Name: _____ Recitation: _____ Andrew Id: _____

15-112 Summer 2018 Quiz 2

Up to 50 minutes. No calculators, scratch paper, notes, books, or computers. Do not use any topics that were not taught during week1 or week2. In particular, do not use sets, dictionaries, or recursion. Show your work!

1. (20 points) **Free Response**: Write the function sameSums(L) that takes in a 2D NxN list of integers and returns True if for every row in L there exists a column that has the same sum as the sum of elements in that row. It returns False if there is a case where there is no such row-column pair.

   Consider the following 2D Lists, L0 and L1:

```
L0 = [[1,2,3],
      [1,2,4],
      [1,2,0]]

L1 = [[1,1,1],
      [2,2,2],
      [3,3,3]]

assert(sameSums(L0) == True)
assert(sameSums(L1) == False)
```

   Note that sameSums(L0) == True because, in L0, the sum of row 0 is 1+2+3=6, which is the same as the sum of col 1 (2+2+2=6); the sum of row 1 is 1+2+4=7, which is the same as the sum of col 2 (3+4+0=7); sum of row 2 is 1+2+0=3, which is the same as the sum of col 0 (1+1+1=3). The rows and columns in L1 do not meet that condition.

   Hint: It might be helpful to write a helper function that, given a 2D list and a column index, returns the sum of the elements in that column.

   You may not use sets, dictionaries, or recursion on this question.

2. (50 points) **Free Response**: Using our animation framework and assuming that run() is already written, write init(data), mousePressed(event, data), keyPressed(event, data), timerFired(data) and redrawAll(data) for a 2-player animation with the following elements:

   - A 10x10 grid is the displayed on the screen. It should fill the entire canvas for any width and height specified. There is no margin.
   - Player One is a red circle that starts in the middle of the bottom row (row = 9, col = 5). On each timerFired call, the red circle should move up a row. When the circle moves off the top of the grid, it should wrap around and end up back at the bottom-most row.
   - Player One can move the circle left and right by pressing the arrow keys. If Player One presses the left arrow key, the red circle should move one column to the left in the grid. Similarly, when Player One presses the right arrow key, the red circle should move right one column.
   - If the red circle goes off the screen to the left or right, the game is over, and the text "Game over!" should appear anywhere on the screen.
   - Player Two uses green circles. If Player Two clicks inside of any cell, a green circle is drawn in that cell. Each green circle should fill the entire cell, but it should not be larger than the cell.
   - If the red circle, Player One, ever moves into a cell containing one of Player Two's green circles, that green circle is removed from the grid.
   - If Player Two ever places a green circle on top of Player One's red circle, the game is over, and the text "Game Over!" should appear anywhere on the screen.

   Note: Feel free to abbreviate canvas, data, and event as c, d, and e. Remember that data comes with data.width and data.height set to the width and the height of the canvas. It also comes with data.timerDelay set to 100 milliseconds, but you shouldn't need to change or use this value.

   You cannot assume make2dGrid from the lecture notes is written for you.

   You may not use sets, dictionaries, or recursion on this question.

Extra Space To Write Solution to Animation

3. (20 points) **Code Tracing**

Indicate what the following program prints. Place your answers (and nothing else) in the box under the code. Show your work anywhere outside the box.

```
import copy
def ct(a):
    b = a
    c = copy.copy(a)
    d = copy.deepcopy(a)
    b.append(42)
    a[0][0] = 99
    b = b[::-1]
    d[1][2] = 112
    c[0][1] = 17
    c[0] = d[1].pop()
    a[0] = a[1]
    print("a =", a)
    print("b = ", b)
    print("c =", c)
    print("d = ", d)

e = [[1, 2, 3],[4, 5, 6]]
ct(e)
print("e = ", e)
```

4. (10 points) **Reasoning Over Code**: Find an argument (the value of L) for roc that makes it return True. Place your answer (and nothing else) in the box. Show your work anywhere outside the box.

   Hint: L is a non-rectangular 2D list.

```
def roc(L):
    assert(len(L) == 4)
    swaps = 0
    for rowIndex in range(len(L)):
        if(rowIndex % 2 == 1):
            L[rowIndex], L[swaps] = L[swaps], L[rowIndex]
            swaps += 1
    res = []
    for rowIndex in range(len(L)):
        res.append(len(L[rowIndex]))
        if(sum(L[rowIndex]) != rowIndex):
            return False
    return res == [1,2,3,4]
```