fullName:_____andrewID:_____ recitationLetter:_____

15-112 S24

WS11+Quiz11 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) which we have not covered in the notes in weeks 1-11.
- We may test your code using additional test cases.
- Assume almostEqual(x, y) and rounded(n) are both supplied for you. You must write all other helper functions you wish to use, unless we specify otherwise.

This page is left blank on purpose. We will not grade anything on this page.

Writing Session 11 (10% of HW11)

Indicate which of the following statements about HW11 solveKingsTour(initialBoard) are True by writing the word True or False in each blank. Please write the whole word, either "True or False." Note: solvesKingsTour(initialBoard) is the required problem, not to be confused with the bonus problem *is*KingsTour.

1._____ initialBoard is represented as a 2D list of integers, where a 0 indicates an unvisited square and a positive integer indicates the order in which a square was visited.

2._____ The following 2D list is a potential solution to a 3x3 board:

```
[[1, 2, 3,],
 [4, 5, 6,],
 [9, 8, 7]]
```

3._____ An initialBoard may possess multiple valid solutions, but calling solveKingsTour(initialBoard) will only return one of them.

Quiz11

Free Response 1: Rect class [100pts]

Write the Rect class so that the following test function passes. Do not hardcode to the test cases, and use OOP properly.

```
def testClasses():
    print('Testing classes...', end='')
    rect1 = Rect(4, 3)
    assert(rect1.width == 4)
    assert(rect1.height == 3)
    assert(str(rect1) == '4x3')
    assert(str([rect1]) == '[4x3]')

    assert(rect1.area() == 12) # width times height

    assert(rect1 == Rect(4, 3))
    assert(rect1 != Rect(3, 4)) # 4x3 rect is different from 3x4 rect!
    assert(rect1 != 'do not crash here')

    s = set()
    assert(rect1 not in s)
    s.add(rect1)
    assert(rect1 in s)
    assert(Rect(4, 3) in s)
    assert(Rect(3, 4) not in s)

    rect1.scale(10) # mutate to scale width and height both by 10
    assert(str(rect1) == '40x30')
    assert(rect1.area() == 1200)

    rect2 = rect1.makeHalfRect()  # returns a new rect with half the width
                                  # and half the height (use integer division)
    assert(str(rect2) == '20x15')
    assert(str(rect1) == '40x30') # note: do not mutate rect1 here

    assert(rect2 != rect1)
    rect2.scale(2)
    assert(rect2 == rect1)

    print('Passed')
```

Begin your FR1 answer on the following page. Reminder: You may not unstaple the quiz.

Begin your FR1 answer here:

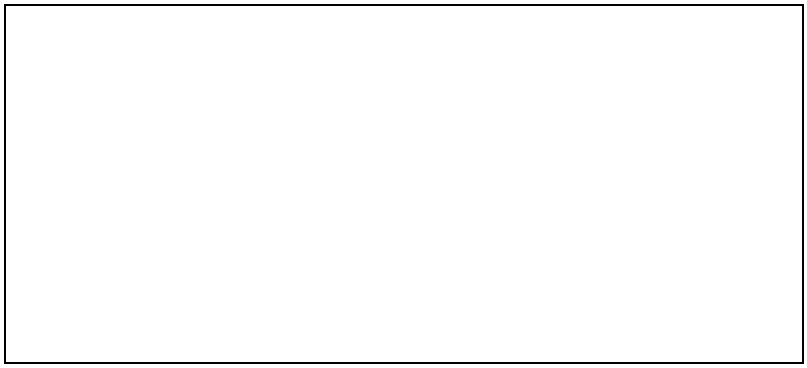
You may continue your FR1 answer here:

You may continue your FR1 answer here:

The problems below are not required. Indicate what the following code prints. Place your answers (and nothing else) in the boxes below.

bonusCt1 [optional, 1pt]

```
def bonusCt1(n):
    def f(n): return (n%10)+f(n//10) if n else 0
    def g(n,d): return n if f(n)%10 == 0 else g(n+d,d)
    return g(n,-1), g(n,+1)
print(bonusCt1(123))
```



bonusCt2 [optional, 1pt]

```
def bonusCt2(L,i=0,j=0):
    def f(n, k=2):
        return [n] if k==n else f(n, k+1) if n%k else [ ]
    n = len(L)
    if j == n: return [ ]
    elif i == n: return bonusCt2(L, 0, j+1)
    else: return f(10*L[i]+L[j]) + bonusCt2(L, i+1, j)
print(bonusCt2([1,2,3]))
```