Python for Math and Stat Fall 2024
Exam 1

1. (20 pts) For the following 4 problems, write down what each code block would display if executed in a Jupyter cell. Write ERROR if the code produces an error.

   (a)
   ```
   (20 // 3, '20' * 3)
   ```

   (b)
   ```
   nums = list(range(10, 90, 10))
   nums[2::2]
   ```

   (c)
   ```
   ints = [7, 2, 0, 4]
   [(k in ints) for k in range(4)]
   ```

   (d)
   ```
   word = 'CUB'
   for x in word:
       print(x + 'x', end='')
   ```

   Solution:

   (a) `(6, '202020')`

   (b) `[30, 50, 70]`

   (c) `[True, False, True, False]`

   (d) `CxUxBx`

2. (10 pts) Write a **list comprehension** that takes a non-empty string `word` and produces a list containing the first character of `word`, first two characters, and so on, ending with the entire word.

   Example:
   For `word = 'buff'`, the result is `['b', 'bu', 'buf', 'buff']`.

   Solution:

   ```
   [word[:n+1] for n in range(len(word))]
   ```

   OR

   ```
   [word[:n] for n in range(1, 1+len(word))]
   ```

3. (10 pts) Write a function **`create_IDs(names, locknums)`** that creates student identifiers by concatenating student names to their locker numbers. The function takes 2 non-empty lists of the same length. List `names` will contain the student names in `str` format, and list `locknums` will contain their corresponding locker numbers in `int` format. The function returns a list of the identifiers in string format.

   Example:
   `students = ['Alice', 'Bob', Carol']`
   `lockers = [25, 431, 8]`
   `create_ID(students, lockers)`
   returns
   `['Alice25', 'Bob431', 'Carol8'].`

   Solution:

```
def create_ID(names, locknums):
    return [name + str(locker) for name, locker in zip(names, locknums)]
```

   OR

```
def create_ID(names, locknums):
    ids = []

    for i in range(len(names)):
        ids.append(names[i] + str(locknums[i]))

    return ids
```

4. (10 pts) Write a function **`nums_end_digit(digit, nums)`** that takes as input a single `digit` and a list of positive `int`s called `nums`. It returns a new list containing the elements in `nums` that end in `digit`. Assume `digit` is an `int`. If `digit` is not a single digit 0–9, the return value is an empty list. The function should use a loop or list comprehension. Do not condense all the code into one line or use string operations.

   Examples:
   `nums_end_digit(5, [45, 90, 5, 52])` returns `[45, 5]`.
   `nums_end_digit(0, [45, 90])` returns `[90]`.
   `nums_end_digit(2, [45, 90])` returns `[]`.
   `nums_end_digit(90, [45, 90])` returns `[]`.

   Solution:

```
def nums_end_digit(digit, nums):
    if not 0 <= digit <= 9:
        return []
    else:
        return [n for n in nums if n % 10 == digit]
```