CS 2316 Individual Homework 2 – Conditionals & Loops
Due: Wednesday, September 3rd, before 11:55pm
Out of 100 points

File to submit: HW2.py

Students may only collaborate with fellow students currently taking CS 2316, the TA's, and the lecturer. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc.

For Help:

- TA Helpdesk – Schedule posted on class website.
- Email TA's or use Piazza

Notes:

- **Don't forget to include the required comments and collaboration statement (as outlined on the course syllabus).**
- ***Do not wait until the last minute*** *to do this assignment in case you run into problems*
- **Read the entire specifications document before starting this assignment.**

Simple Functions
You will write a few python functions for practice with the language. In your HW2.py file, include a comment at the top with your name, section, GTId/Email, and your collaboration statement. Also, include each of the following functions below. For purpose of this homework, you may assume that all inputs will be valid.

1. **countLetter**
2. **gradeReplacement**
3. **findAvg**
4. **timeValueMoney**
5. **numHourglass**
6. **gpaCalculator**
7. **nutrition**
8. **clockTurtle**

1. countLetter(10pts)

Description:
Write a function that takes in two string parameters. The first will be a sentence, the second will be a letter. Your function will analyze the string and print out the number of occurrences of the letter. Note with caution that your function should be able to recognize both uppercase and lowercase letters! The format of the information to be printed can be found under “Test Cases”.

Parameters:
aString (String): A string
aLetter (String: A string

Return Value:
None

Test Cases:
1. countLetter(“so it goes”, ”s”) prints “The letter ‘s’ occurs 2 time(s).”
2. countLetter(“Would you kindly”, ”k”) prints “The letter ‘k’ occurs 1 time(s).”
3. countLetter(“CS is sOoOoOo fun!”, “o”) prints “The letter ‘o’ occurs 6 time(s).”

2. gradeReplacement (10pts)

Description:
Write a function that takes in a list of exam grades and returns the average after performing a grade replacement policy. The grade replacement policy takes the lowest grade from the list and replaces it with the second lowest grade in the list. There will be at least 2 numbers in the list, but there is no upper bound for the length of the list.

Parameters:
gradeList (List): A list of exam grades as integers

Return:
(Float) The average of all the exam grades after replacement

Test Cases:
1. gradeReplacement([100,90,80,70]) returns 87.5
2. gradeReplacement([100,100,100,100]) returns 100.0
3. gradeReplacement([90, 80, 80, 90, 85]) returns 85.0
4. gradeReplacement([30, 80, 44, 90, 85]) returns 68.6

3. findAvg (10pts)

Description:
Write a function that takes in a list. The elements in the list can be of any type (e.g. an integer, a float, a string, another list, etc.). This function should calculate the average of all the integers and floats contained in the list. That is, all non-number data types should be ignored. You may assume that at least one integer or float will be in each list.

Parameters:
findAvg(List): A list containing elements of various data types

Return Value:
(Float): The average of all numbers in the list

Test Cases:
1. findAvg([“hi”, 1.1, [2, 3], 15, {“32”:32}, 4, True]) returns 6.7
2. findAvg([True, False, 0, 1]) returns 0.5
3. findAvg([None, “32”, 10, “five”]) returns 10.0
4. findAvg([4242, 4010, 7646, [4311], “4133”]) returns 5299.333333
5. findAvg([32/4, 16.9, 500]) returns 174.96667

4. timeValueMoney(10pts)

Description:
Write a function that takes in a list containing integers/floats and a float between 0 and 1. The values in the list will represent cashflows(CF). Each cashflow's index+1 will be the time period(t) the cashflow is received. The function will return the present value of the cashflows based on the float parameter passed in (which represents the interest rate (r))using the formula:
PV = SUM(Cashflow/(1+ r)^t). For example given the list [10,20,30] and interest rate 0.1. The present value would be calculated as PV = 10/1.1 + 20/1.1^2 + 30/1.1^3

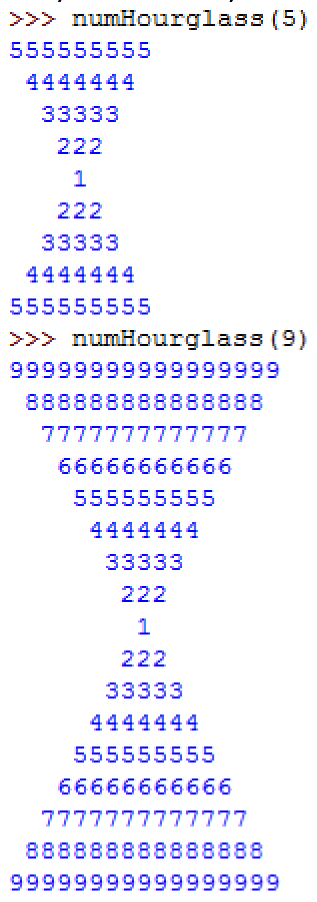
Parameter:
aList (List): a list representing cashflows
intRate (Float): a float representing the interest rate

Return Values:
(float): The present value of all cashflows rounded to two decimal places

Test Cases:
1. timeValueMoney([10,20,30], 0.1) returns 48.16
2. timeValueMoney([500,400,-30], 0.08) returns 782.08
3. timeValueMoney([330,-400,-450.32,20,1500.00], 0.06) returns 713.95

5. numHourglass(15pts)

Description:
Write a function that will take in an integer X and return an hourglass with a maximum width of 2X-1. Each row will be made of integers representing said row, see the output examples below for clarification.
Parameters:
aNum (Int): An integer representing the maximum width of the hourglass
Return Value:
None

Test Cases:
Notice that you have three 2’s, seven 4’s, nine 5’s, etc.

```
>>> numHourglass(5)
555555555
 4444444
  33333
   222
    1
   222
  33333
 4444444
555555555
>>> numHourglass(9)
99999999999999999
 888888888888888
  7777777777777
   66666666666
    555555555
     4444444
      33333
       222
        1
       222
      33333
     4444444
    555555555
   66666666666
  7777777777777
 888888888888888
99999999999999999
```

5. **gpaCalculator (20pts)**

Description:
Write a function that takes in 2 lists of equal length. The first list will contain numbers from 0-100 representing final grades in different classes. The second list will represent the number of credit hours each class grants upon completion. The indexes in both lists coincide. For example, if the first element in the first list is a 97 and the first element in the second list is 3, this data represents a 3 hour class that one received a grade of 97 in. Using this data you will need to calculate the number of quality points received from each class. Quality points are calculated as the grade point achieved in the class times the number of credit hours of that class. Grade point are determined as follows: 4 for [90,100], 3 for [80,90), 2 for [70,80), 1 for [60,70) and 0 for [0,60). Once the quality points have been determined, the grade point average is calculated as the sum of quality points over the sum of the credit hours.
Your function should return the calculated GPA.
Parameters:
gradesList (List): This list will hold finals grades (out of 100)
hoursList (List): This list will hold the amount of course hours granted
Return Value:
(Float): A floating point value representing the grade point average.

Test Cases:
1. gpaCalculator([87,81,91,99,93],[3,3,4,3,3]) returns 3.625
2. gpaCalculator([74,90,88,40],[3,4,4,3]) returns 2.4286
3. gpaCalculator([100,30,50,75,90],[3,3,1,4,2]) returns 2.153846
4. gpaCalculator([59,82,74,79,66],[4,3,3,1,3]) returns 1.42857

7. nutrition(10pts)

Description:
This function will take in a list of tuples. Each tuple will contain two integers which represent a specific food's calorie and protein content (in grams), respectively. The tuples will always be of the form (calories,protein). This list represents all the food eaten in a day. The function will also take in an integer representing weight (in lbs.). The goal of the function is to determine whether or not the amount of protein consumed was greater than half of the body weight while not exceeding 2500 calories.

If these conditions are met, the function should print out the grams of protein consumed, as well as the number of calories under the 2500 threshold you are in the following format: "Nice! You consumed X grams of protein and still have Y calories left!"

If these conditions are not met due to insufficient protein intake, the function should print out the difference between half the body weight and grams of protein consumed, along with the number of calories consumed in the following format: "Sorry, you still need to eat X grams of protein, and have already consumed Y calories."

If these conditions are not met due to exceeding 2500 calories, the function should print out the amount of protein consumed and the difference in the amount of calories consumed and the 2500 allotment in the following format: "Sorry, you ate X grams of protein but consumed Y too many calories."

If none of the conditions are met, the function should print out the difference in half the body weight and grams of protein consumed, as well as the amount of calories over 2500 in the following format: "Sorry, you still need to eat X grams of protein, but consumed Y too many calories."

Parameters:
aList (List): A list of tuples representing the calorie and protein content of various foods
aNum (integer): An integer representing body weight

(in lbs.)

Return Values:
None

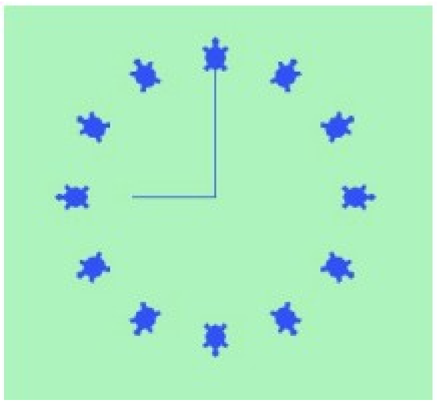
Test Cases:
1. nutrition([(180,3), (750,10), (493,20), (390, 0), (910, 16)], 150) prints “Sorry, you still need to eat 26.0 grams of protein, but consumed 223 too many calories.”
2. nutrition([(330,14), (543,22), (599,34), (120, 11)], 119) prints “Nice! You consumed 81 grams of protein and still have 908 calories left!”
3. nutrition([(500,8), (443, 2), (200,10), (840, 15), (732, 9), (901, 14)], 110) prints “Sorry, you ate 58 grams of protein but consumed 1116 too many calories.”
4. nutrition([(400,13), (130,4)], 178) prints “Sorry, you still need to eat 72.0 grams of protein, and have already consumed 530 calories.”

8. **clockTurtle (15pts)**

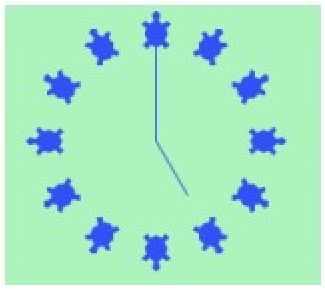
Description:
Write a function that uses the turtle module to draw a clock with a given clickHour, as the short hand of clock, and aNum, as the radius. You may assume that the long hand of the clock will stay at 12 at all times. You do need to draw the clock layout using the turtle module. At each hour position (12, 1, 2, 3, 4, 5, etc.), make your turtle leave a stamp of itself. (You can change the turtle shape if you want).
Parameters:
clockHour (Integer): an integer between 1 and 12 representing the short hand of the clock
aNum (Integer): radius of the clock

Return Values:
None

Examples:
clockTurtle(9,100)



clockTurtle(5,70)



Grading Rubric

countLetter (10pts)

- Loops through input string	2pts
- Identifies lowercase letters	2pts
- Identifies uppercase letters	2pts
- Keeps track of the number of occurrences of letter	2pts
- Prints correct information in the correct format	2pts

gradeReplacement(10pts)

- Correctly identifies the lowest grade	2pts
- Correctly identifies the second lowest grade	2pts
- Replaces lowest grade with second lowest grade successfully	4pts
- Returns correct average	2pts

findAvg(10pts)

- Loops through input list	1pts
- Identifies which elements are valid ints/floats	1pt
- Keeps track of valid values	2pts
- Counts number of valid values	1pt
- Correctly sums values	1pt
- Correctly calculates average of valid values	2pts
- Returns correct average	2pts

timeValueMoney (10pts)

- Loops through input list	1pt
- Correctly implements PV formula	5pts
- Rounds PV to two decimal places	2pts
- Returns correct PV	2pts

numHourglass (15pts)

- Correct number of rows	5pts
- Each row is the correct number	5pts
- Not hardcoded	5pts

gpaCalculator(20pts)

- Takes in two correctly formatted parameters	2pts
- Loops through both lists	2pts
- References correct elements in both lists when calculating qlty.pts	4pts
- Correctly calculates quality points for each grade-hours pair	6pts
- Correctly calculates GPA	4pts
- Returns GPA in the correct format	2pts

nutrition(10pts)

Function header and parameters are correct	2pts
Correct output for sufficient protein/insufficient calories case	2pts
Correct output for insufficient protein/sufficient calories case	2pts
Correct output for insufficient protein/insufficient calories case	2pts
Correct output for sufficient protein/sufficient calories case	2pts

clockTurtle(15pts)

Takes in 2 parameters	1pt
Long hand stays at 12	3pts
Short hand is drawn at correct hour position	5pts
Short hand is (recognizably) shorter than long hand	1pts
Turtle is stamped at each hour position	2pts
Turtles are stamped in the correct orientation	1pts
The clock size changes as radius changes	2pts