fullName:_____

andrewID:_____

recitationLetter:_____

15-112 F22

Quiz5 version A

You **MUST** stop writing and hand in this **entire** quiz when instructed in lecture.

- You may not unstaple any pages.
- Failure to hand in an intact quiz will be considered cheating. Discussing the quiz with anyone in any way, even briefly, is cheating. (You may discuss it only once the quiz has been posted to the course website.)
- You may not use your own scrap paper. If you must use additional scrap paper, raise your hand and we will provide some. You must hand any scrap paper in with your paper quiz, and we will not grade it.
- You may not ask questions during the quiz, except for English-language clarifications. If you are unsure how to interpret a problem, take your best guess.
- You may not use any concepts (including builtin functions) we have not covered in the notes this semester.
- You may not use dictionaries, sets, or recursion.
- We may test your code using additional test cases. Do not hardcode.
- Assume almostEqual(x, y) and roundHalfUp(n) are both supplied for you. You must write all other helper functions you wish to use.
- **Write your answers entirely inside the boxes!**

- **Note: There are three required problems in the following order: FR1, FR2, and CT1. Don't forget CT1!**

Free Response 1: destructiveTile(L) [45 points]

Write the function destructiveTile(L) which takes a 2d rectangular list L (with at least one row and one column), and **destructively** modifies the list so that it contains twice as many rows and columns, and with the inner values arranged as if four duplicates of L were placed inside. Study the test cases to understand the pattern! As usual for destructive functions, this function returns None.
Also, make sure that **none of the rows are aliased** to each other!

Note: If you do not know how to write this destructively, you may write it nondestructively instead for half-credit. A nondestructive function must return a new list with the correct contents and must not mutate L.

```
def testDestructiveTile():
    L =         [['a', 'b'],
                 ['c', 'd']]
    assert(destructiveTile(L) == None)
    assert(L == [['a', 'b', 'a', 'b'],
                 ['c', 'd', 'c', 'd'],
                 ['a', 'b', 'a', 'b'],
                 ['c', 'd', 'c', 'd']])

    L =         [[112]]
    assert(destructiveTile(L) == None)
    assert(L == [[112, 112],
                 [112, 112]])

    L =         [[10],
                 [20],
                 [30]]
    assert(destructiveTile(L) == None)
    assert(L == [[10, 10],
                 [20, 20],
                 [30, 30],
                 [10, 10],
                 [20, 20],
                 [30, 30]])

    #Continued on the next page...
```

```
    #Continued on the previous page...
    L =         [[10, 20, 30]]
    assert(destructiveTile(L) == None)
    assert(L == [[10, 20, 30, 10, 20, 30],
                 [10, 20, 30, 10, 20, 30]])
    #Test that the rows are not aliased:
    L[0][0] = 40
    assert(L == [[40, 20, 30, 10, 20, 30],
                 [10, 20, 30, 10, 20, 30]])

    print('Passed!')
testDestructiveTile()
```

Begin your FR1 answer here

Continue your FR1 answer here

Free Response 2: makeMultiplicationTable(n) [40 points]

Write the function makeMultiplicationTable(n) that takes a positive integer n and returns a multiplication table in the form of a 2D list with n rows and n columns. Each cell should be the product of the integer at the beginning of its row and the top of its column. Look at the test cases to identify this pattern.

```
def testMakeMultiplicationTable():
    assert(makeMultiplicationTable(1) == [[1]])

    assert(makeMultiplicationTable(2) == [[1, 2],
                                          [2, 4]])

    assert(makeMultiplicationTable(3) == [[1, 2, 3],
                                          [2, 4, 6],
                                          [3, 6, 9]])

    assert(makeMultiplicationTable(5) == [[1, 2, 3, 4, 5],
                                          [2, 4, 6, 8, 10],
                                          [3, 6, 9, 12, 15],
                                          [4, 8, 12, 16, 20],
                                          [5, 10, 15, 20, 25]])
    print('Passed!')
```

You may begin your FR2 answer here or on the following page

You may continue your FR2 answer here

CT1: Code Tracing [15pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```
import copy
def ct1(x):
    y = x
    z = copy.copy(x)
    y += [112]
    x[0] = y[0] + y[1]
    z[1].append(5)
    x = x[0]
    print(f'x = {x}')
    print(f'y = {y}')
    print(f'z = {z}')

L = [[10],[25]]
ct1(L)
print(f'L = {L}')
```

bonusCT: Code Tracing [2pts bonus]

This question is optional. Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```
def bonusCt(n):
    return ([[[x]*x for x in range(n)]
                    for y in [[z]*z
                    for z in range(n)][n//2]][0][1])
print(bonusCt(5))
```