Name: ______________________________ Andrew Id: ____________________

15-112 Spring 2024 Quiz 3

Up to 25 minutes. No calculators, no notes, no books, no computers. Show your work!
Do not use strings, lists, tuples, dictionaries, sets, try/except, or recursion on this quiz.

1. **Code Tracing**: Indicate what the following two programs print. Place your answers (and nothing else) in the boxes below the code.

   (a) (5 points) CT1

```
def CT1():
    for i in range(9, 1, -3):
        for j in range(i//2, 1, -1):
            if i%j == 0:
                print(f"{j} is a factor of {i}")
    print(i*j)

print(CT1())
```

2. (7 points) **Free Response**: nth9ish

Write a function `nth9ish(n)` which takes as an input a number (n) and returns the nth number whose digits sum is a multiple of 9. *Note: 9 is the 1st 9ish number.*

You would likely benefit from writing a helper function, `is9ish(n)` that returns `True` if `n` is 9ish and `False` otherwise.

Consider the following test cases:

```
# Normal cases
assert(nth9ish(1) == 9)
assert(nth9ish(4) == 36)
assert(nth9ish(10) == 90)
assert(nth9ish(203) == 1827)

# Abnormal cases
assert(nth9ish(0) == None)
assert(nth9ish(-5) == None)
```

3. (8 points) **Free Response**: Digit Contribution

Given a number n, the position i of a certain digit d in the number indicates its importance. In other words, d contributes $d \times 10^i$ to the value of the number (n). Write a function `digitContribution(n, d)` that returns the sum of the contributions of digit d to a given integer n based on its positions in n. For example, `digitContribution(4525, 5)` returns 505 since digit 5 appears in positions 2 and 0 and $(5 \times 10^2 + 5 \times 10^0) = 505$.

Consider the following additional test cases:

```
# Normal cases
assert(digitContribution(1234567859005, 5), 500050005) # 5 appears in positions 0, 4, and 8
assert(digitContribution(2, 2), 2) # 2 appears in position 0
assert(digitContribution(-302, 3), 300) #given the absolute value, 3 appears in position 2


# Abnormal cases
assert(digitContribution(58502, 9), None)  # 9 doesn't appear
assert(digitContribution(206, 0), None) #0 doesn't contribute to the value
assert(digitContribution(206, -2), None) # d is negative
assert(digitContribution(206, 25) ,None) # d is more than 1 digit
```