Name: ______________________ Andrew Id: ______________

15-112 Spring 2024 Exam 1 Review

Up to 25 minutes. No calculators, no notes, no books, no computers. Show your work!
Do not use lists, tuples, dictionaries, sets, try/except, or recursion on this review.
0 points

1. (points) **Code Tracing**:

```
def ct1(s):
    t = ''
    for i in range(len(s)):
        t += s[i:] + s[i:][::-1]
    for c in s[::-1]:
        t = (t.replace(c+c,'') + str(t.count(c) + 1))
    print(t)


print(ct1('abcd'))
```

Solution:

```
9753
None
```

2. (points) **Code Tracing**:

```
def ct2(n):
    def f(n):
        r,m = 0,n
        while m>0:
            r,m = r+m,m-1
        return 2*r-n
    def g(n):
        return (f(n+1) - f(n-1))
    return f(g(n))

print(ct2(3))
```

Solution:

144

3. (points) **Free Response**: hasBalancedParentheses(s)

   Write the function `hasBalancedParentheses`, which takes a string and returns True if that code has balanced parentheses and False otherwise (ignoring all non-parentheses in the string). We say that parentheses are balanced if each right parenthesis closes (matches) an open (unmatched) left parenthesis, and no left parentheses are left unclosed (unmatched) at the end of the text. So, for example, `"( ( ( ) ( ) ) ( ) )"` is balanced, but `"(15 112 is Fun := * ))"` is not balanced, and `"( ) ) ("` is also not balanced. Hint: keep track of how many right parentheses remain unmatched as you iterate over the string.

Solution:

```
def hasBalancedParentheses(s):

    if not type(s)==str:
        return

    balance = 0

    for c in s:
        if c == '(':
            balance += 1
        elif c == ')':
            balance -= 1

            if balance < 0:
                return False

    return balance == 0
```

4. (points) **Free Response**: wordsEndingInNumber(s)

Write the function `wordsEndingInNumber(s)` which, given a string s, returns a string with a comma-separated list of words from the string that end with a number. You may assume that words in the string are separated by spaces. For example: `wordsEndingInNumber("purple monkey5 dish washer12 74 2evergreen")` returns `"monkey5,washer12,74"`.

Solution:

```
def wordsEndingInNumber(s):

    if not type(s)==str:
        return

    ending_with_number = ""

    for w in s.split():
        if w[-1].isdigit():
            ending_with_number+=(w+",")

    return ending_with_number[:len(ending_with_number)-1]
```

5. (points) **Free Response**: Animations: Stop the Circle!

Write a game animation with the following features.
1. When the game starts, a red square with side lengths of 50 pixels is drawn centered on the canvas, and a blue circle of radius 20 pixels is drawn at a random location on the canvas.
2. When the user clicks inside the circle immediately after clicking inside the red square, the circle radius decreases by half, and the circle moves to a new random location.
3. Four times per second, the circle's radius grows by 1 pixel.
4. If any part of the circle ever goes beyond the edge of the canvas, a game-over message appears at the center of the canvas, and the square and the circle are removed from the screen.
5. If the game is over, the user can restart the game by pressing r.

Notes:

- Do not hardcode for a 400x400 canvas. However, you may assume the canvas is square and at least 100x100 pixels.
- You will be penalized if your code results in an MVC violation.
- Make reasonable choices for anything not specified above.
- To solve this, you need to write onAppStart, onKeyPress, onMousePress, redrawAll, and onStep.
- You do not need to include any imports or the main function.

Solution:

```
from cmu_graphics import *
import random

def restart(app):
    app.r = 20
    app.cx = random.randint(app.r, app.width-app.r)
    app.cy = random.randint(app.r, app.height-app.r)
    app.gameOver = False
    app.pressedRect = False
    app.nSteps = 0


def onAppStart(app):
    app.stepsPerSecond = 4
    restart(app)

def redrawAll(app):
    if app.gameOver:
        drawLabel("game Over", app.width//2, app.height//2)
    else:
        drawCircle(app.cx, app.cy, app.r, fill="blue")
        drawRect(app.width//2, app.height//2, 50, 50, align='center', fill="red")

def dist(x1, y1, x2, y2):
    return ((x1 - x2)**2 + (y1 - y2)**2)**0.5

def onMousePress(app, x, y):
    if app.width//2 - 10 <= x <= app.width//2 + 10 \
    and app.height//2 - 10 <= y <= app.height//2 + 10:
        app.pressedRect = True
    else:
            if dist(app.cx, app.cy, x, y) < app.r and app.pressedRect == True:
                app.r //= 2
                app.cx = random.randint(app.r, app.width-app.r)
                app.cy = random.randint(app.r, app.height-app.r)
            app.pressedRect=False
def onKeyPress(app, key):
    if key == 'r':
        restart(app)


def gameOver(app):
    if app.cx + app.r >= app.width or app.cx - app.r < 0:
        return True

    if app.cy + app.r >= app.height or app.cy - app.r < 0:
        return True

    return False


def onStep(app): #app.stepsPerSecond= 4
    app.nSteps += 1 #tracks how many times onStep is called
    if app.nSteps % 15 == 0: #every 15 seconds
            app.r += 1
    if gameOver(app):
        app.gameOver = True


runApp(600, 300)
```