fullName:____

andrewID:____

recitationLetter:__________

15-112 F22

Midterm1 version C

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

- You may not unstaple any pages.
- Failure to hand in an intact exam will be considered cheating. Discussing the exam with anyone in any way, even briefly, is cheating. (You may discuss it only once the exam 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 exam, and we will not grade it.
- You may not ask questions during the exam, except for English-language clarifications. If you are unsure how to interpret a problem, take your best guess. We have provided a box at the beginning of the exam where you can write any assumptions that you would like us to consider. (This is entirely optional and will not likely impact your grade. We expect that most students will leave this box empty.)
- You may not use any concepts (including builtin functions or modules) 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!**

If you are unsure how to interpret a problem, take your best guess.
We have provided this box at the beginning of the exam where you can write any assumptions about specific problems that you would like us to consider. **This is entirely optional** and will not likely impact your grade. We expect that most students will leave this box empty.

Please clearly indicate the number of the problem followed by the assumption you are making. (For example, "FR4: I assume app.width and app.height will be greater than 0.")

Multiple Choice / Short Answer [6pts total]

MC1. For `x = [[]]`, what does `len(x)` evaluate to?
Select the best answer (fill in one circle).

◯ 0

◯ 1

◯ 2

◯ Causes a syntax error

◯ Causes a runtime error

SA2. How many unique list objects exist after running the following code?
(In other words, how many unique lists are there in a box-and-arrow diagram, or how many list objects would we see if we were to execute the following code in pythontutor.com?)

```
rows = 7
cols = 3
A = []
for row in range(rows):
    A += [[0]*cols]
```

Answer:

Consider the following app code for questions MC3.1 and MC3.2

```
from cmu_112_graphics import *
def resetShape(app):
    app.size = 100

def createShape(app, canvas):
    canvas.create_rectangle(0, 0, app.size, app.size)

def growShape(app):
    app.size *= 1.05

def shrinkShape(app):
    app.size *= 0.9

def appStarted(app):
    resetShape(app)

def keyPressed(app, event):
    shrinkShape(app)

def timerFired(app):
    growShape(app)

def redrawAll(app, canvas):
    createShape(app, canvas)
```

MC3.1 Mark each of the following functions as part of the model, view, or controller.
Select the best answer for each function (fill in one circle per row).

createShape	○ Model	○ View	○ Controller
growShape	○ Model	○ View	○ Controller
shrinkShape	○ Model	○ View	○ Controller
keyPressed	○ Model	○ View	○ Controller
timerFired	○ Model	○ View	○ Controller
redrawAll	○ Model	○ View	○ Controller

MC3.2 Which of the following could be called from redrawAll without causing an MVC violation?
Select ALL that apply (fill in at least one square).

☐ resetShape ☐ growShape ☐ shrinkShape ☐ None of the above

CT1: Code Tracing [8pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below. Hint: This prints four lines

```
def ct1(n):
    r = 0
    while n > 0:
        d = n % 10
        n //= 10
        if d % 2 == 0:
            r += d
        if r % 2 == 0:
            r += 1
        else:
            r -= 1
        print(r)

ct1(2560)
```

CT2: Code Tracing [6pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below. Hint: This prints one line

```
def ct2(s):
    r = ''
    if len(s) < 6:
        r = 'X'
    for i in range(len(s)):
        if s[i].isupper():
            r = s[i] + r
        elif s[i].isalpha():
            c = chr(ord(s[i])+1)
            r = r + c
        else:
            r = r + str(i)
    return r

print(ct2("r0B0\t"))
```

CT3: Code Tracing [8pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below. Hint: This prints four lines

```
import copy
def ct3(X):
    Y = X[2]
    Y[1].append(9)
    Z = copy.copy(X)
    Z[1].remove(4)
    Z = Z[1] + [0]
    Z.remove(0)
    print(f'Y = {Y}')
    print(f'Z = {Z}')
    while len(X) > 1:
        X[0] += 1
        X = X[:len(X)-1]
    print(f'X = {X}')

L = [1, [4, 0, 4], [112, [15]]]
ct3(L)
print(f'L = {L}')
```

Free Response 1: nthPalindromicPrime(n) [20 points]

Write the function nthPalindromicPrime(n) that takes a non-negative integer and returns the nth palindromic prime.
A prime number is any integer which has exactly two unique factors: 1 and itself. Some primes include 2, 7, 31, and 149

A palindrome is any number which is the same when its digits are reversed, such as: 5, 18481, 323, and 88

The 0th palindromic prime is 2. The beginning of the sequence of palindromic primes is as follows: 2, 3, 5, 7, 11, 101, 131, 151...

Note: You **must write isPrime(n)** as a helper function, which takes an integer n and returns True if n is prime and False otherwise. (You may write other helper functions if you wish, but it is not required.)

Hint: You may use any concepts covered in the notes prior to this midterm (and not just the ones covered in weeks 1 and 2).

```
def testNthPalindromicPrime():
    assert(nthPalindromicPrime(0) == 2)
    assert(nthPalindromicPrime(1) == 3)
    assert(nthPalindromicPrime(4) == 11)
    assert(nthPalindromicPrime(5) == 101)
    assert(nthPalindromicPrime(15) == 757)
    assert(nthPalindromicPrime(20) == 10301)
```

You may begin your FR1 answer here or on the next page

Begin or continue your FR1 answer here

Continue your FR1 answer here

Free Response 2: interleave(L) [16 points]

Write the nondestructive function interleave(L) that takes a rectangular 2D list L and returns a 1D list where the rows are interleaved as shown in the test cases. As a first example, observe the 3x4 list and test case below:

```
L = [['r0c0', 'r0c1', 'r0c2', 'r0c3'],
     ['r1c0', 'r1c1', 'r1c2', 'r1c3'],
     ['r2c0', 'r2c1', 'r2c2', 'r2c3']]
assert(interleave(L) == ['r0c0','r1c0','r2c0','r0c1','r1c1','r2c1',
                  'r0c2','r1c2','r2c2','r0c3','r1c3','r2c3']
```

The result is a 1D list of length 12.

Remember not to destructively modify L!

```
def testInterleave():
    L = [[1, 2,  3,  4 ],
         [5, 6,  7,  8 ],
         [9, 10, 11, 12]]
    assert(interleave(L) == [1, 5, 9, 2, 6, 10, 3, 7, 11, 4, 8, 12])
    L = [[1, 4],
         [2, 5],
         [3, 6]]
    assert(interleave(L) == [1, 2, 3, 4, 5, 6])
    L = [['a', 'b', 'c']]
    assert(interleave(L) == ['a', 'b', 'c'])
    L = [[True ],
         [False],
         [42   ]]
    assert(interleave(L) == [True, False, 42])
    L = [[]]
    assert(interleave(L) == [])
```

Begin your FR2 answer on the following page

Begin your FR2 answer here

You may continue your FR2 answer here

You may continue your FR2 answer here

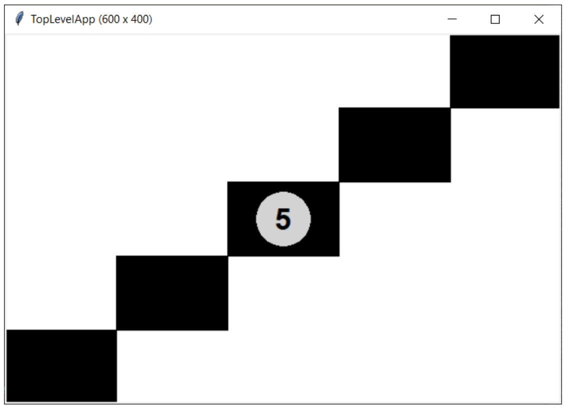
Free Response 3: Diagonal square app [20 points]

Write an animation with the following features:

1. Initially, 5 equally-sized rectangles are drawn from the bottom-left of the canvas to the top-right of the canvas as shown below. The rectangles should touch at their corners and should be filled black.
2. Pressing the "Up" key should increase the number of rectangles by one, up to a maximum of 20. Pressing the "Down" key should decrease the number of rectangles by one, down to a minimum of 1. (Do not allow the app to crash)
3. A circle with radius 30 should start in the center of the canvas. Text inside the circle always shows the current number of rectangles on the canvas.
4. Every 1000mS, the number of rectangles should decrease by 1, down to a minimum of 1.
5. Clicking the mouse anywhere on the canvas should center the circle and text at that new location.
6. The rectangles should resize appropriately with the canvas so that they always touch the top left and bottom corner. The circle and text do not resize with the canvas.

Additional Notes:

- Assume cmu_112_graphics is imported
- You must follow MVC rules
- You may abbreviate app, canvas, and event as a, c, and e
- You may make reasonable assumptions for any unspecified details like text size, circle color, etc.
- The example image shows what your app should look like before the keys or mouse are used.



Begin your FR3 answer on the following page

Begin your FR3 answer here

You may continue your FR3 answer here

You may continue your FR3 answer here

You may continue your FR3 answer here

You may continue your FR3 answer here

Free Response 4: decode(s) [16 points]

Write the function decode(s) which takes a string s (possibly multi-line) and returns the secret message whose words are formed by the uppercase letters in each line.

The result should be a single string with the words separated by one space. Each word is formed by the uppercase letters in a line, and each line of s should add one word to the result, unless the line contains no uppercase letters. Look at the test cases for examples!

Hint: Make sure not to add any extra spaces between words, or at the beginning or end of the message.

```
def testDecode():
    assert(decode("AbCdEf123") == "ACE")



    s = '''AxolotLs Might gO to Space wiTh
       DOgs aNd maybE cats!'''
    assert(decode(s) == "ALMOST DONE")

    s = '''
        LOOK
        FOR secret hints
        on the UPPER bookCASE
        next to the bANDanas!

        ummm... NEWLINES?
        '''
    assert(decode(s) == "LOOK FOR UPPERCASE AND NEWLINES")

    s = '''

        WATCH out for
        lines that have
        no UPPERcaSe
        letters

        '''
    assert(decode(s) == "WATCH UPPERS")
```

Begin your answer on the following page

Begin your FR4 answer here

You may continue your FR4 answer here

You may continue your FR4 answer here

bonusCT1: Code Tracing [1pt bonus]

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

```
def bonusCt1(n):
    return eval(str(list(range(n)))[1:-1]
                .replace(', ','-(') +
                (n-1)*')')
print(bonusCt1(50))
```

bonusCT2: Code Tracing [1pt bonus]

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

```
def z(n):
    m = -n/2
    while n > 0: m, n = m+n, n-1
    return round(2*m)
def bonusCt2(n):
    while n < 100: n = round(z(z(n))/n)
    s = str(n)
    for i in range(1, 12345):
        d = chr(int(s[:i]))
        if d.isdigit():
            return d * int(s[i:])
    return 42
print(bonusCt2(2))
```