andrewID:_____ recitationLetter:_____

Quiz 3 version C (25min)

Part 1: CTs and Fill-In-The-Blank

You may not run any code in Part 1. Once you move to Part 2, you may not return to Part 1.

CT1: Code Tracing [15pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below. Note that you may not run this code.

```
def ct1(s):
    d = 1
    t = ''
    for c in s.upper():
        if c.isspace():
            d -= 1
        elif c.isalpha():
            t += chr(ord(c) + d)
            d += 1
    return t
print(ct1('A\ne g!'))
```

CT2: Code Tracing [15pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below. Note that you may not run this code.

```
def ct2(s):
    t = s
    s += s[:2]
    t += s[::-1]
    print(t.replace('cb','f'))
    return t.find('fa')
print(ct2('abc'))
```

Fill-In-The-Blank: Graphics [24pts]

Fill in the blanks so that this code draws the given image (shown on the projector). Do not add any new lines, or delete any given lines. Just fill in the blanks. Your image does not need to resize and you may assume the canvas is 400x200 (though we may project it larger for visibility).

Note that you may not run this code.

```
from cmu_112_graphics import *

def redrawAll(app, canvas):
    dx = 50 # Width of first rectangle
    dy = 50 # Height of first rectangle
    for i in range(8):
        x0 = ____________________ # (A)
        y0 = 0
        x1 = ____________________ # (B)
        y1 = ____________________ # (C)
        dy = ____________________ # (D)
        color = ____________________ # (E)
        canvas.create_rectangle(x0, y0, x1, y1, fill=color)
    canvas.create_text(200, 0, text='Amazing',
                       anchor=____________________, # (F)
                       fill='yellow', font='Arial 50 bold')

runApp(width=400, height=200)
```

Part 2: Free Response

Free Response: finalLocation(path) [46pts]

This problem takes a string path, such as the following:

- 'Go 5 up'
- 'Go 3 left then 5 right'
- 'Go 3 left then 2 up then 15 right'
- 'Go 3 RIGHT then 2 DOWN then 5 UP then 2 RIGHT'

In general, the string will always start with 'Go', then a series of distances and directions, separated by 'then'. The distances will always be non-negative integers. The directions will always be 'left', 'right', 'up', or 'down', though case-insensitively.

Assume that you start at the origin (0,0), and that x increases to the right, and y increases upwards (like math, not like graphics).

With that in mind, write the function finalLocation(path) that takes a path as just described and returns the (x,y) location that you will be at after following this path, if you started at (0,0). See the test cases for examples.

Hint: To receive any credit, you must not use sets, dictionaries, or other concepts we have not covered in the notes this semester, and you may not use lists except for looping through the output of certain string methods. Also, we may test your code against test cases not shown here. Do not hardcode.

```
# Note: almostEqual(x, y) and roundHalfUp(n) are both supplied for you.
# You must write all other helper functions you wish to use.
def almostEqual(d1, d2, epsilon=10**-7): #helper-fn
    return (abs(d2 - d1) < epsilon)

import decimal
def roundHalfUp(d): #helper-fn
    # Round to nearest with ties going away from zero.
    rounding = decimal.ROUND_HALF_UP
    return int(decimal.Decimal(d).to_integral_value(rounding=rounding))
#----------------

#Write your function here:
def finalLocation(path):
    return (x, y)  #Hint: This is how to return two values
```

```
def testFinalLocation():
    print('Testing finalLocation()...', end='')
    assert(finalLocation('Go 5 up') == (0, 5))
    assert(finalLocation('Go 3 left then 5 right') == (2, 0))
    assert(finalLocation('Go 3 left then 2 up then 15 right') == (12, 2))
    assert(finalLocation('Go 3 RIGHT then 2 DOWN then 5 UP then 2 RIGHT')
                                                                == (5, 3))
    print('Passed!')

testFinalLocation()
```

Bonus Part 3

bonusCT1: Code Tracing [2pts]

This question is optional. Indicate what the following code prints. Place your answers (and nothing else) in the box below. Note that you may not run this code.

```
def bonusCt(x):
    s, n, d = '1', 7, 5
    for x in range(x):
        x,s = 2*int(2*s),3*str(10*x)
        s = str(100*int(x)).replace('0','1')
        n += 7
    while str(n) in s:
        s = s.replace(str(n), str(d))[::-1]
        d += 1
        n *= 2
    return int(s[:4]) - int(s[4:])
print(bonusCt(2))
```