15-112
Spring 2019 Exam 1
February 28, 2019

Name:

Andrew ID:

- You may not use any books, notes, or electronic devices during this exam.
- You may not ask questions about the exam except for language clarifications.
- Show your work on the exam to receive credit.
- You may use the backs of pages as scratch paper. Nothing written on the back of any pages will be graded.
- All code samples run without crashing. Assume any imports are already included as required.
- You may assume that math, string, tkinter, and copy are imported; do not import any other modules.
- Do not use these post-midterm 1 topics/constructs: sets, maps/dictionaries, recursion, or classes/OOP.

Don't write anything in the table below.

Question	Points	Score
1	15	
2	10	
3	10	
4	20	
5	20	
6	25	
Total:	100	

1. **Short Answer**

   Answer each of the following *very briefly*.

   (a) (3 points) Briefly describe the top-down design approach to problem solving.

   (b) (3 points) Complete this function that returns `True` if the two circles intersect and `False` otherwise:

   ```
   def circlesIntersect(cx1, cy1, r1, cx2, cy2, r2):
   ```

   (c) (3 points) What does it mean for a data type to be immutable? Also, list two Python data types that are immutable.

(d) (3 points) Name and briefly explain what the three elements of MVC are.

(e) (3 points) Write the non-destructive list function `randomizeList(a)` which, given a list `a` creates and a returns a new list that contains the same items as `a` but in a random order.

2. **Code Tracing**

Indicate what each will print. Place your answer (and nothing else) in the box below each block of code.

(a) (5 points) CT1

```
import copy
def ct1(a):
    b = copy.copy(a)
    c = copy.deepcopy(a)

    a[0][1] = "c"
    b[1][1] = 8
    c[0][0] = 9
    b[1] = ["d", 5.2]
    c.append(b[1].pop())

    print("a:", a)
    print("b:", b)
    print("c:", c)

g = [ ["a", 8.7], ["b", 3.4] ]
ct1(g)
print("g:", g)
```

(b) (5 points) CT2

```
def a(s):
    r = []
    for i in range(len(s)):
        if s[i] in "0123456789":
            r.append(i)
    print(r)
    return r

def b(a):
    r = []
    for j in a:
        r.append(str(j))
    print(r)
    return r

def ct2(s):
    x = a(s)

    r = ""
    for i in range(len(s)):
        if i not in x:
            r += s[i]
    print(r)
    t = "".join(b(x))
    d = int(t)
    d -= sum(x)
    print(d)

print(ct2("IL8VE112"))
```

3. **Reasoning Over Code**

For each function, find values of the parameters so that the roc function will return `True`. Place your answer (and nothing else) in the box below each block of code.

(a) (5 points) ROC1

```
def roc1(s):
    assert(len(s) == 9)
    i = 1
    r = ""
    for c in s:
        if c.isdigit():
            n = int(c)
            if n != i:
                return False
            elif s.find(c) != n:
                return False
            i += i
        else:
            r += c

    return r[::-1] == "sheep"
```

(b) (5 points) ROC2

```
def argh(n, b):
    c = 0
    while n > 0:
        c += n%10
        n //= b
    return c

def roc2(n):
    assert(n >= 100000 and n <= 999999)

    for c in str(n):
        if str(n).count(c) > 1 or c == '0':
            return False

    return argh(n,100)%2 == 1 and \
           argh(n,1000)%2 == 0 and \
           argh(n//10,100) == 10
```

4. (20 points) **Free Response: Fractional Prime**

   Note: For full credit, you may not use strings or lists for this problem. For a 10-pt deduction, you may use strings and/or lists.

   A fractional prime (coined term) is a number that is prime and the sum of its digits divided by the number of its digits is also prime.

   For example, 59 is a fractional prime. The number 59 itself is prime, and the sum of its digits divided by the number of digits ($\frac{5+9}{2} = 7$) is also prime. 1151 is also a fractional prime, as 1151 is prime and so is $\frac{1+1+5+1}{4} = 2$.

   Write the function `isFractionalPrime(n)` that takes an integer value n and returns `True` if the number is a fractional prime and `False` otherwise. Then, write `nthFractionalPrime(n)` that takes an integer value n and returns the nth fractional prime. You may write any additional helper functions you desire. **Note: You do not need to write `isPrime(n)`; you may assume that it has already been provided.**

   nthFractionalPrime(0) is 2, followed by 3, 5, 7, 13, 19, 31, 37, 59, 73, 1061, 1151, ...

Additional Space for Answer to Question 4

5. (20 points) **Free Response: Find Word**

In this problem you will write the function `findWord(L, word)` that takes as arguments a non-ragged, 2-dimensional list `L` of letters and a string `word`. The function searches `L` for `word`, returning the coordinates of the first letter of the word if it the word is present, and `None` otherwise. A word is considered present if it can be found forwards or backwards in a horizontal or vertical direction. You do not need to consider diagonal matches.

Consider the following example:

```
L = [
 ['t', 'k', 'z', 'd', 'd', 'f', 'g'],
 ['u', 'a', 'd', 'f', 'e', 'c', 'a'],
 ['g', 'o', 'd', 'r', 'e', 'a', 'q'],
 ['r', 'o', 'd', 'e', 'n', 't', 'w']
]

findWord(L, "cat") returns (1,5)
findWord(L, "need") returns (3,4)
findWord(L, "dog") returns (2,2)
findWord(L, "rodent") returns (3, 0)
findWord(L, "mouse") returns None
```

Note: You may assume `L` is not ragged. You may assume `L` and `word` will contain only lowercase letters. If a word is present more than once, you may return the location of any of the matches.

Additional Space 1 for Answer to Question 5

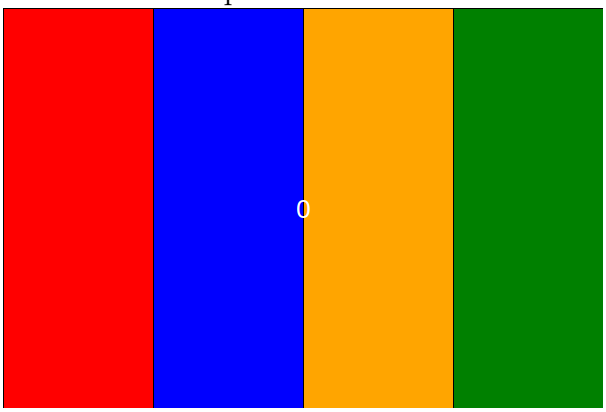
Additional Space 2 for Answer to Question 5

6. (25 points) **Free Response: Color Grid Animation**

Assuming the run() function is already written for you, write `init`, `keyPressed`, `mousePressed`, `redrawAll`, and `timerFired` so that when the animation is first run:

1. The screen is equally separated into four colors: red, blue, orange, and green.
2. The current score (initially 0) is displayed in the center of the screen in white.

Here is an example:



Game play proceeds as such:

1. Every four seconds, the colors rotate to the right. This means that while initially the colors are red, blue, orange, and green, after four seconds they will be green, red, blue, orange. (Every color moves to the right and the color that was furthest right becomes the first color.)
2. If the user clicks inside the red cell, they receive one point and the order of colors displayed is randomized.
3. If the user presses 'r' at any time, then the game starts over.
4. If the user presses the up arrow, then the time between color rotations increases by 1 second. (It should not be allowed to go above 10 seconds.)
5. If the user presses the down arrow, then the time between color rotations decreases by 1 second. (It should not be allowed to go below 1 second.)
6. If the user scores 10 points, then the game is over and the score is replaced with the text "You Win!"
7. When the game is over, nothing on the screen should change unless the game is reset using 'r'.

Make reasonable assumptions for anything not specified here. Do not hardcode values for data.width or data.height. You are only allowed to use one timer and one `timerFired` function. We recommend that, to save time writing, you abbreviate canvas, event, and data: use c, e and d, respectively.

You may make use of the `randomizeList` function from problem 1(e) and assume that it works, even if your implementation does not.

Additional Space 1 for Answer to Question 6

Additional Space 2 for Answer to Question 6

Additional Space 3 for Answer to Question 6