fullName:_____andrewID:_____
recitationLetter:_____

15-112 F23

Midterm1 version B

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 quiz, except for English-language clarifications. (If you must ask an English-language question, please come out to the nearest aisle after you have our attention. We cannot come to you.) 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 1-5 / units 1-4.
- We may test your code using additional test cases. Do not hardcode.
- 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.

Sign your initials below to assert that you will not discuss or share information about the exam with anyone in any way, even briefly, until the exam has been posted on the course website.

x_____

Multiple Choice / Short Answers [2pts ea, 8pts total]

MC1. Which code block will print the entries of L in numerical order from least to greatest?

```
L = [ [1, 2, 3],
      [4, 5, 6],
      [7, 8, 9] ]
print3x3List(L)
```

Select the best answer (fill in one circle).

◯ **A**

```
def print3x3List(L):
    for i in range(3):
        for j in range(3):
            print(L[i,j], end=' ')
        print()
```

◯ **B**

```
def print3x3List(L):
    for i in range(3):
        for j in range(3):
            print(L[j,i], end=' ')
        print()
```

◯ **C**

```
def print3x3List(L):
    for i in range(3):
        for j in range(3):
            print(L[i][j], end=' ')
        print()
```

◯ **D**

```
def print3x3List(L):
    for i in range(3):
        for j in range(3):
            print(L[j][i], end=' ')
        print()
```

MC2. What does the following code print?

```
s = 'lowercase'
s.upper()
print(s)
```

Select the best answer (fill in one circle).

- ○ lowercase
- ○ Lowercase
- ○ LOWERCASE
- ○ Causes an error
- ○ None of the above

SA1. How many times is 'COOL' printed when running the following code?

```
count = 0
for x in range(3):
    for y in range(7):
        count += 1
    print('COOL')
```

Write your answer as an integer in the box below:

SA2. What does the following code print?

```
print(len(f'{3**2}+{4**2}'))
```

Write your answer as an integer in the box below:

CT1: Code Tracing [8pts]

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

```
def ct1(x):
    for i in range(5, 17, 2):
        x = 10*x + i
        if x > 1000:
            print(x)
            x //= 100
    while x > 0:
        i += x % 10
        x = x // 10
    return i

print(ct1(5))
```

CT2: Code Tracing [8pts]

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

```
def ct2(s):
    r = ''
    t = ''
    for i in range(len(s)):
        if s[i] in s[i+1:]:
            n = s[i:]
            n = n.count(s[i])
            print(str(i)+str(n))
            r += s[i]
        else:
            t = s[i] + t
    return r + 'a' + t
print(ct2('xyzxyx'))
```

CT3: Code Tracing [8pts]

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

```
def ct3(L):
    i = 0
    M = [True]
    N = L
    while i < len(L):
        if type(L[i]) == int:
            M.append(L.pop(i))
            N = N[1:]
        i += 1
    return [M, N]


L = ['xyz', 15, 112, False, 'abc', 70]
print(ct3(L))
print(L)
```

Free Response 1: lettersInFront(s) [14pts]

Write the function `lettersInFront(s)` which takes a string s (which may contain any valid ASCII characters) and returns a new string which contains all the same characters, but with all letter characters appearing at the beginning of the string. The letters at the front of the result string must be in the same relative order in which they appeared in s. The non-letter characters should also be in the same relative order. For example:
`lettersInFront('1Ch34?e\te&7')` returns `'Chee134?\t&7'`

Note that all the letters in the result are in the beginning of the string. Note also that the tab character `'\t'` is NOT a letter, and thus it remains with the other non-letters. See the test cases for more examples.

You are free to use anything we have covered thus far in the course.
To receive any credit, you must not use sets, dictionaries, recursion, or other concepts we have not covered in the notes this semester. Also, we may test your code against test cases not shown here, so your function must work in general for the kinds of inputs specified above.

```
def testLettersInFront():
    assert(lettersInFront('1Ch34?e\te&7') == 'Chee134?\t&7')
    assert(lettersInFront('\tea\time4me') == 'eaimeme\t\t4')
    assert(lettersInFront('many\naps 4 Chee?') == 'manyapsChee\n 4 ?')
    assert(lettersInFront('1,2,buckle1sh00') == 'bucklesh1,2,100')
    assert(lettersInFront('oddBall') == 'oddBall')
    assert(lettersInFront('123') == '123')
    assert(lettersInFront('') == '')
```

Begin your FR1 answer here or on the following page

You may begin or continue your FR1 answer here

Free Response 2: isOnePrime(n) + nthOnePrime(n) [18pts]

A "onePrime" is defined as any prime number that begins with 1 OR ends with 1, but NOT both. So 41 is a onePrime, as is 193, but 11 is not, and 191 is not. See the test cases for more examples.

Write the function `nthOnePrime(n)` which takes a non-negative integer n and returns the nth onePrime. As part of your solution to `nthOnePrime(n)`, you MUST also write the helper function `isOnePrime(num)` which takes any integer num and returns True if num is a onePrime and False otherwise. We have provided test cases for both of these functions below.

Note: The 0th onePrime is 13, followed by 17, 19, and 31.

To receive any credit, you must not use sets, dictionaries, recursion, or other concepts we have not covered in the notes this semester. Also, we may test your code against test cases not shown here, so your function must work in general for the kinds of inputs specified above.

```
def testIsOnePrime():
    assert(isOnePrime(41)==True)
    assert(isOnePrime(193)==True)
    assert(isOnePrime(11)==False)
    assert(isOnePrime(191)==False)
    assert(isOnePrime(13)==True)
    assert(isOnePrime(1)==False)
    assert(isOnePrime(-13)==False) # Negative
    assert(isOnePrime(10)==False) # Not prime
    assert(isOnePrime(23)==False) # No leading/trailing 1

def testNthOnePrime():
    assert(nthOnePrime(0) == 13)
    assert(nthOnePrime(1) == 17)
    assert(nthOnePrime(2) == 19)
    assert(nthOnePrime(3) == 31)
    assert(nthOnePrime(7) == 103)
    assert(nthOnePrime(23) == 211)
    assert(nthOnePrime(155) == 2011)
```

Begin your FR2 answer on the following page

Begin your FR2 answer here

You may continue your FR2 answer here

Free Response 3: intWithLargestPotential(L) [18pts]

In this problem, we will say that an integer's "potential" is the largest number that can be formed by rearranging its digits. So, for example:

1. The number 1234 has a potential of 4321, because 4321 is the largest number that can be formed by its digits
2. The number 929 has a potential of 992.
3. The number 5 has a potential of 5.

Write the **non-mutating** function `intWithLargestPotential(L)` which takes a non-empty 1D list of positive integers and returns the integer in the list which has the largest potential.

For example, `intWithLargestPotential([929, 1005, 1234, 5])` returns 1005, because 1005 has a potential of 5100. This is larger than the potential of 929, 1234, or 5. If there is a tie between two integers with the same potential, return the largest integer.

Remember, this function returns an integer that is present in the list, rather than the integer's potential. **Also, your function must not mutate L!**

To receive any credit, you must not use sets, dictionaries, recursion, or other concepts we have not covered in the notes this semester. Also, we may test your code against test cases not shown here, so your function must work in general for the kinds of inputs specified above.

```
def testIntWithLargestPotential():
    L = [929, 1005, 1234, 5]
    assert(intWithLargestPotential(L) == 1005)
    L = [427, 123, 631, 98]
    assert(intWithLargestPotential(L) == 427)
    L = [1, 2, 3, 4, 5]
    assert(intWithLargestPotential(L) == 5)
    L = [1200, 999]
    assert(intWithLargestPotential(L) == 1200)
    L = [31, 13, 3]
    assert(intWithLargestPotential(L) == 31) #In a tie, 31 is larger than 13
    L = [1]
    assert(intWithLargestPotential(L) == 1)
```

Begin your FR3 answer on the following page

Begin your FR3 answer here

You may continue your FR3 answer here

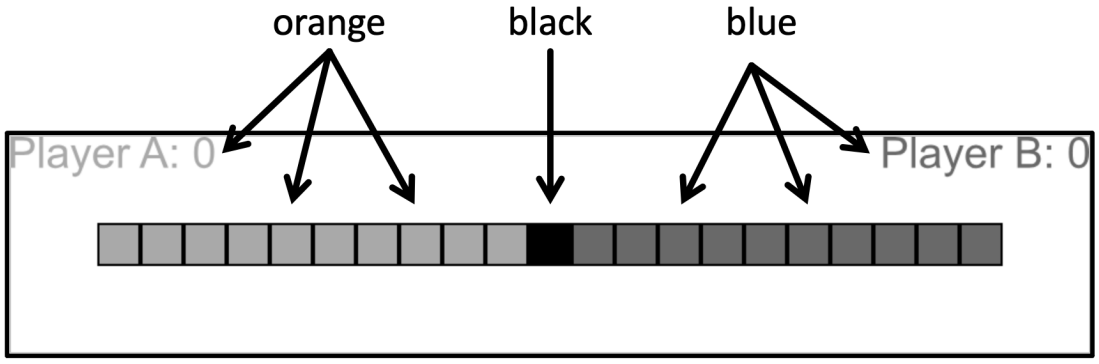
You may continue your FR3 answer here

Free Response 4: Push against time game [18pts]

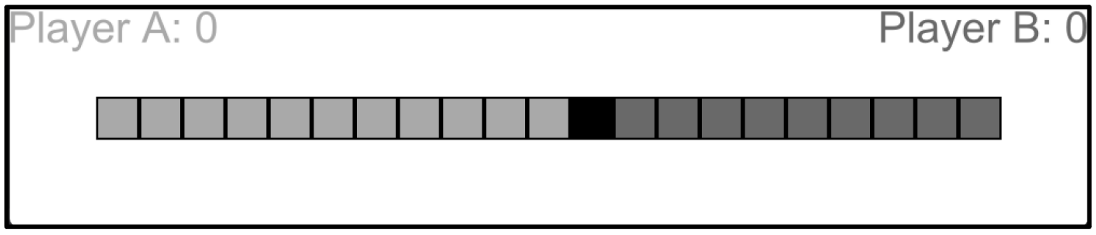
Write an animation with the following features. Read the whole problem before you begin!

The app always draws the following:

- A row of 21 squares:
  - The squares are drawn next to each other without any space between them
  - Each square is 30x30 with a 'black' border (default border width)
  - The square in the center of the row is centered at the center of the viewport
  - See below for square colors
- Scores for Player A and Player B:
  - The text will be 'Player A: ' (or 'Player B: ') followed by the current score of each player (an integer)
  - The score for Player A have an aligned in the "left-top" of the canvas
  - The score for Player B will be aligned in the "right-top" of the canvas
  - The font size for both will be 30
  - Player A text will be 'orange'
  - Player B text will be 'blue'
- Note: The canvas will not be resized, BUT do NOT assume a specific size of the canvas



When app starts, black square is in the center



After first 'a' key press, black square shifts to the right with
11 orange squares to the left and 9 blue to the right

CONTINUED ON NEXT PAGE

When the app starts:

- The center square will be 'black'
- All squares to the left of the black square will be 'orange'
- All squares to the right of the black square will be 'blue'
- The scores for both Player A and Player B will be 0
- The app will be paused

When the 'a' key is pressed:

- Unpause the app if it is paused
- If the black square is in the far right location of the 21 squares:
  - Increase the score of Player A by 1
  - Reset the location of the black square to be the center of the row
  - Pause the game
- Otherwise, shift the black square one square to the right

Every 0.25 seconds:

- Do nothing if the app is paused
- If the black square is in the far left location of the 21 squares:
  - Increase the score of Player B by 1
  - Reset the location of the black square to be the center of the row
  - Pause the game
- Otherwise, shift the black square one square to the left

Notes:

- Do NOT assume a specific size of the canvas (e.g., the canvas will NOT be 400x400)
- You may assume the canvas is big enough to fit all of the squares
- Reminder default alignment for drawLabel is `align='center'`
- You will be penalized if your code results in an MVC violation
- You may use the default values for anything not specified above, e.g., no need to specific border width or the font type (other than the size)
- Please write your code using the provided function headers for onAppStart, redrawAll, onKeyPress, onStep. As usual, you may write and use additional helper functions if you wish. (You may also add other event functions, but this is not recommended or necessary.)

Begin your FR4 answer on the following page

Begin your FR4 answer here:

```
from cmu_graphics import *

def onAppStart(app):









def redrawAll(app):
```

Continue your FR4 answer here

```
def onKeyPress(app, key):
```

Continue your FR4 answer here

```
def onStep(app):




runApp()
```