tutoring and games. Artificial Intelligence in Education, 1–10.

Michael, D. R., & Chen, S. L. (2005). *Serious games: Games that educate, train and inform*. Muska & Lipman/Premier-Trade.

Miller, C. S., Lehman, J. F., & Koedinger, K. R. (1999). Goals and learning in microworlds. *Cognitive Science*, *23*(3), 305–336.

Mitgutsch, K. (2012). Learning through play - A delicate matter: Experience-based recursive learning computer games. In J. Fromme & A. Unger (Eds.), *Computer Games and New Media Cultures* (pp. 571–584). Dordrecht: Springer Netherlands. doi:10.1007/978-94-007-2777-9

Mitgutsch, K., & Alvarado, N. (2012). Purposeful by design? A Serious game design assessment framework. In *FDG '12: Proceedings of the 7th International Conference on Foundations of Digital Games* (pp. 121–128). New York: ACM.

Mukhopadhyay, M., & Parhar, M. (2001). Instructional design in multi- channel learning system. British Journal of Education Technology, 32(5), 543–556.

National Center for Women in Information Technology (NCWIT). (2015). Women by the numbers: Revolutionizing the face of technology. *National center for women and information technology*. Boulder: University Press of Colorado. Retrieved at NCWIT.org: http://www.ncwit.org/BytheNumbers

National Research Council (NRC). (2003). *Beyond productivity: Information technology, innovation, and creativity*. Washington DC: National Academies Press.

National Research Council (NRC). (2004). Report of a workshop on the scope and nature of computational thinking. *Committee for the Workshops on Computational Thinking*. Washington DC. Retrieved from http://onlinelibrary.wiley.com/doi/10.1002/cbdv.200490137/abstract

National Research Council (NRC). (2005). *Avoiding surprise in an era of global technology advances*. Washington DC: National Academies Press. Retrieved from http://www.nap.edu/catalog.php?record_id=11286

National Research Council (NRC). (2007). *Rising above the gathering storm: Energizing and employing America for a brighter economic future*. Washington DC: National Academies Press. Retrieved from http://www.nap.edu/catalog.php?record

National Research Council (NRC). (2010). *The rise of games and high performance computing for modeling and simulation*. Washington DC: The National Academies Press. Retrieved from http://www.nap.edu/openbook.php?record_id=12816

Nicholson, S. (2012). A user-centered theoretical framework for meaningful gamification. Paper Presented at *Games+Learning+Society 8.0*, Madison, WI. Retrieved from http://scottnicholson.com/pubs/meaningfulframework.pdf

Ochoa, X., & Duval, E. (2008). Quantitative analysis of user-generated content on the Web. *Distribution*, *34*(1), 19–26.

Owens, B. B., & Stephenson, C. (2011). *CSTA K – 12 Computer Science Standards*. New York, NY. Retrieved from http://dl.acm.org/citation.cfm?id=2325380

Papert, S. (1971). Teaching children to be mathematicians versus teaching about mathematics. *International Journal of Mathematical Education in Science and Technology*, *3*(3). doi: 10.1080/0020739700030306

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Papert, S. & Harel, I. (1991). *Constructionism*. Ablex Publishing Corporation.

Pink, D. H. (2009). *Drive: The Surprising Truth about What Motivates Us*. New York: Riverhead Books.

Pope, D. (2003). Doing school: How we are creating a generation of stressed-out, materialistic, and miseducated students. Yale University Press.

Potter, W. J., & Levine-Donnerstein, D. (1999). Rethinking validity and reliability in content analysis. *Journal of Applied Communication Research*. Retrieved from http://doi.org/10.1080/00909889909365539

Prensky, M. (2003). Digital game-based learning. *Computers in Entertainment (CIE) - Theoretical and Practical Computer Applications in Entertainment*, *1*(1). doi:10.1145/950566.950596

Rathnude, K., & Csikszentmihalyi, M. (2005). Middle school students' motivation and quality of experience: A comparison of Montessori and traditional school environments. *American Journal of Education, 111*, 341-371.

Raybourn, E. M. & Bos, N. C. (2005). Design and evaluation challenges of serious games. In *CHI EA '05: Extended Abstracts on Human Factors in Computing Systems* (pp. 2049–2050). doi:10.1145/1056808.1057094

Raybourn, E. M., Deagle, E., Mendini, K., & Heneghan, J. (2005). Adaptive thinking & leadership simulation game training for special forces officers. *I/ITSEC 2005 Proceedings, Interservice/ Industry Training, Simulation and Education Conference*. Orlando, FL.

Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. *SIGCSE '10: Proceedings of the 41st ACM technical symposium on Computer science education*. New York: ACM Press.

Richards, C., Thompson, C. W., & Graham, N. T. C. (2014). Beyond Designing for Motivation : The Importance of Context in Gamification. In *CHI PLAY '14 Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play* (pp. 217–226). Toronto, Canada: ACM New York.

Rieber, L. P. (1996). Seriously considering play: Designing interactive learning environments based on the blending of microworlds, simulations, and games. *Educational Technology Research & Development*, *44*(2), 43–58. Retrieved from http://www.jstor.org/stable/30221022

Ritzhaupt, A. D. (2009). Creating a game development course with limited resources: An evaluation study. *Transactions on Computing Education*, *9*(1).

Rovai, A. P., Baker, J. D., & Ponton, M. K. (2014). *Social Science Research Design and Statistics: A Practitioner's Guide to Research Methods and IBM SPSS Analysis.* Chesapeake, VA: Watertree Press.

Ryan, R. M., Connell, J. P., & Grolnick, W. S. (1993). When achievement is *not* intrinsically motivated: A theory of self-regulation in school. In A. K. Boggiano & T. S. Pittman (Eds.), *Achievement and Motivation: A Social-developmental Perspective*. New York: Cambridge University Press.

Ryan, R. M. & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist, 55*, 68–78.

Ryan, R. M., & Powelson, C. L. (1991). Autonomy and relatedness as fundamental to motivation and education. *The Journal of Experimental Educational: Unraveling Motivation*, *60*(1), 49–66.

Ryan, R. M., Rigby, C. S., & Przybylski, A. (2006). The motivational pull of video games: A self-determination theory approach. *Motivation and Emotion*, *30*(4), 344–360. Retrieved from ACM: http://doi.org/10.1007/s11031-006-9051-8

Salen, K. (2008). *The Ecology of Games: Connecting Youth, Games, and Learning*. (K. Salen, Ed.). doi:10.1162/dmal.9780262693646.vii

Sanchez, A., & Smith, P. A. (2007). Emerging technologies for military game-based training. *SpringSim*, *2*(1), 296–301.

Schiefele, U., Krapp, A., & Winteler, A. (1992). Interest as a predictor of academic achievement: A meta-analysis of research. In K. A. Renninger, S. Hidi, & A. Krapp (Eds.), The role of interest in learning and development (pp. 183– 212). Hillsdale, NJ: Erlbaum.

Schmitz, B., Czauderna, A., Klemke, R., & Specht, M. (2011). Game based learning for computer science education. In *CSERC '11: Computer Science Education Research Conference* (pp. 81–86). Heerlen, The Netherlands: Open Universiteit, Heerlen. Retrieved from http://dl.acm.org/citation.cfm?id=2043601

Shaffer, D. W. (2008). *How Computer Games Help Children Learn*. New York: Palgrave Macmillan.

Sheldon, L. (2012). *The Multiplayer Classroom: Designing Coursework as a Game*. Boston, MA: Course Technology Press.

Shernoff, D. J. (2013). *Optimal Learning Environments to Promote Student Engagement*. New York: Springer.

Shernoff, D. J., Csikszentmihalyi, M., Schneider, B., & Shernoff, E. S. (2003). Student engagement in high school classrooms from the perspective of flow theory. *School Psychology Quarterly, 18*, 158-176.

Siegel, M., (2003). The sense-think-act paradigm revisited. In ROSE '03: *Proceedings of the 1st International Workshop on Robotic Sensing, 5-6 June*. IEEE.

Siemens, G. (2005). Connectivism: A learning theory for the digital age. *International Journal of Instructional Technology and Distance Learning*, *2*(1), 3–10.

Sime, M., Green, T., & Guest, D. (1976). Scope marking in computer conditionals: A psychological evaluation. International Journal of Man-Machine Studies, 9, 107–118.

Simonsen, J., Svabo, C., Strandvad S. M., Samson, K., Hertzum, M., Hansen, O. E. (Eds.) (2014). *Situated Design Methods*. MIT Press.

Snyder, L., Barnes, T., Garcia, D., Paul, J., & B. Simon (2012). The first five computer science principles pilots: Summary and comparisons. *ACM Inroads*, *3*(2), 54–57.

Squire, K. (2011). *Video Games and Learning*. New York: Teachers College Press.

Squire, K. D. (2003). Video games in education. *International Journal of Intelligent Games & Simulation*, *2*(1), 49–62.

Squire, K. (2008). Open-ended video games: A model for developing learning for the interactive age. In K. Salen (Ed.), *The Ecology of Games: Connecting Youth, Games, and Learning* (pp. 167–198). Cambridge, MA: MIT Press.

Stake, R. E. (1995). *The Art of Case Study Research*. Thousand Oaks, CA: SAGE Publications.

Stott, A. and Neustaedter, C. (2013), Analysis of Gamification in Education, Technical Report 2013-0422-01, Connections Lab, Simon Fraser University, Surrey, BC, Canada, April, 8 pages.

Strauss, A. & Corbin, J. (1990). *Basics of Qualitative Research*. Newbury Park, CA: SAGE Publications.

Sweetser, P. & Johnson, D. (2004). Player-centered game environments: Assessing player opinions, experiences and issues. In *ICEC 2004: Proceedings of the Third International Conference in Entertainment Computing - LNCS 3166, Springer Verlag, New York*, 321-332.

Tapscott, D. (1998). Growing up digital: The rise of the Net generation. New York: McGraw-Hill.

Tew, A. E. (2010). Assessing Fundamental Introductory Computing Concept Knowledge in a Language Independent Manner Assessing Fundamental Introductory Computing Concept Knowledge, (December).

Tew, A. E., Fowler, C., & Guzdial, M. (2005). Tracking an Innovation in Introductory CS Education from a Research University to a Two-Year College. In SIGCSE '05: Proceedings of the 36th SIGCSE technical symposium on Computer science education (pp. 416–420).

Thomas, D. R. (2006). A General Inductive Approach for Analyzing Qualitative Evaluation Data. *American Journal of Evaluation*, *27*(2), 237–246.

Treanor, M., Schweizer, B., Bogost, I., & Mateas, M. (2012). The micro-rhetorics of Game-o-Matic. In *FDG '12: Proceedings of the 7th International Conference on Foundations of Digital Games* (pp. 18–25). New York, NY: ACM. doi:10.1145/2282338.2282347

Trochim, William M. The Research Methods Knowledge Base, 2nd Edition. Retrieved at http://www.socialresearchmethods.net/kb/

Turner, J. C., & Meyer, D. K. (2004). A classroom perspective on the principle of moderate challenge in mathematics. *Journal of Educational Research, 97*, 311-318.

U.S. Department of Labor. (2013). Bureau of Labor Statistics, Occupational Category: 15–0000. Washington DC: Author. Retrieved from http://www.bls.gov/oes/current/oes150000.htm

Utting, I., Cooper, S., & Kölling, M. (2010). Alice, Greenfoot, and Scratch – A discussion. *ACM Transactions on Computing Education (TOCE)*, *10*(4), 1–11. doi:10.1145/1868358.1868364.http

Van Eck, R. (2006). Digital game-based learning: It's not just the digital natives who are restless. *EDUCAUSE Review*, *41*(2 (March/April 2006)), 16–30. Retrieved from http://www.educause.edu/apps/er/erm06/erm062.asp

Von Ahn, L., & Dabbish, L. (2008). Designing games with a purpose. *Communications of the ACM*, *51*(8).

Vygotsky, L. S. (1978). *Mind in Society: The Development of Higher Psychological Processes*.

Wagner, T. (2012). *Creating Innovators: The Making of Young People Who Will Change the World*. Scribner.

Wideman, H. H., Owston, R. D., Brown, C., Kushniruk, A., Ho, F., & Pitts, K. C. (2007). Unpacking the potential of educational gaming: A new tool for gaming research. *Simulation Gaming*, *38*(1), 10–30.

Williams, D. (2006). Groups and goblins: The social and civic impact of online gaming. *Journal of Broadcasting and Electronic Media*, *50*(4), 651-670.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35.

Wong, W. L., Shen, C., Nocera, L., Carriazo, E., Tang, F., Bugga, S., Narayanan, H., Wang, H., & Ritterfeld, U. (2007). Serious video game effectiveness. In *Proceedings of the international conference on Advances in computer entertainment technology* (pp. 49–55). New York: ACM.

World Technology Evaluation Center (WTEC). (2009). International assessment of simulation-based engineering and science. Baltimore, MD: Author. Retrieved from WTEC: http://www.wtec.org/sbes

Yee, N. (2006). Motivations for play in online games. *Cyberpsychology & Behavior : The Impact of the Internet, Multimedia and Virtual Reality on Behavior and Society*, *9*(6), 772–5. doi:10.1089/cpb.2006.9.772

APPENDIX A
DETAILED LIST OF CSTA STANDARDS

Level 1 Elementary
Computer Science and Me
Primary (Grades K-3)

Strand One: Computational Thinking	1. Use technology resources (e.g., puzzles, logical thinking programs) to solve age-appropriate problems. 2. Use writing tools, digital cameras, and drawing tools to illustrate thoughts, ideas, and stories in a step-by-step manner. 3. Understand how to arrange (sort) information into useful order, such as sorting students by birth date, without using a computer. 4. Recognize that software is created to control computer operations. 5. Demonstrate how 0s and 1s can be used to represent information.
Strand Two: Collaboration	1. Gather information and communicate electronically with others with support from teachers, family members, or student partners. 2. Work cooperatively and collaboratively with peers, teachers, and others using technology.
Strand Three: Computing Practice and Programming	1. Use technology resources to conduct age-appropriate research. 2. Use developmentally appropriate multimedia resources (e.g., interactive books and educational software) to support learning across the curriculum. 3. Create developmentally appropriate multimedia products with support from teachers, family members, or student partners. 4. Construct a set of statements to be acted out to accomplish a simple task (e.g., turtle instructions). 5. Identify jobs that use computing and technology. 6. Gather and organize information using concept-mapping tools.
Strand Four: Computers and Communications Devices	1. Use standard input and output devices to successfully operate computers and related technologies.
Strand Five: Community, Global and Ethical Impacts	1. Practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems and software. 2. Identify positive and negative social and ethical behaviors for using technology.

Level 1 Elementary
Computer Science and Me
Intermediate (Grades 4-6)

Strand One: Computational Thinking	1. Understand and use the basic steps in algorithmic problem-solving (e.g., problem-statement and exploration, examination of sample instances, design, implementation, and testing). 2. Develop a simple understanding of an algorithm (e.g., search, sequence of events, or sorting) using computer-free exercises. 3. Demonstrate how a string of bits can be used to represent alphanumeric information. 4. Describe how a simulation can be used to solve a problem. 5. Make a list of sub-problems to consider while addressing a larger problem. 6. Understand the connections between computer science and other fields.
Strand Two: Collaboration	1. Use productivity technology tools (e.g.,word processing, spreadsheet, presentation software) for individual and collaborative writing, communication, and publishing activities. 2. Use online resources (e.g., email, online discussions, collaborative web environments) to participate in collaborative problem-solving activities for the purpose of developing solutions or products. 3. Identify ways that teamwork and collaboration can support problem solving and innovation.
Strand Three: Computing Practice and Programming	1. Use technology resources (e.g., calculators, data collection probes, mobile devices, videos, educational software, and web tools) for problem-solving and self-directed learning. 2. Use general-purpose productivity tools and peripherals to support personal productivity, remediate skill deficits, and facilitate learning. 3. Use technology tools (e.g., multimedia and text authoring, presentation, web tools, digital cameras, and scanners) for individual and collaborative writing, communication, and publishing activities. 4. Gather and manipulate data using a variety of digital tools. 5. Construct a program as a set of step-by-step instructions to be acted out (e.g., make a peanut butter and jelly sandwich activity). 6. Implement problem solutions using a block- based visual programming language. 7. Use computing devices to access remote information, communicate with others in support of direct and independent learning, and pursue personal interests.

	8. Navigate between webpages using hyperlinks and conduct simple searches using search engines. 9. Identify a wide range of jobs that require knowledge or use of computing. 10. Gather and manipulate data using a variety of digital tools.
Strand Four: Computers and Communications Devices	1. Demonstrate an appropriate level of proficiency with keyboards and other input and output devices. 2. Understand the pervasiveness of computers and computing in daily life (e.g., voice mail, downloading videos and audio files, microwave ovens, thermostats, wireless Internet, mobile computing devices, GPS systems). 3. Apply strategies for identifying simple hardware and software problems that may occur during use. 4. Identify that information is coming to the computer from many sources over a network. 5. Identify factors that distinguish humans from machines. 6. Recognize that computers model intelligent behavior (as found in robotics, speech and language recognition, and computer animation).
Strand Five: Community, Global and Ethical Impacts	1. Discuss basic issues related to responsible use of technology and information, and the consequences of inappropriate use. 2. Identify the impact of technology (e.g., social networking, cyber bullying, mobile computing and communication, web technologies, cyber security, and virtualization) on personal life and society. 3. Evaluate the accuracy, relevance, appropriateness, comprehensiveness, and biases that occur in electronic information sources. 4. Understand ethical issues that relate to computers and networks (e.g., equity of access, security, privacy, copyright, and intellectual property).

Level 2 Middle School

Computer Science and Community

Grades 6-9

Strand One: Computational Thinking	1. Use the basic steps in algorithmic problem solving to design solutions (e.g., problem statement and exploration, examination of sample instances, design, implementing a solution, testing, and evaluation). 2. Describe the process of parallelization as it relates to problem solving.

	3. Define an algorithm as a sequence of instructions that can be processed by a computer. 4. Evaluate ways that different algorithms may be used to solve the same problem. 5. Act out searching and sorting algorithms. 6. Describe and analyze a sequence of instructions being followed (e.g., describe a character's behavior in a video game as driven by rules and algorithms). 7. Represent data in a variety of ways including text, sounds, pictures, and numbers. 8. Use visual representations of problem states, structures, and data (e.g., graphs, charts, network diagrams, flowcharts). 9. Interact with content-specific models and simulations (e.g., ecosystems, epidemics, molecular dynamics) to support learning and research. 10. Evaluate what kinds of problems can be solved using modeling and simulation. 11. Analyze the degree to which a computer model accurately represents the real world. 12. Use abstraction to decompose a problem into sub problems. 13. Understand the notion of hierarchy and abstraction in computing including high-level languages, translation, instruction set, and logic circuits. 14. Examine connections between elements of mathematics and computer science including binary numbers, logic, sets and functions. 15. Provide examples of interdisciplinary applications of computational thinking
Strand Two: Collaboration	1. Apply productivity/multimedia tools and peripherals to group collaboration and support learning throughout the curriculum. 2. Collaboratively design, develop, publish, and present products (e.g., videos, podcasts, websites) using technology resources that demonstrate and communicate curriculum concepts. 3. Collaborate with peers, experts, and others using collaborative practices such as pair programming, working in project teams, and participating in group active learning activities. 4. Exhibit dispositions necessary for collaboration: providing useful feedback, integrating feedback, understanding and accepting multiple perspectives, socialization.
Strand Three: Computing Practice and Programming	1. Select appropriate tools and technology resources to accomplish a variety of tasks and solve problems. 2. Use a variety of multimedia tools and peripherals to support personal productivity and learning throughout the curriculum.

	3. Design, develop, publish, and present products (e.g., webpages, mobile applications, animations) using technology resources that demonstrate and communicate curriculum concepts. 4. Demonstrate an understanding of algorithms and their practical application. 5. Implement problem solutions using a programming language, including: looping behavior, conditional statements, logic, expressions, variables, and functions. 6. Demonstrate good practices in personal information security, using passwords, encryption, and secure transactions. 7. Identify interdisciplinary careers that are enhanced by computer science. 8. Demonstrate dispositions amenable to open-ended problem solving and programming (e.g., comfort with complexity, persistence, brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge). 9. Collect and analyze data that is output from multiple runs of a computer program.
Strand Four: Computer and Communications Devices	1. Recognize that computers are devices that execute programs. 2. Identify a variety of electronic devices that contain computational processors. 3. Demonstrate an understanding of the relationship between hardware and software. 4. Use developmentally appropriate, accurate terminology when communicating about technology. 5. Apply strategies for identifying and solving routine hardware problems that occur during everyday computer use. 6. Describe the major components and functions of computer systems and networks. 7. Describe what distinguishes humans from machines focusing on human intelligence versus machine intelligence and ways we can communicate. 8. Describe ways in which computers use models of intelligent behavior (e.g., robot motion, speech and language understanding, and computer vision).
Strand Five: Community, Global and Ethical Impacts	1. Exhibit legal and ethical behaviors when using information and technology and discuss the consequences of misuse. 2. Demonstrate knowledge of changes in information technologies over time and the effects those changes have on education, the workplace, and society. 3. Analyze the positive and negative impacts of computing on human culture. 4. Evaluate the accuracy, relevance, appropriateness, comprehensiveness, and bias of electronic information sources concerning real-world problems.

	5. Describe ethical issues that relate to computers and networks (e.g., security, privacy, ownership, and information sharing). 6. Discuss how the unequal distribution of computing resources in a global economy raises issues of equity, access, and power.

Level 3.A – General High School Student
Computer Science in the Modern World
Grades 9-10

Strand One: Computational Thinking	1. Use predefined functions and parameters, classes and methods to divide a complex problem into simpler parts. 2. Describe a software development process used to solve software problems (e.g., design, coding, testing, verification). 3. Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. 4. Compare techniques for analyzing massive data collections. 5. Describe the relationship between binary and hexadecimal representations. 6. Analyze the representation and trade-offs among various forms of digital information. 7. Describe how various types of data are stored in a computer system. 8. Use modeling and simulation to represent and understand natural phenomena. 9. Discuss the value of abstraction to manage problem complexity 10. Describe the concept of parallel processing as a strategy to solve large problems. 11. Describe how computation shares features with art and music by translating human intention into an artifact.
Strand Two: Collaboration	1. Work in a team to design and develop a software artifact. 2. Use collaborative tools to communicate with project team members (e.g., discussion threads, wikis, blogs, version control, etc.). 3. Describe how computing enhances traditional forms and enables new forms of experience, expression, communication, and collaboration 4. Identify how collaboration influences the design and development of software products.

Strand Three: Computing Practice and Programming	1. Create and organize Web pages through the use of a variety of web programming design tools. 2. Use mobile devices/emulators to design, develop, and implement mobile computing applications. 3. Use various debugging and testing methods to ensure program correctness (e.g., test cases, unit testing, white box, black box, integration testing) 4. Apply analysis, design, and implementation techniques to solve problems (e.g., use one or more software lifecycle models). 5. Use Application Program Interfaces (APIs) and libraries to facilitate programming solutions. 6. Select appropriate file formats for various types and uses of data. 7. Describe a variety of programming languages available to solve problems and develop systems. 8. Explain the program execution process. 9. Explain the principles of security by examining encryption, cryptography, and authentication techniques. 10. Explore a variety of careers to which computing is central. 11. Describe techniques for locating and collecting small and large-scale data sets. 12. Describe how mathematical and statistical functions, sets, and logic are used in computation.
Strand Four: Computer and Communications Devices	1. Describe the unique features of computers embedded in mobile devices and vehicles (e.g., cell phones, automobiles, airplanes). 2. Develop criteria for purchasing or upgrading computer system hardware. 3. Describe the principal components of computer organization (e.g., input, output, processing, and storage). 4. Compare various forms of input and output. 5. Explain the multiple levels of hardware and software that support program execution (e.g., compilers, interpreters, operating systems, networks). 6. Apply strategies for identifying and solving routine hardware and software problems that occur in everyday life. 7. Compare and contrast client-server and peer- to-peer network strategies.

	8. Explain the basic components of computer networks (e.g., servers, file protection, routing, spoolers and queues, shared resources, and fault-tolerance). 9. Describe how the Internet facilitates global communication. 10. Describe the major applications of artificial intelligence and robotics.
Strand Five: Community, Global and Ethical Impacts	1. Compare appropriate and inappropriate social networking behaviors. 2. Discuss the impact of computing technology on business and commerce (e.g., automated tracking of goods, automated financial transactions, e-commerce, cloud computing). 3. Describe the role that adaptive technology can play in the lives of people with special needs. 4. Compare the positive and negative impacts of technology on culture (e.g., social networking, delivery of news and other public media, and intercultural communication). 5. Describe strategies for determining the reliability of information found on the Internet. 6. Differentiate between information access and information distribution rights. 7. Describe how different kinds of software licenses can be used to share and protect intellectual property. 8. Discuss the social and economic implications associated with hacking and software piracy. 9. Describe different ways in which software is created and shared and their benefits and drawbacks (commercial software, public domain software, open source development). 10. Describe security and privacy issues that relate to computer networks. 11. Explain the impact of the digital divide on access to critical information.

Level 3.B – General High School Student
Computer Science Concepts and Practices (Computer Science Principles)
Grades 10-12

Strand One: Computational Thinking	1. Classify problems as tractable, intractable, or computationally unsolvable. 2. Explain the value of heuristic algorithms to approximate solutions for intractable problems.

	3. Critically examine classical algorithms and implement an original algorithm. 4. Evaluate algorithms by their efficiency, correctness, and clarity. 5. Use data analysis to enhance understanding of complex natural and human systems. 6. Compare and contrast simple data structures and their uses (e.g., arrays and lists). 7. Discuss the interpretation of binary sequences in a variety of forms (e.g., instructions, numbers, text, sound, and image). 8. Use models and simulations to help formulate, refine, and test scientific hypotheses. 9. Analyze data and identify patterns through modeling and simulation. 10. Decompose a problem by defining new functions and classes. 11. Demonstrate concurrency by separating processes into threads and dividing data into parallel streams.
Strand Two: Collaboration	1. Use project collaboration tools, version control systems, and Integrated Development Environments (IDEs) while working on a collaborative software project. 2. Demonstrate the software life cycle process by participating on a software project team. 3. Evaluate programs written by others for readability and usability.
Strand Three: Computing Practice and Programming	1. Use advanced tools to create digital artifacts (e.g., web design, animation, video, multimedia). 2. Use tools of abstraction to decompose a large-scale computational problem (e.g., procedural abstraction, object-oriented design, functional design). 3. Classify programming languages based on their level and application domain 4. Explore principles of system design in scaling, efficiency, and security. 5. Deploy principles of security by implementing encryption and authentication strategies. 6. Anticipate future careers and the technologies that will exist. 7. Use data analysis to enhance understanding of complex natural and human systems.

	8. Deploy various data collection techniques for different types of problems.
Strand Four: Computer and Communications Devices	1. Discuss the impact of modifications on the functionality of application programs. 2. Identify and describe hardware (e.g., physical layers, logic gates, chips, components). 3. Identify and select the most appropriate file format based on trade-offs (e.g., accuracy, speed, ease of manipulation). 4. Describe the issues that impact network functionality (e.g., latency, bandwidth, firewalls, and server capability). 5. Explain the notion of intelligent behavior through computer modeling and robotics.
Strand Five: Community, Global and Ethical Impacts	1. Demonstrate ethical use of modern communication media and devices. 2. Analyze the beneficial and harmful effects of computing innovations. 3. Summarize how financial markets, transactions, and predictions have been transformed by automation. 4. Summarize how computation has revolutionized the way people build real and virtual organizations and infrastructures. 5. Identify laws and regulations that impact the development and use of software. 6. Analyze the impact of government regulation on privacy and security. 7. Differentiate among open source, freeware, and proprietary software licenses and their applicability to different types of software. 8. Relate issues of equity, access, and power to the distribution of computing resources in a global society.

Level 3.C – General High School Student

Topics in Computer Science

1. **AP Computer Science A**
2. **Projects-Based Courses**
    a. Desktop Publishing
    b. Technical Communications
    c. Multimedia
    d. Graphics
    e. Game Programming
    f. Web Development

g. Web Programming
h. Emergent Technologies
i. Free and Open Source Software (FOSS) Development

3. Courses Leading to Industry Certification

a. A+ Certified Technician
b. Quick Security+
c. Certified Internet Webmaster (CIW)

APPENDIX B
CASE STUDY ONE: DETIALED LIST OF BADGES

Badge Name	Required Tasks	Category	Badge Type	Required? (Learning Objective)	Points Awarded
You Win the Internet!	Completed all Missions (required Quests and Boss Fights)	Mastery	Quest Badge	Yes	1000
Mission Completed!	Completed any Mission	Mastery	Quest Badge	Yes	200
Master of Abstraction	Completed all *Abstraction* & *Data* Quests Completed the *Abstraction* & *Data* Boss Fight	Abstracting	Quest Badge	Yes	50
Data Abstractor	Completed all *Data* Quests	Abstracting	Quest Badge	Yes	225
Abstract Control Model	Completed all *Abstraction* Quests	Abstracting	Quest Badge	Yes	225
AI Analyst	Completed all *Algorithms* Quests Completed *Algorithms* Boss Fight	Analyzing Computational Artifacts	Quest Badge	Yes	500
A Worldly Mind	Completed all *Impact* Quests	Collaborating	Quest Badge	Yes	500
A Series of Tubes	Completed all *Internet* Quests	Communicating	Quest Badge	Yes	500
A Creative Creator of Creativity	Completed all *Creativity* Quests	Connecting Computing	Quest Badge	Yes	500
|-|4x0rz	Completed all *Programming* Quests	Developing Computational Artifacts	Quest Badge	Yes	500
Epic Quest Completed!	Completed Epic *Proposal*, Epic *Artifact*, & Epic *Document*	Semester Project	Quest Badge	Yes	300
A Proposal of Epic Proportions	Completed the Epic Quest *Proposal*	Semester Project	Quest Badge	Yes	100
An Epic Artifact	Completed the Epic *Artifact*	Semester Project	Quest Badge	Yes	50
An Epic Document	Completed the *Written Report*	Semester Project	Quest Badge	Yes	50
Boss Defeated!	Completed any quiz	Boss Fight	Quest Badge	Yes	50
Hello, World!	Registered for the course website	All	Community Badge	Yes	10
What's Up?	Changed avatar picture	Autonomy	Community Badge	No	10
Journalist	Created a new topic in forums	Purpose	Community Badge	No	25
"I have not failed…"	Failed any quiz	Mastery	Community Badge	No	75
+1 for Creativity	Submission demonstrating extra creativity	Autonomy	Community Badge	No	50

+1 for Extra Effort	Submission demonstrating above and beyond quality	Mastery	Community Badge	No	50
Over-Achiever	Completed any Side Quest	Autonomy	Community Badge	No	125
Nobody asked, but…	Posted a comment on the website	Purpose	Community Badge	No	30
Active Learner	Rewarded for active participation in community	Purpose	Community Badge	No	75
Commentator	Logged into website 10 times & commented 5 times on other members' posts	Purpose	Community Badge	No	75
Community Activist	Rewarded for earning any 3 community badges	Mastery	Community Badge	No	75
Find a Pick-up-Group (PUG)	Joined a group on the website	Purpose	Community Badge	No	50
Group Chaperone	Invited a friend to a group on the website	Purpose	Community Badge	No	25
Innovator	Nominated a classmate for excellent work	Mastery	Community Badge	No	100
Looking for Group (LFG)	Created a group on the website	Purpose	Community Badge	No	100
Shout-out to the world!	Updated activity stream once	Autonomy	Community Badge	No	10
Status Update	Updated activity stream 5 times	Autonomy	Community Badge	No	75

APPENDIX C
CASE STUDY ONE: PROXY PRE-TEST AND POST-TEST

EXTRA CREDIT SURVEY FOR ATLS/CSCI-1220-001

PARTICIPANT INFORMED CONSENT

Thank you for taking the time to complete this survey!
By completing this survey, you will help give the instructor insight into your experiences as a student in ATLS/CSCI 1220 Virtual Worlds: Introduction to Computer Science. Your responses to these questions will help improve the course for future students who take this class.

The survey will ask you questions regarding your experiences in computer programming, video gaming, and your experiences as a student in ATLS/CSCI 1220. The questionnaire will take you approximately 10 to 15 minutes to complete. Although absolute anonymity cannot be guaranteed over the Internet, we will ensure your privacy is kept confidential by collecting data on a private web-server hosted by SurveyGizmo.com. Any information that would permit identification of individuals will be kept strictly confidential. Should the data from this survey be published for educational research purposes, no individual information about you will be disclosed.

Your participation in this study is voluntary. By agreeing to voluntarily participate in this survey, you will gain 2% extra credit toward your final grade for CSCI/ATLS-1220 in Fall 2014. You can choose to stop the survey at any time by closing your Web browser window.

If you have any questions or concerns about completing this questionnaire, you may contact the instructor, Kara A. Behnke, directly at kara.consigli@colorado.edu. By checking "I agree" to the terms and conditions for taking this study, and clicking the "Next" button below, you indicate that you have read and agreed to the disclosure statement above and that you are voluntarily agreeing to participate in this survey.*
[ ] I agree to the terms and conditions of taking this survey.

STUDENT DEMOGRAPHICS

(Q1) Please fill out your name: (This information will only be used to give you extra credit for ATLS/CSCI-1220)*

_____

(Q2) What is your gender?*
( ) Male
( ) Female
( ) Transgender
( ) Genderfluid
( ) Prefer not to disclose

(Q3) What racial and/or ethnic group(s) do you most identify with?*
[ ] American Indian or Alaska Native
[ ] Black or African American
[ ] East Asian or Pacific Islander
[ ] Latino or Hispanic
[ ] Middle Eastern or Central Asian

[ ] White or Caucasian
[ ] Other: _____
[ ] Prefer not to disclose

(Q4) What is your current standing as a student this semester (Fall, 2014) at CU Boulder?*
( ) Freshmen (1st year)
( ) Sophomore (2nd year)
( ) Junior (3rd year)
( ) Senior (4th year)
( ) 5th Year Senior (or above)
( ) Auditor (Unclass)
( ) Other

(Q5) What is your declared major of study? (e.g., Open-option, Computer Science, Integrated Physiology, etc.)*

_____

PREVIOUS EXPERIENCE WITH COMPUTER SCIENCE & PROGRAMMING

(Q6) Do you currently own a personal computer?*
( ) Yes.
( ) No.

(Q7) Do you currently own a "smart phone"?*
( ) Yes.
( ) No.

(Q8) Which of the following PC operating systems are you familiar with?
[ ] Mac OS X (version 10.7 or above)
[ ] Mac OS X (version 10.6 or below)
[ ] Windows 8 or 8.1
[ ] Windows 7
[ ] Windows Vista
[ ] Windows XP
[ ] Linux (Ubuntu, Mint, etc.)
[ ] Rasbian (Raspberri Pi)
[ ] Other

(Q9) Which of the following programming languages have you used before this class?
[ ] C family (C++, C#, Objective-C, C, etc.)
[ ] OOP (Java, VisualBasic, etc.)
[ ] Web (HTML, PHP, Javascript, Ruby, JQuery, CSS, ActionScript, etc.)
[ ] Scripting (Python, Matlab, Perl, etc.)
[ ] Visual (Scratch, AgentSheets, etc.)
[ ] Other (Gaming Macros etc.)
[ ] None of the above

(Q10) Have you taken a course related to computer science or programming before college (e.g., high school, middle school, etc.)?*
( ) Yes.
( ) No.
( ) I don't know.

(Q11) If you have taken a course related to computer science before college, when did you take it? (e.g., high school, middle school, etc.)

_____

(Q12) Before taking ATLS/CSCI-1220, have you taken a computer science or programming course in college?
( ) Yes
( ) No
( ) I don't know

(Q13) Before taking this class, did you think computer science was a creative practice?
( ) Yes
( ) No
( ) Somewhat
( ) I don't know, here's why:: _____

(Q14) During your enrollment in ATLS/CSCI-1220 this fall, were you also enrolled in another computer science or programming course here at CU?
( ) Yes
( ) No
( ) I don't know

(Q15) Before taking this class, have you ever built a website?
( ) Yes
( ) No

(Q16) Before taking this class, have you ever built a desktop computer?
( ) Yes
( ) No

(Q17) Before taking this class, have you ever developed a video game?
( ) Yes
( ) No

(Q18) Before taking this class, have you ever developed a mobile app?
( ) Yes
( ) No

(Q19) Before taking this class, how would you rate your programming skills?
No Experience
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Advanced Experience

(Q20) Before taking this class, how would you rate your level of interest in computer science?*
( ) Not at all interested ( ) Slightly interested ( ) Moderately interested ( ) Very interested ( ) Extremely interested

---

EXPERIENCE WITH GAMING

(Q21) Do you actively play video games? ("video games" includes any console games, computer or PC games, virtual worlds, mobile apps, social-media games, etc.)*
[ ] Yes, I actively play video games.
[ ] No, I don't play video games at all.
[ ] I used to play video games, but not anymore.
[ ] I sometimes play video games.

(Q22) If you play video games, approximately what age did you first begin playing?

_____

(Q23) During an average week during the semester, how many hours do you spend playing video games? ("video games" includes any console games, computer or PC games, virtual worlds, mobile apps, social-media games, etc.)*

( ) None.
( ) Less than 1 hour per week.
( ) 1-5 hours per week.
( ) 5-10 hours per week.
( ) 10-15 hours per week.
( ) 15 or more hours per week.

(Q24) During an average week during the summer, how many hours do you spend playing video games? ("video games" includes any console games, computer or PC games, virtual worlds, mobile apps, social-media games, etc.)*

( ) None.
( ) Less than 1 hour per week.
( ) 1-5 hours per week.
( ) 5-10 hours per week.
( ) 10-15 hours per week.
( ) 15 or more hours per week.

(Q25) I consider myself to be a gamer.*

( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q26) Please list any video game titles that you currently play (e.g., Destiny, League of Legends, Candy Crush, etc.). Type 'none' if you do not currently play any games.*

_____
_____

(Q27) Which of the following gaming platforms have you played on before (e.g., played at your house, at a friend's house, at a library, etc.)?*

[ ] Atari
[ ] Odyssey
[ ] Sega Genesis
[ ] Dreamcast
[ ] PC (e.g., CDs or DVDs)
[ ] Gameboy, Gameboy Color
[ ] Nintendo DS, 3DS
[ ] Nintendo (NES)
[ ] Super Nintendo (SNES)
[ ] Nintendo 64 (N64)
[ ] Nintendo Gamecube
[ ] Nintendo Wii
[ ] Nintendo WiiU
[ ] Xbox
[ ] Xbox 360
[ ] Xbox One
[ ] Playstation
[ ] Playstation 2
[ ] Playstation 3
[ ] Playstation 4
[ ] PSP
[ ] Playstation Vita
[ ] Smart phone (iPhone, Droid, Windows Phone)
[ ] Tablet (iPad, Surface, Nexus, etc.)
[ ] Gaming Emulators

[ ] Social Media (Facebook, Zynga, etc.)
[ ] Websites (Flash games, League of Legends, etc.)
[ ] Online Gaming Marketplaces (Steam, Battle.net, etc.)
[ ] None of these
[ ] Other: _____

(Q28) Which of the following gaming platforms have you owned or subscribed to before?*
[ ] Atari
[ ] Odyssey
[ ] Sega Genesis
[ ] Dreamcast
[ ] PC (e.g., CDs or DVDs)
[ ] Gameboy, Gameboy Color
[ ] Nintendo DS, 3DS
[ ] Nintendo (NES)
[ ] Super Nintendo (SNES)
[ ] Nintendo 64 (N64)
[ ] Nintendo Gamecube
[ ] Nintendo Wii
[ ] Nintendo WiiU
[ ] Xbox
[ ] Xbox 360
[ ] Xbox One
[ ] Playstation
[ ] Playstation 2
[ ] Playstation 3
[ ] Playstation 4
[ ] PSP
[ ] Playstation Vita
[ ] Smart phone (iPhone, Droid, Windows Phone)
[ ] Tablet (iPad, Surface, Nexus, etc.)
[ ] Gaming Emulators
[ ] Social Media (Facebook, Zynga, etc.)
[ ] Websites (Flash games, League of Legends, etc.)
[ ] Online Gaming Marketplaces (Steam, Battle.net, etc.)
[ ] None of these
[ ] Other: _____

(Q29) I enjoy playing video games in my spare time.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q30) How often did you play video games as a child? (including console games, PC, arcade, etc.)?*
( ) Never ( ) Seldom ( ) Sometimes ( ) Often ( ) Frequently

(Q31) Compared to five years ago:
( ) I play video games more frequently now.
( ) I play video games less frequently now.
( ) There has been little change in the frequency of my video game playing.

(Q32) I like to do the following: (Please select any that apply)*
[ ] Watch family members play games at home (e.g., I'm physically present to watch siblings or parents play games)
[ ] Watch other people play games online (e.g., I watch other people play games via Twitch or YouTube)
[ ] Watch my friends play games in-person (e.g., I am physically present to watch my friends play games, but I do not necessarily play myself)
[ ] Play games by myself without anyone else around (e.g., single-player experiences without anyone else present)
[ ] Play games by myself with others physically present (e.g., single-player experiences with others watching me play)

[ ] Play games by myself and upload video content to Twitch or YouTube so others can watch me play
[ ] Play games with my friends in-person (e.g., play physically present with others)
[ ] Play games with my friends online (e.g., play with friends over the Internet)
[ ] Play games with my family in-person (e.g., play physically present with family)
[ ] Play games with my family online (e.g., play with family over the Internet)
[ ] Play games at an arcade
[ ] Play games at a library
[ ] Play games at a bar
[ ] Play games at a tea/coffee shop
[ ] Play games at home or in my dorm
[ ] Play games on campus
[ ] None of these
[ ] Other: _____

EXPERIENCES WITH THE COURSE & WEBSITE

(Q33) After taking this class, how would you rate your programming skills?
No Experience
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Advanced Experience

(Q34) After taking this class, how would you rate your level of interest in computer science?*
( ) Not at all interested ( ) Slightly interested ( ) Moderately interested ( ) Very interested ( ) Extremely interested

(Q35) After taking this class, do you think computer science is a creative practice?
( ) Yes
( ) No
( ) Somewhat
( ) I don't know, here's why:: _____

(Q36) After taking this class, I will enroll in additional computer science classes here at CU Boulder.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q37) After taking this class, I have a better understanding of how computing systems work.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q38) This course broadened my general perspective of what computer science involves.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q39) Overall, I feel that the content of this course was of high quality.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q40) The learning resources (e.g., website resources, readings, Internet tools, etc.) were valuable to me.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q41) The assignment activities allowed me to demonstrate what I understood.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q42) The midterm examination allowed me to demonstrate what I understood.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q43) I was motivated to learn in this course.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q44) The course website was easy to use.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q45) I prefer flexible homework deadlines rather than strict homework deadlines.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q46) The course website provided a sense of community.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q47) The badges and points on the website were confusing.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q48) The green progress bar was useful for completing assignments throughout the semester.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q49) The gaming elements of the course (e.g., Quests, Missions, Boss Fights etc.) made sense to me.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q50) The gaming elements of the course (e.g., Quests, Missions, Boss Fights etc.) made the course more interesting.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q51) I could easily collaborate with other students by using the course website.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q52) I liked the use of a leaderboard on the website.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q53) The leaderboard on the website brought a sense of competition to the course.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q54) I liked collecting badges on the website.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q55) Overall, I disliked the self-paced structure of the course.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q56) Raising my rank on the leaderboard helped motivate me to complete coursework.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q57) Collecting badges motivated me to do things I otherwise would not have done on the course website.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q58) Overall, I feel that the workload for a 4-credit course was appropriate.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q59) The lectures helped me better understand the concepts covered in the assignments.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q60) What did you LIKE about ATLS/CSCI-1220 as a course? Select all that apply.*
[ ] The topics presented in lecture
[ ] Submitting assignments whenever I want
[ ] The use of gaming elements to structure the course (e.g., quests as homework, badges, leaderboard etc.)
[ ] The topics of the assignments and reading materials
[ ] Having an open-ended semester project
[ ] Online discussions (e.g., forums, comments etc.)
[ ] Using the course website instead of Desire2Learn (D2L)
[ ] Submitting all of the assignments online
[ ] Collecting badges and points on the course website
[ ] No final examination
[ ] Using external Internet tools to complete assignments (e.g., Lightbeam, TOR, etc.)
[ ] Playing video games (e.g., Lightbot, Super Mario, etc.)
[ ] Guest-speakers

[ ] The types of programming assignments
[ ] Having a personal profile on the course website
[ ] Taking the online quizzes as many times as I want
[ ] Getting recognized for creativity or extra effort
[ ] Having a progress bar to indicate coursework completion
[ ] How I received feedback from the TA and/or instructor
[ ] None of the above
[ ] Other: _____

(Q61) Which of the following would you CHANGE about the ATLS/CSCI-1220 course? Select all that apply.*
[ ] No gaming elements
[ ] More gaming elements
[ ] Add a required textbook
[ ] Lessen amount of assignments
[ ] Increase amount of assignments
[ ] Add strict deadlines to assignments, projects, etc.
[ ] More specific requirements for the semester project
[ ] Add a recitation section to the course
[ ] Scheduled time as a class in the computer laboratory
[ ] More support available (TA time)
[ ] Lectures more focused on programming
[ ] Lectures less focused on programming
[ ] More guest-speakers
[ ] Less guest-speakers
[ ] Final examination instead of semester project
[ ] Less assignments focused on programming
[ ] More assignments focused on programming
[ ] Cover fewer topics over the semester
[ ] Cover more topics over the semester
[ ] None of the above
[ ] Other: _____

(Q62) Did your experiences in this course convince you to declare or change your major to computer science here at CU Boulder?
( ) Yes
( ) No
( ) Maybe
( ) I was already majoring in CS when I enrolled in this course
( ) I don't know, here's why:: _____

(Q63) Did your experiences in this course convince you to take additional courses in computer science here at CU Boulder?
( ) Yes
( ) No
( ) Maybe
( ) I don't know, here's why:: _____
( ) I was already majoring in CS when I enrolled in this course so I have to take more CS classes anyway

(Q64) Do you plan to learn more about programming on your own (e.g., take classes via CodeAcademy or KhanAcademy, online tutorials, tinkering, etc.)?
( ) Yes
( ) No
( ) Maybe
( ) I don't know, here's why:: _____

(Q65) Please describe in detail what you liked about this course below.*

_____
_____

(Q66) Please describe in detail what you disliked about this course below.*

_____
_____

ATTITUDES ABOUT COMPUTER SCIENCE & PROGRAMMING

(Q67) I like using computing tools to solve problems.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q68) I can effectively learn how to understand computing concepts by experimenting on my own.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q69) I can effectively learn how to understand computing concepts by taking computer science classes.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q70) I can effectively learn how to understand computing concepts by using online resources.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q71) I think programming is useful for other academic and professional disciplines besides just computer science.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q72) I perform poorly in mathematics courses.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q73) Knowledge of computer science will allow me to secure a good job.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q74) Having programming skills will allow me to secure a good job.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q75) The challenge of solving problems using computer science does not appeal to me.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q76) I can effectively analyze the ethical, legal, and social implications of computing.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q77) I can write successful computer programs using helpful resources.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q78) Having a broad understanding of computer science is beneficial for my life.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q79) I cannot write successful computer programs on my own.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q80) I can effectively use online search tools to solve a problem.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q81) You need to be good at math in order to learn computer science.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q82) I can effectively use abstractions and models to achieve goals.*

( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q83) Everyone should have a basic understanding of how computing systems work.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

---

THANK YOU!

APPENDIX D
CASE STUDY ONE: ONE YEAR FOLLOW-UP SURVEY

1-YEAR FOLLOW-UP WITH ATLS/CSCI-1220 STUDENTS

INTRODUCTION TO SURVEY

You are being asked to take this survey because you enrolled as a student in ATLS/CSCI-1220 "Virtual Worlds: Introduction to Computer Science" during the Fall 2014 academic semester. By completing this survey, you will help give the instructor, Kara A. Behnke, insight into your experiences and attitudes about computer science since your time enrolled in the course.

Your responses to these questions will help improve computer science education at the University of Colorado Boulder. The questionnaire will take you approximately 5 to 10 minutes to complete. Although absolute anonymity cannot be guaranteed over the Internet, we will ensure your privacy is kept confidential. Any information that would identify you will be kept strictly confidential and no individual information about you will be disclosed.

You participation in taking this study is voluntary. For completing this survey, you will be entered into a raffle for a chance to win a $20 gift card to Starbucks. If you have any questions or concerns about completing this survey, please contact Kara A. Behnke directly at kara.consigli@colorado.edu.

By checking the "I agree" and clicking the "Next" button, you indicate that you have read and agreed to the disclosure statement above and that you are voluntarily agreeing to participate in this survey.*
[ ] I agree to the terms and conditions of taking this survey.

1-YEAR SINCE 1220!

(Q1) What is your name? (Your name will not be used for any reason except for verifying your enrollment in ATLS/CSCI-1220 last Fall. Your personal information will be kept safe, confidential, and will not be disclosed without your consent).*

_____

(Q2) Since your enrollment in ATLS/CSCI-1220, have you enrolled in or taken any additional Computer Science or programming courses at CU Boulder in the last year, such as CSCI-1300, ATLS-2519, etc.?*

( ) Yes
( ) No
( ) I'm not sure

(Q3) Please describe the Computer Science or programming course you enrolled into, when you took this course, and please describe your overall experience as a student in this course.*

_____
_____

(Q4) If you are unsure, please describe the course you enrolled into, when you took this course, and please describe your overall experience as a student in this course.*

_____

_____

(Q5) Since your enrollment in ATLS/CSCI-1220, have you participated in any free online courses or programming tutorials in the last year, such as Khan Academy, Coursera, Code.org., etc.?*

( ) Yes
( ) No
( ) I'm not sure

(Q6) Please describe the online course or website tutorials you used, approximately how long you participated, and please describe your overall experience of these online activities.*

_____
_____

(Q7) Since your enrollment in ATLS/CSCI-1220, in the last year have you learned more about computer science or programming through hands-on experiences, such as participating in a hack-a-thon, a programming workshop, learning through a summer internship, etc.?*

( ) Yes
( ) No
( ) I'm not sure

(Q8) Please describe the hands-on learning experience you engaged in, approximately how long you participated, and please describe your overall experience while participating in these activities.*

_____
_____

(Q9) After taking ATLS/CSCI-1220, did you change your major to Computer Science? *

( ) Yes
( ) No
( ) I was already majoring in CS
( ) I wanted to but couldn't because I'm close to graduating
( ) Other - Write In: _____

(Q9.5) Please explain why you changed your major to Computer Science.*

_____
_____

Please rate how much you agree or disagree with the following statement:
(Q10)"After taking ATLS/CSCI-1220 last Fall, I have a better understanding of how technology works in the real world."*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

Please rate how much you agree or disagree with the following statement:
(Q11) "After taking ATLS/CSCI-1220 last Fall, I do not want to learn anything more about programming or Computer Science."*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

Please rate how much you agree or disagree with the following statement:
(Q12) "I think programming is a useful skill for many fields of work, not just for Computer Science."*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

Please rate how much you agree or disagree with the following statement:
(Q13) "Taking ATLS/CSCI-1220 last Fall helped show me that Computer Science is a creative field and practice."*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

(Q14) What was the most memorable experience (good or bad) during your time as a student in ATLS/CSCI-1220 Fall 2014?*

_____
_____

THANK YOU!

APPENDIX E
CASE STUDY ONE: CODE BOOK

MIDTERMS

1. Attitudes toward Computer Science
    a. Fun
    b. Interesting
    c. Exciting
    d. Like
    e. Cool
    f. Enjoyable
    g. Good
    h. Useful
    i. Beneficial
    j. Important
    k. Doable
    l. None
    m. Afraid
    n. Hard
    o. Confusing
    p. Nerd
    q. Bad
    r. Distance of Self
    s. Subset of Population
    t. Change
    u. No change

2. References
    a. Major
    b. Job
    c. Family
    d. Friend
    e. Website
    f. Previous Experience
    g. Important Person
    h. Hobbies
    i. Using Computers
    j. Other Fields

3. Accountable Disciplinary Knowledge
    a. Math
    b. Jargon
    c. Code
    d. Previous Experience
    e. Change

f. No Change
g. Concepts
h. Sub-discipline
i. Uses
j. Capabilities
k. Online Class
l. Homework
m. Self-Work
n. Hardware
o. Fluency
p. Broad
q. Interest
r. Humanities
s. Impact
t. Powerful

4. Recommend
    a. Yes
    b. No
    c. Eh (neutral)
    d. You
    e. Specialize
    f. Everybody
    g. Other Students
    h. People
    i. Society
    j. CS Major
    k. CS Interest
    l. TAM (Technology, Arts & Media)
    m. General Population

5. Lectures
    a. Relevant
    b. Interesting
    c. Informative
    d. Lots of Work
    e. More Code
    f. Useful
    g. Beneficial
    h. Introductory CS
    i. Computers
    j. Code
    k. Society

6. Definitions
    a. Code
    b. Math

c. Logic
d. Not Code
e. Better Than Other Fields
f. Creative
g. Sub-discipline
h. Uses
i. Capabilities
j. Hardware
k. Fluency
l. CS Lite
m. Impact
n. Other Fields
o. Intro
p. Relative
q. Society

7. Course Structure
    a. Website
    b. Deadline
    c. Self-Directed
    d. Situated Learning
    e. Scaffolding
    f. Connectivism
    g. Freedom
    h. Redemption
    i. Game Elements
    j. Constructionist
    k. Competition
    l. Grade on Effort
    m. Challenge
    n. Responsibility
    o. Play
    p. Exploration
    q. Experiential Learning
    r. Self-Reflections

8. Course Content
    s. Breadth
    t. Types of Assignments
    u. Creativity
    v. Programming
    w. Quests
    x. Game Design
    y. Interesting
    z. Purpose
    aa. CS-Lite
    bb. Lectures

cc. Enjoyment
dd. Competency
ee. Balance
ff. Non-Traditional
gg. More
hh. Fun
ii. Flexibility
jj. Relevancy
kk. Deadlines
ll. Too Much Reading
mm. Feedback
nn. Challenging
oo. Community
pp. Depth Over Breadth

APPENDIX F
MIDTERM EXTRA-CREDIT SURVEY QUESTIONS

1. Before taking this course, what was your general attitude about computer science?
2. Generally speaking, has this course changed your perception of computer science? Why or why not?
3. What do you like and/or dislike about this course?
4. Would you recommend this course to other students? Why or why not?

APPENDIX G
CASE STUDY TWO: TECHNOVAITON JUDGING RUBRIC

DIRECTIONS: The judging rubric for Technovation 2015 contains objective and subjective scoring. The objective score composes of the Ideation, Technical, and Entrepreneurial scores. It is possible for all teams in your scoring batch to receive a perfect score on these objective scoring. The Overall Impression score is where you should score the apps in your batch relative to each other for a subjective score. Maximum points on the subjective part should only be awarded if the submission has truly exhausted their research and development for a complete prototype.

Objective Score

Ideation Score	**0**	**1**	**3**	**5**	**SCORE**	**COMMENTS**
Did the team identify a real problem in their community?	No	The problem identified is more of a nuisance, and does not have larger social implications.	The problem seems to be real, but could use a little more detail.	Yes		
Does the app address the problem that they identified?	No	The app addresses a tangential problem.	The app addresses some parts of the problem.	Yes		
				10 maximum possible points		
Technical Score	**0**	**1**	**3**	**5**	**SCORE**	**COMMENTS**
Is the prototype they submitted fully functional? (All buttons and links functional and no obvious bugs.)	No	Only superficial functionality. (i.e. screen transitions)	Mostly, except for a few minor issues. I can still get the general idea.	Yes, no bugs that I could see.		
Does the prototype go beyond static information? (e.g. calls another app on the phone, saves information to an external server to	No, only information stored directly in the app is used.	The app uses other resources, but it seems unnecessary for the purpose of the app.	Mostly, but there are one or two other places where the app could have used an external service to be more effective.	Yes, the app uses multiple resources and does so effectively.		
Does the prototype match the feature list defined in their product description?	No	Less than half of the features listed are in the prototype and minimal explanation for why features are missing.	More than half of the features are in the prototype, and there's a reasonable explanation for the missing features.	Yes, everything stated is included.		
User Interface. Is the UI intuitive and easy to use?	No	All the functionality is there, but I had to watch the demo video to and read the product description to understand the app.	The app was obvious after thoroughly reading the product description.	A quick skim of the defined problem and product description was enough to understand how to use the app.		
				20 maximum possible points		

	Entrepreneurial score (Business Plan and Pitch Video - broken down into its components)	**0**	**1**	**2**	**3**	**SCORE**	**COMMENTS**
	Product Description	None	A short and vague description.	Describes the app, but value of the app is vague.	Describes app and value added.		
	Potential Market Size	None	Groups of people mentioned, but no estimates done.	Estimates done and some groups defined. Could have been more thorough.	Estimates done and groups defined. Estimates clearly explained.		
	Competitive Analysis	None	Competitors are named, but explanations are sparse or nonexistent.	Competitors are named, and explanations are provided. Could have been more thorough.	Analysis is exhaustive.		
	Potential Revenue	None	Calculations are suspect, and explanations are unclear.	Calculations exist, but basis of the calculations could use more explanation.	Calculations and explanations are thorough and believable.		
	Branding and Promotion	None	A logo or promotional sources are included, but explanations are sparse or nonexistent.	Logo and limited promotional sources are included. Explanations for promotional plan could be more thorough.	Logo and promotional plans are included, well explained and exhaustive.		
					15 maximum possible points		
SubjectiveScore	**Overall Impression -** Award points based on which ones gave the best pitch in your batch.	**0**	**2**	**5**	**10**		**COMMENTS**
SubjectiveScore	**Overall Pitch Quality. Is the Pitch compelling? Can you see this app being used by consumers?**	Not at all. Argument was flawed and difficult to follow.	I understand their argument, but I'm not sold.	Compelling arguments were made, and a small following may form.	Yes! The argument was compelling enough to think this app has true growth potential.		
SubjectiveScore					10 Maximum total points		
TotalScore	**Deliverables (subtract 1 point for each item missing)**	Pitch Video	Demo Video	Prototype Source code / Screenshots	100-word app descrip / Business plan		
TotalScore	**Bonus points: Has the app already been launched the Play Store or App Store? (2 possible bonus points)**						
TotalScore	**TOT**				**55 maximum total points**		

APPENDIX H
CASE STUDY TWO: PRE-TEST & POST-TEST SURVEY

TECHNOVATION ENROLLMENT SURVEY

PRE-TEST SURVEY

1) Please fill-out your name:*

_____

2) I am a hard worker.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

3) I am confident using technology.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

4) I know about the design process used to create technology products.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

5) I am diligent (hard working and careful) in my work.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

6) I know what a prototype is.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

7) I incorporate my knowledge of the consumer into the products that I develop.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

8) I am confident talking about business models.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

9) I know how technology products are marketed.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

10) I know about user interface design.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Disagree

11) Setbacks and obstacles don't discourage me. I bounce back from disappointments faster than most people.*
( ) Strongly Disagree ( ) Agree ( ) Neutral ( ) Agree ( ) Strongly Agree

12) I finish whatever I begin.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

13) I know how to develop code for an app.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

14) I often set a goal but later choose to follow a different one.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

15) I am comfortable making presentations about my school projects.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

16) I am taking or plan to take advanced classes in math and science.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

17) I am interested in starting my own technology company.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

18) I am considering a career in technology.*
( ) Strongly Disagree ( ) Disagree ( ) Neutral ( ) Agree ( ) Strongly Agree

19) What kind of job/career do you want to have when you are older? (Please select only one answer).*
( ) Technology

( ) Business entrepreneurship

( ) Writing

( ) Science

( ) Art

( ) Engineering

( ) Don't know

( ) Other (please specify): _____

20) Please tell us which best describes you. (Please check all that apply).*
[ ] American Indian / Alaska Native

[ ] African American / Black

[ ] Asian

[ ] Hispanic / Latina

[ ] Native Hawaiian / Other Pacific Islander

[ ] White

[ ] I choose not to respond

[ ] Other (please specify): _____

21) Which of these applications do you use regularly? (Please check all that apply)*
[ ] Facebook

[ ] Twitter

[ ] Snapchat

[ ] Tumblr

[ ] LinkedIn

[ ] Instagram

[ ] None

[ ] Other (please specify): _____

Thank You!

APPENDIX I
CASE STUDY TWO: EXPERIENCE SAMPLING METHOD SURVEY

Please answer the following questions about the activities you did today in Technovation.

	Totally disagree	Somewhat disagree	Neutral (neither agree nor disagree)	Somewhat agree	Totally agree
I think the activities today were really interesting.	○	○	○	○	○
I enjoyed doing the activities today.	○	○	○	○	○
I had to concentrate really hard when doing the activities.	○	○	○	○	○
When doing the activities today, I didn't think about anything else.	○	○	○	○	○
The activities today were challenging.	○	○	○	○	○
Doing these activities is important.	○	○	○	○	○
I figured out how to solve a tough problem during today's activities.	○	○	○	○	○
The activities felt more like homework than play.	○	○	○	○	○
I was learning a lot when doing the activities.	○	○	○	○	○
The activities were fun to do.	○	○	○	○	○
I had enough time to finish today's activities.	○	○	○	○	○
I prefer doing these activities in a group than by myself.	○	○	○	○	○
I could have put in more effort into completing today's activities.	○	○	○	○	○

(Optional)

Any comments or suggestions to improve the activities today or Technovation in general?

_____

APPENDIX J
CASE STUDY TWO: FOCUS GROUP SCRIPT

FOCUS GROUP(S) INTERVIEW INSTRUMENT

Below is the protocol for student focus groups for the research being conducted under the auspices of the Principal Investigator (PI), Kara A. Behnke, a doctoral student in the ATLAS Institute at the University of Colorado at Boulder, 320 UCB, Boulder, CO 80309-0270. This project is being done under the direction of Dr. John K. Bennett, Associate Vice Chancellor for Innovation Initiatives, Professor of Computer Science & Engineering, CU Denver; Professor of Computer Science, CU Boulder (on leave).

FOCUS GROUP SCRIPT

Moderator actions will be *highlighted in italics*. Moderator script will be **highlighted in bold.**

Good morning/afternoon! Thank you for taking the time to join our discussion about Technovation and the website we are using for this program.

As you know, I'm the ____RESEARCH COORDINATOR__ of this study. I am conducting research on effective teaching strategies for the Technovation. I hope to gain insight from your experiences as participants so far in the program so that I can teach it better in the future. So, I'll be asking you some questions about your experiences so far in Technovation and using the website to learn.

It is important that you speak up and that you only speak one at a time. We promote a community of respect and consideration here at [non-profit community center], so please respect your peers when it is their turn to speak.

I will only use your first names here. When I write-up this interview later, I won't use your real name or other information that may identify you. Your privacy and information will be kept safe. I want you to feel safe when speaking your mind, so please know that what you say won't get you in trouble (unless what you say harms you or somebody else). Please don't repeat who said what here when you leave this room.

For the next 20 minutes or so, I will ask you some questions, and I will listen to what you have to say. Please, feel free to respond to each other and to speak directly to others in the group.

I want to hear from all of you. I am interested in all of your opinions and experiences about Technovation and using the website for this program.

If it is okay with all of you, we will turn on the recorder and start now.

This student focus group is being conducted for Kara Behnke's dissertation study on [DATE] by __MODERATOR___.
Video/audio recording ID number: _____.
FOCUS-GROUP START TIME: _____.

INTRODUCTIONS (FIRST PHASE)
Moderator actions will be *highlighted in italics*. Moderator script will be **highlighted in bold.**

Let's begin with introductions.

1). **Why did you choose to enroll in Technovation?**

2). **How comfortable are you with using technology?** *MODERATOR NOTES - MAKE A LIST OF TECHNOLOGY USED*

3). **Have any of you taken a technology class before signing up for Technovation?**

4). **Before signing up for Technovation, did you think you were capable of creating technology or a business? Why or why not?**
MODERATOR NOTES - MAKE A LIST OF ATTITUDES OF TECH/BUSINESS

5). **How do you like the Technovation program so far?**
CONTINUE LIST: LIKES/DISLIKES

SUMMARIZE: **It looks like there were (quite a few/some) positive aspects of _____ that motivated you to learn more about computer science. These features include:** *NAME CATEGORIES*.

WEBSITE EXPERIENCES (SECOND PHASE)

Now I would like to ask you all some questions about your experiences specifically using the website that we use for Technovation.

1). **Is the website easy to use?**
FOLLOW-UP WITH EMERGENT QUESTIONS

2). **What sorts of things on the website were challenging for you?**
FOLLOW-UP WITH EMERGENT QUESTIONS

3). **Do you feel that the website helps you learn?**
FOLLOW-UP WITH EMERGENT QUESTIONS

4). **Does the website help you stay on track with your goals?**
FOLLOW-UP WITH EMERGENT QUESTIONS

5). **Overall, what have been your experiences so far using the website?**
FOLLOW-UP WITH EMERGENT QUESTIONS

6). **Do you find the website interesting?**
FOLLOW-UP WITH EMERGENT QUESTIONS

7). **Do you like using the website to work with your team?**
FOLLOW-UP WITH EMERGENT QUESTIONS

8). **How do you think your experience in Technovation would be different if we used something else?**
FOLLOW-UP WITH EMERGENT QUESTIONS

9). **Do you feel that the website has helped you achieve your goals so far for Technovation?**
FOLLOW-UP WITH EMERGENT QUESTIONS

10). **Do you think the website helps support a sense of community– why or why not?**
EMERGENT QUESTION: **If so, in what ways?**

SUMMARIZE: **I'm really interested in creating ways to make Technovation more engaging for you. It looks like there are (quite a few/some) positive aspects of _____, whereas some/many areas of the website like _____ could use more work or be more clear.**

FINAL REFLECTION (FINAL PHASE)
Before we end today's session, I want to ask a few more questions about your overall experiences with technology.

1). **Before participating in Technovation, had you considered a career in technology?**
CONSIDER EMERGENT QUESTIONS

2). **Would you like to learn more about technology (in college, after high school, etc.)?**

CONSIDER EMERGENT QUESTIONS

3). **How has Technovation changed the way you view technology or business?**
CONSIDER EMERGENT QUESTIONS

4). **Would your recommend Technovation to your friends – why or why not?**

CONSIDER EMERGENT QUESTIONS

5). **What would make Technovation more fun?**

CONSIDER EMERGENT QUESTIONS

5). **What would make the website more fun?**

CONSIDER EMERGENT QUESTIONS

SUMMARIZE: **Overall, it seems like there are (quite a few/some) positive aspects of _____, whereas some/many areas of the course like _____ could use more work or be more clear.**

SESSION WRAP-UP

To summarize what we discussed, you said… *SUMMARIZE THE POSITIVE AND NEGATIVE ASPECTS OF STUDENT EXPERIENCES IN CSCI/ATLS-1220, ACKNOWLEDGING DIFFERENT POINTS OF VIEW.*

Finally, as I told you in the beginning, the purpose of this focus group is to get information about your experiences as a student in Technovation. Is there anything important that you think I left out?

This focus group is one of a series we are holding for the other section of the course, so any suggestions you could make for improving it would be help full.

Thank you, everyone, for participating in today's Focus Group! Now, let's move onto the next unit lesson/part of today's session.

FOCUS-GROUP END TIME: _____.

APPENDIX K
CASE STUDY THREE: PRE-TEST & POST-TEST SURVEY

Please answer the following questions as best you can. You can circle the answer, put a check-mark, or fill in the bubble.

1. I like to learn about technology.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
2. I know what a 'robot' does.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
3. If I don't succeed at first, I keep trying until I do.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
4. I know what a 'prototype' does.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
5. I think science and technology are important.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
6. I know what an 'engineer' does.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
7. I like to ask questions and solve problems.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
8. I know what a 'hypothesis' is.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
9. I can work in a team to solve problems and learn new things.
   Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
10. I know what a '3D model' is.
    Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
11. Learning new things is important to me.
    Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
12. I know what 'design thinking' is.
    Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
13. I like to learn new things, even if it is difficult at first.
    Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
14. If I don't know how to do something, I can figure out how to do it.
    Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely
15. I plan to take advanced math, science, or technology classes in the future.
    Not at all ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 Definitely

APPENDIX L
CASE STUDY THREE: INTERMITTENT SURVEYS

Please answer these questions about the **Cubelets and Robot Investigator activities** that you did today. Use the smiley-faces to help you answer. You can fill in the bubble, put a check mark, or circle the bubble to mark your answer.

	 Totally disagree	 Somewhat disagree	 Neutral (neither agree nor disagree)	 Somewhat agree	 Totally agree
I think the activities today were really interesting.	○	○	○	○	○
I enjoyed doing the activities today.	○	○	○	○	○
I had to concentrate really hard when doing the activities.	○	○	○	○	○
When doing the activities today, I didn't think about anything else.	○	○	○	○	○
The activities today were really challenging (difficult but fun).	○	○	○	○	○
I could figure out the activities' hardest problems if I tried.	○	○	○	○	○
Doing these activities is important.	○	○	○	○	○
Building Cubelets robots felt more like homework than play.	○	○	○	○	○
Doing the Robot Investigator activities felt more like homework than play.	○	○	○	○	○
I was learning a lot when doing the activities.	○	○	○	○	○
The Cubelets activities were fun to do.	○	○	○	○	○
The Robot Investigator activities were fun to do.	○	○	○	○	○
I had enough time to do the Cubelets activities.	○	○	○	○	○
I had enough time to do the Robot Investigator activities.	○	○	○	○	○

APPENDIX M
CASE STUDY FOUR: PRE-TEST SURVEY

LEARN TO CODE - PRE-TEST

Q1) What is your name? (Your name will not be used for any reason except for tracking your learning progress. Your personal information is kept safe and confidential).*

_____

Q2) Which class are you taking this survey for? (e.g., CSCI-1300, Meaning of IT, etc.)*

_____

Q3) Have you ever taken a computer science or programming class before?*
( ) Yes
( ) No
( ) Not sure
( ) Taking my first class now (Fall 2015)
( ) Other - Write In: _____

Q4) Please list the computer science or programming classes you have taken in the past.

_____
_____
_____
_____

Q5) Have you ever used a website or online training course to learn how to code or computer program?
( ) Yes
( ) No
( ) Not sure
( ) Other - Write In: _____

Q6) Which online websites or training courses have you used?

_____
_____
_____
_____

COMPUTER PROGRAMMING & CODING

Q7) This line of code uses which of the following?

```
x *= 10
  *
```

( ) Boolean
( ) relational operator

( ) mathematical operator
( ) logical operator
( ) string

Q8) This line of code uses which of the following?

```
name = "Jane Smith"
 *
```

( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q9) This line of code uses which of the following?

```
flag = TRUE
 *
```

( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q10) This line of code uses which of the following?

```
answer = 1 AND ( 1 OR 0 )
 *
```

( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q11) What is the output after this code is done?

```
PRINT "Hello" + "World"
 *
```

( ) Hello
( ) World
( ) HelloWorld
( ) Hello World
( ) WorldHello

Q12) What is the value of x after this code is done?

```
var  x = 10
if ( x == 10 )
     x = 20
 *
```

( ) 0
( ) 10
( ) 20

( ) 30
( ) 40

Q13) What is the final value of 'element'?

```
number = 81

IF number >= 90 THEN
    element = 'fire'
ELSE IF number >= 80 THEN
    element = 'water'
ELSE IF number >= 70 THEN
    element = 'metal'
ELSE IF number >= 60 THEN
    element = 'earth'
ELSE
    element = 'wood'
ENDIF
```

( ) fire
( ) water
( ) metal
( ) earth
( ) wood

Q14) What is the final value of 'jump'?

```
jump = 1
count = 1

WHILE  count  < 4 DO
    jump = jump + 1
    count = count + 2
ENDWHILE
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q15) What is the output the last time the loop is run?

```
FOR ( x = 1, x <= 5,  x++ )
    xSquared = x * x
    PRINT x,  xSquared
ENDFOR
```

( ) 1 1
( ) 2 4
( ) 3 9
( ) 4 16

Q16) The following code is commonly referred to as a:

```
DEFINE findMax(num1, num2)
    IF num1 >= num2 THEN
        maxNum = num1
    ELSE
        maxNum = num2
    ENDIF
  RETURN maxNum
ENDDEF
```

( ) variable
( ) branching
( ) function
( ) enumeration
( ) polymorphism

Q17) Which of the following statements correctly creates an object for the class?

```
CLASS Car
    DEFINE initialize Car()
        wheels = 4
        currentSpeed = 100
    ENDDEF

    DEFINE speedUp(newSpeed)
        currentSpeed = newSpeed
    ENDDEF
ENDCLASS
 *
```

( ) car1 = new Car( )
( ) new Car( )
( ) car1( )
( ) new Car = Car( )
( ) Car = car1( )

Q18) Which of the following statements correctly calls a method?

```
CLASS Car
    DEFINE initialize Car()
        wheels = 4
        currentSpeed = 100
    ENDDEF

    DEFINE speedUp(newSpeed)
        currentSpeed = newSpeed
    ENDDEF
ENDCLASS
 *
```

( ) car1.Currentspeed(100)
( ) car1.wheels
( ) car1.speedUp(200)
( ) car1.initializeCar()
( ) car1.newSpeed

Q19) What is the output for the following code?

```
name = "Jane Smith"
name.upper()
 *
```

( ) JANE SMITH
( ) Jane Smith
( ) jane smith
( ) JANE Smith
( ) Jane SMITH

Q20) What is the output for the following code?

```
name = "Jane Smith"
name.count('n')
 *
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q21) What is the output for the following code?

```
name = "Jane Smith"
name.index('Sm')
 *
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q22) What is the output for the following code?

```
sports = ['soccer', 'football', 'hockey']
length(sports)
 *
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q23) What is the output for the following code?

```
sports = ['soccer', 'football', 'hockey']
sports[1]
 *
```

( ) hockey
( ) football
( ) soccer
( ) sports

Q24) What is the output for the following code?

```
sports = ['soccer', 'football', 'hockey']
add(sports[3], 'baseball')
delete(sports[2])
 *
```

( ) soccer, football, hockey
( ) soccer, football, baseball
( ) football, hockey, baseball
( ) baseball, football, hockey
( ) soccer, baseball, hockey

THANK YOU!

APPENDIX N
CASE STUDY FOUR: POST-TEST SURVEY

LEARN TO CODE - POST-TEST

INTRODUCTION

Q1) What is your name? (Your name will not be used for any reason except for tracking your learning progress. Your personal information is kept safe and confidential).*****

_____

Q2) Which of the following learning resources did you choose to use for this study?*

[ ] Gidget
[ ] Khan Academy
[ ] Grok Learning
[ ] Scratch
[ ] Other - Write In: _____

Q3) Approximately how many hours did you spend using Gidget?*

_____

Q4) What did you like about using Gidget?*

_____
_____
_____
_____

Q5) What did you dislike about using Gidget?*

_____
_____
_____
_____

Q6) I was motivated to use Gidget because it is a game.*

( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q7) I think Gidget was boring to use.*

( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q8) Overall, Gidget helped motivate me to learn.*

( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q9) Approximately how many hours did you spend using Khan Academy?*

_____

Q10) What did you like about using Khan Academy?*

_____
_____
_____
_____

Q11) What did you dislike about using Khan Academy?*

_____
_____
_____
_____

Q12) The "energy points" in Khan Academy helped me stay motivated.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q13) The "badges" in Khan Academy helped me stay motivated.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q14) The "progress bar" in Khan Academy helped me stay motivated.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q15) The "knowledge map" in Khan Academy helped me stay motivated.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q16) I think Khan Academy was boring to use.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q17) Overall, Khan Academy helped motivate me to learn.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

LEARNING & MOTIVATION

Q18) I am learning how to code because I think it is interesting.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q19) I am learning how to code for my own good.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q20) I'm learning how to code because I am supposed to do it.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q21) There may be good reasons to learn how to code, but personally I don't see any.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q22) I think programming is pleasant.*

( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q23) I think programming is good for me.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q24) I am learning programming because it is something that I have to do.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q25) I am not sure if learning programming is worth it.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q26) I am learning how to code because it is fun.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q27) I am learning how to code because it is my personal decision.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q28) I am learning how to code because I don't have any choice.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q29) I don't know why I am learning how to code; I don't see what it brings me.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q30) I felt good when doing the programming activities.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q31) I did the programming activities because I believe it is important for me.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q32) I am learning programming because I feel that I have to do it.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

Q33) I am not sure if programming is a good thing to pursue.*
( ) Strongly Disagree ( ) Disagree ( ) Disagree somewhat ( ) Undecided ( ) Agree somewhat
( ) Agree ( ) Strongly Agree

COMPUTER PROGRAMMING & CODING

Q34) This line of code uses which of the following?

x *= 10*
( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q35) This line of code uses which of the following?

```
name = "Jane Smith"*
```

( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q36) This line of code uses which of the following?

```
flag = TRUE*
```

( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q37) This line of code uses which of the following?

```
answer = 1 AND ( 1 OR 0 )*
```

( ) Boolean
( ) relational operator
( ) mathematical operator
( ) logical operator
( ) string

Q38) What is the output after this code is done?

```
PRINT "Hello" + "World"*
```

( ) Hello
( ) World
( ) HelloWorld
( ) Hello World
( ) WorldHello

Q39) What is the value of x after this code is done?

```
var  x = 10
if ( x == 10 )
     x = 20
 *
```

( ) 0
( ) 10
( ) 20
( ) 30
( ) 40

Q40) What is the final value of 'element'?

```
number = 81

IF number >= 90 THEN
      element = 'fire'
ELSE IF number >= 80 THEN
```

```
        element = 'water'
    ELSE IF number >= 70 THEN
        element = 'metal'
    ELSE IF number >= 60 THEN
        element = 'earth'
    ELSE
        element = 'wood'
    ENDIF*
```

( ) fire
( ) water
( ) metal
( ) earth
( ) wood

Q41) What is the final value of 'jump'?

```
    jump = 1
    count = 1

    WHILE  count
        jump = jump + 1
        count = count + 2
    ENDWHILE*
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q42) What is the output the last time the loop is run?

```
    FOR ( x = 1, x <= 5,  x++ )
        xSquared = x * x
        PRINT x,  xSquared
    ENDFOR*
```

( ) 1 1
( ) 2 4
( ) 3 9
( ) 4 16

Q43) The following code is commonly referred to as a:

```
    DEFINE findMax(num1, num2)
        IF num1 >= num2 THEN
            maxNum = num1
        ELSE
            maxNum = num2
        ENDIF
      RETURN maxNum
    ENDDEF*
```

( ) variable
( ) branching
( ) function

( ) enumeration
( ) polymorphism

Q44) Which of the following statements correctly creates an object for the class?

```
CLASS Car
    DEFINE initialize Car()
        wheels = 4
        currentSpeed = 100
    ENDDEF

    DEFINE speedUp(newSpeed)
        currentSpeed = newSpeed
    ENDDEF
ENDCLASS*
```

( ) car1 = new Car( )
( ) new Car( )
( ) car1( )
( ) new Car = Car( )
( ) Car = car1( )

Q45) Which of the following statements correctly calls a method?

```
CLASS Car
    DEFINE initialize Car()
        wheels = 4
        currentSpeed = 100
    ENDDEF

    DEFINE speedUp(newSpeed)
        currentSpeed = newSpeed
    ENDDEF
ENDCLASS*
```

( ) car1.Currentspeed(100)
( ) car1.wheels
( ) car1.speedUp(200)
( ) car1.initializeCar()
( ) car1.newSpeed

Q46) What is the output for the following code?

```
name = "Jane Smith"
name.upper()*
```

( ) JANE SMITH
( ) Jane Smith
( ) jane smith
( ) JANE smith
( ) Jane SMITH

Q47) What is the output for the following code?

```
name = "Jane Smith"
name.count('n')*
```

( ) 1

( ) 2
( ) 3
( ) 4
( ) 5

Q48) What is the output for the following code?

```
name = "Jane Smith"
name.index('Sm')
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q49) What is the output for the following code?

```
sports = ['soccer', 'football', 'hockey']
length(sports)*
```

( ) 1
( ) 2
( ) 3
( ) 4
( ) 5

Q50) What is the output for the following code?

```
sports = ['soccer', 'football', 'hockey']
sports[1]*
```

( ) hockey
( ) football
( ) soccer
( ) sports

Q51) What is the output for the following code?

```
sports = ['soccer', 'football', 'hockey']
add(sports[3], 'baseball')
delete(sports[2])*
```

( ) soccer, football, hockey
( ) soccer, football, baseball
( ) football, hockey, baseball
( ) baseball, football, hockey
( ) soccer, baseball, hockey

THANK YOU!