CS 2316

Individual Homework 1 – Python Practice

Due: Tuesday, May 24th, before 11:55 PM
Out of 100 points

Files to submit: **1. HW1.py**

This is an INDIVIDUAL assignment!

Collaboration at a reasonable level will not result in substantially similar code. 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. You should not exchange code or write code for others.

For Help:

- TA Helpdesk – Schedule posted on class website.
- Email TA's or use T-Square Forums

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.

Part 1 – Simple Functions

You will write a few python functions for practice with the language. In your HW1.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.

1. hPaToAtm
2. trapezoidArea
3. footballScore
4. bodyMassIndex

Function Name: **hPaToAtm**

Parameters:

None

Return Value:

None

Description:

1. Write a user-interactive function to convert hectoPascals (hPa) to Atmospheres (Atm). The conversion equation is as follows (A is Atmospheres and H is hectoPascals):

$$A = \frac{H}{1013.25 \frac{hPa}{Atm}}$$

2. Get the number of hectoPascals from the user; make sure to use a descriptive prompt so the user knows what to enter (e.g. "Enter number of hectoPascals"). You may assume that the user will enter a valid float (e.g. 34.4) and do not have to do error checking.
3. Convert the number entered by the user to atmospheres using the formula above.
4. Print the calculated number of atmospheres; be sure to add a label to the display value so the user knows what the value means (e.g. display "35 atmospheres" instead of just 35)

Function Name: **trapezoidArea**

Parameters:

None

Return Value:

None

Description:

Write a user-interactive function to calculate the volume of a sphere.

1. Get the bases, height, and the unit of measurement of the trapezoid from the user; make sure to use a descriptive prompt so the user knows what to enter
2. Calculate the area of the trapezoid using the following formula:
$$A = \frac{(b_1 + b_2) \cdot h}{2}$$
3. Print the calculated area; be sure to add a label to display value so the user knows what the value means with the units entered by the user (e.g. display "Area of the trapezoid is 35 units-squared" instead of just 35, where units would be replaced with whatever unit of measurement the user entered).

Function Name: **footballScore**

Parameters:

None

Return Value:

None

Description:

Write a user-interactive function to make a guess at how a team earned the points they did for a particular score in an American football game.

1. Get the team's score as an integer from the user; make sure to use a descriptive prompt so the user knows what to enter.
2. Compute the number of touchdowns, field goals, two-point conversions, and extra points that could make up the score the user entered (We are assuming no safeties). You should compute each of these numbers in the above order using the following information:
   - There are 6 points in a touchdown
   - There are 3 points in a field goal

- There are 2 points in a 2-point conversion
- There is 1 point in an extra point.
- The mod (a.k.a. remainder) operator in python is % and will show the remainder left after an integer division. It IS useful for this problem!

3. Print the calculated number of touchdowns, field goals, two-point conversions, and extra points on **one line;** be sure to add appropriate labels to the display values so the user knows what the value means (e.g. display "3 touchdowns, 1 field goals, 1 two-point conversions, 0 extra points"). Also note that for low scores (below 6), you may get a score that is not legal according to the rules of American football; this is OK.

Function Name: **bodyMassIndex**

Parameters:

None

Return Value:

None

Description:

Write a user-interactive program to calculate a person's BMI, body mass index.

1. Ask for the person's weight in pounds; make sure to use a descriptive prompt so the user knows what to enter.
2. Ask for the person's height in inches; again, use a descriptive prompt.
3. Calculate the person's BMI using the information they provided and the formula below (you will need to convert the formula into a legal python equation):

$$\text{BMI} = \frac{weight(\text{lb})}{height^2\,(\text{in}^2)} \times 703$$

4. Print the result. The value must be formatted (both the wording and the number of decimal places) EXACTLY as follows: "Your BMI is 24.9." (where the value for BMI represents whatever you calculated based on the inputs). *You may have to use string formatting to remove extra decimal places from your calculation.*

Part 2 – Complex Functions

Now you will write a few python functions that return data for practice with the language. Also include each of the three following functions.

Function Name: hPaToAtmRedux

Parameters:

1. hpa - a number representing the number of hectoPascals to be converted; note, may be a float or an int, depending on what value is sent in when the function is called

Return Value:

A floating point number representing the number of atmospheres calculated.

Test Cases:

1. hPaToAtmRedux(243) --> 0.239822354

Description:

Write a function that takes in a number of hectoPascals, calculates the corresponding number of atmospheres, and ***returns*** the result. Use the conversion listed above.

Function Name: **calcDistance**

Parameters:

1. x_1 – a number representing the x coordinate of the first point as an integer
2. y_1 – a number representing the y coordinate of the first point as an integer
3. x_2 – a number representing the x coordinate of the second point as an integer
4. y_2 – a number representing the y coordinate of the second point as an integer

Return Value:

A floating point number representing the distance between the two points.

Test Cases:

1. calcDistance(5,10,15,20) --> 14.142135623730951
2. calcDistance(-1,15,7, -3) --> 19.697715603592208

Description:

Write a function calcDistance that will calculate and return the distance between the two given points as a floating point number. Use the distance formula given below:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Note that to successfully complete this problem, you will need to take the square root. To do this, you must import the math module using "import math" before you use the square root function, which is named sqrt. You can call functions from modules you import by saying modulename.functionname(params) instead of just saying functionname(params).

Function Name: **makeSmores**

Parameters:

1. grahamCrackers - an integer representing the number of graham crackers
2. marshmallows - an integer representing the number of marshmallows
3. chocolate - an integer representing the number of bars of chocolate

Return Value:

An integer representing the amount of smores you can make.

Test Cases:

1. makeSmores(6, 2, 3) --> 2
2. makeSmores(1, 2, 2) --> 0
3. makeSmores(11, 20, 20) --> 5

Description:
Write a function that returns the number of smores you can make based on the number of graham crackers, marshmallows, and chocolate specified as inputs. Assume that it takes the following to make 1 smore:

- 2 graham crackers
- 1 marshmallows
- 1 bar of chocolate

Note that the number of smores must be a whole number; you cannot have 1 1/2 of a smore. For example, if you have 12 graham crackers, 4 marshmallows, but only 3 bars of chocolate, you can only make 3 smores (despite the fact that you have enough graham crackers and marshmallow to make 4 smores). Likewise, if you have 3 graham crackers, 100 marshmallows, and 300 bars of chocolate, you can still only make 1 smore.
Hint: Python has a built-in function called min that takes in a comma separate list of numbers and returns the minimum value. E.g. min(5, 3, 7) returns the number 3. This may be useful in coming up with your solution.
You can solve this problem without using conditionals!

Grading

You will earn points as follows for each function that works correctly according to the specifications.

Simple Function

hPaToAtm	**5**
trapezoidArea	**10**
footballScore	**20**
bodyMassIndex	**15**

Complex Functions

hPaToAtmRedux	**15**
calcDistance	**15**
makeSmores	**20**

If your function works, but prints a value when it is supposed to return it, or returns it when you are supposed to print it, the TA will **deduct half of the points** for that function.