11/14/22

CSCI 220
Test III

Name: Coles Bowen

Turn off cell phones and anything that makes a noise.

Nothing is allowed on the desktop but the test, pens, pencils, erasers, and a drink.

No outside materials, including notes and calculators, are allowed.

Raise your hand if you have a question and wait for me to come to you. Please ask your question quietly.

Please write legibly. No points will be given for illegible writing.

With all code, be sure indentations are clear by using space appropriately. Do not write or draw lines indicating indentation. Correct indentation will not be assumed in ambiguous cases.

Be sure to clearly indicate data types but do not write the name of the data type itself. For example, writing "7" is sufficient to indicate this is a string.

In any code that you write on this test, you do not have to write comments. Just write enough code to do what is asked.

If the code requires import statements, you must write the import statement.

If more than one solution is provided for a question, I will count the one that earns the least amount of points.

You may not use any functions or syntaxes we have not covered in class. If you are unsure, please ask.

Please put all answers in the corresponding space provided below each question. **Do not write any answers on the back of any page or it will not be graded.**

not, and, or

1. What is the value of x after each of the following groups of statements?

a)
```
a = 15
b = ""
x = a or b
```
x = " "

b)
```
a = True
b = False
c = True
x = a or (b and c)
```
x = a or F

x = True

c)
```
a = []
b = "ice cream"
x = a and b
```
x = [ ]

d)
```
a = 7
b = "April"
x = a and b
```
x = "April"

file

day1: temp temp temp

day2: temp " " "

2. Heating and cooling degree-days are measures used by utility companies to estimate energy requirements. If the average temperature for a day is below 60, then the number of degrees below 60 is added to the heating degree-days. If the average daily temperature is above 80, the amount over 80 is added to the cooling degree-days. Write a function `degree_days(file_name: string) -> List[float]` that accepts the name of a data file and computes the running total of heating and cooling degree-days based on the data. The function should process the entire file, where each line contains any number of temperature readings for a single day and each reading is separated by a single space. After all the data has been processed, the function should return the heating and cooling degree-day totals in a list where the first element is the heating degree-days and the second element is the cooling degree-days.

if avg temp less than 60, 60 - avg temp added to heating deg days

above 80, avg temp - 80 added to cooling deg days

'temp temp temp'

['temp', 'temp', ...]

```
def degree_days(file_name):
    infile = open(file_name, 'r')
    heating_days = []
    cooling_days = []
    output = []
    for line in infile.readline():
        temp_list = line.split()
        sum = 0
        for temp in temp_list:
            sum = sum + float(temp)
            avg = sum/len(temp_list)
            if avg < 60:
                heating_days.append(60-avg)
            elif avg > 80:
                cooling_days.append(avg-80)

    for i in heating_days:
        output.append(i)
        for i in cooling_days:
            output.append(i)

    return output
```

3. For parts a) and b), write two classes according to the attached specifications, then complete the task in part c). Complete part a) and this page and part b) on the next page.
   a) `Color` class:

```
class Color:
    def __init__(self, red, green, blue):
        self.red = red
        self.green = green
        self.blue = blue

    def get_rbg(self):
        rgb_list = []
        return rgb_list.append(self.red, self.green, self.blue)

    def to_string(self):
        color = "reddish"
        if self.green > self.red and self.green > self.blue:
            color = "greenish"
            return color
        elif self.green = self.red and self.green == self.blue:
            color = "greenish"
            return color
        elif self.blue > self.green and self.blue > self.red:
            color = "blueish"
            return color
```

b) `Flower` **class:**

```
class Flower:
    def __init__(self, name, color):
        self.name = name
        self.color = color
        self.planted = False

    def pick(self):
        self.planted = False

    def plant(self):
        self.planted = True

    def get_color(self):
        return color.to_string()
```

c) Using the classes described in parts a) and b), create a "Rose" `Flower`, then plant it. Its color should be "redish", meaning more red than any other color, and each value of the color (red, green, and blue) must be a randomly generated number between 0 and 255 inclusive. Use the `randint` method from the `random` class to generate the random color values.
Hint: `randint(a, b)` - returns a random integer in range [a, b], including `a` and `b`.

```
from random import randint

r_g_b = (color(randint(0,255), randint(0,255), randint(0,255))
rose = Flower('Rose', r_g_b.get_color)
rose.plant()
```