

CS1026: Assignment 2 – Volume Calculator

Due: October 16th 2019 at 9:00pm

Weight: 8%

Learning Outcome:

By completing this assignment, you will gain skills relating to

- using loops,
- using functions,
- using lists in Python,
- creating and using Python modules,
- testing,
- following program specifications and requirements.

Task:

In this assignment, you will write a **complete** program in Python that computes the volume for cubes, pyramids and ellipsoids. Your program should make use of functions, loops, and lists.

Your program will consist of two files: one is a module, **volume.py**, which computes volumes and the other is a main program, **main.py**, which uses the functions in module **volume.py**. The program is expected to prompt the user for a type of object (e.g. a “cube”) and validate that is one of the expected object types before computing the volume. In addition, your program should keep track of each volume that is calculated. After the user chooses to “quit” your program should display the volume for all the shapes **AND** you should use the function **summarize** to output your lists to a file; the function **summarize** is provided for you in a module **summarize.py**. You are to test your program on three different test cases which are provided for you.

Functional Specifications:

1. Your main module, **main.py**, **should handle the prompting and input for the different shapes and the output**. Specifically, it should:
 - Prompt the user for the shape they are interested in, check to make sure that their input is valid. **Valid input options are:** “cube” or “c”, “pyramid” or “p”, “ellipsoid” or “e”, “quit” or “q”; ***you should accept the input in any combination of upper and lower case letters***. If the user enters an invalid option, your program should continue to prompt the user for a correct choice.
 - Your program should continue to prompt the user for different shapes until the user enters “quit” or “q”.
 - Your program should prompt the user for the necessary dimensions for each of the respective shapes. ***You may assume that the user enters positive integer values and so you DO NOT have to check the input for valid values.***

- Once your program has a valid input, it should then use the correct function in **volumes.py** to prompt the user for the specific values for that shape and compute the volume (see below).
- It should then add the resulting volume to a list of volumes for the same kinds of shapes (i.e., all cube volumes will be in the same list).
- Once the user has entered “quit” (or “q”), your program should sort each list from lowest to highest. It should then print the list of volumes for each shape **in sorted order** and pass the sorted lists to the function summarize.

If there are no calculations are done for a specific shape then a statement should be printed. For instance if there were no pyramid calculations the output would be as shown below.

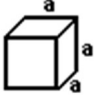
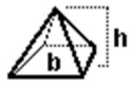
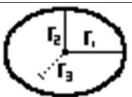
You have reached the end of your session.
 The volumes calculated for each shape are:
 Cube: 125.0, 1953.125
 Pyramid: 200.0
 Ellipsoid: No shapes entered

If the user enters **quit** option before actually calculating any volumes print a message indicating that no volumes were computed, such as:

You have reached the end of your session.
 You did not perform and volume calculations

2. Your module, **volumes.py**, should contain the functions for computing the volumes.

- Each of your functions should calculate the volume of the shape **and return that value (it should not print a message)**; volumes for the different shapes are computed as follows:

Shape	Volume
cube	 $volume = a^3$ where a is the length of a side
pyramid	 $volume = \frac{1}{3}b^2h$ where b is base length and h is height
ellipsoid	 $volume = \frac{4}{3}pi * r_1 * r_2 * r_3$ where pi is π and r is used to represent each radius

3. The function **summarize** in the module **summarize.py** is used to output the results to files. The function **summarize** has 4 parameters: **summarize(clist,plist,elist,testNumber)**. The parameters are, respectively: the list of cube volumes, the list of pyramid volumes and the list of ellipsoid volumes. The last parameter is the test case number (**Hint: when you write your program prompt for the test case number before you start to prompt the user for shapes and dimensions**) . **You must test your program on the following three test cases:**

- Test case 1: use the following (in order) for the different shapes:
 - *Cube: length of side: 11*
 - *Pyramid: base = 10; height = 12*
 - *Ellipsoid: radius1 = 10; radius2 = 6; radius3 = 9*
- Test case 2: use the following (in order) for the different shapes:
 - *Ellipsoid: radius1 = 6; radius2 = 7; radius3 = 8*
 - *Ellipsoid: radius1 = 1; radius2 = 1; radius3 = 1*
 - *Cube: length of side: 1*
 - *Pyramid: base = 1; height = 3*
 - *Cube: length of side: 9*
 - *Cube: length of side: 5*
 - *Pyramid: base = 12; height = 12*
- Test case 3: use the following (in order) for the different shapes:
 - *Pyramid: base = 10; height = 13*
 - *Pyramid: base = 1; height = 1*

You will notice that the function **summarize** outputs a file with your results. It creates a filename with the test case number that you provide. For the three test cases you should end up with three files: **Asgn2TestCase1**, **Asgn2TestCase2** and **Asgn2TestCase3**.

Non-functional Specifications:

1. Include brief comments in your code identifying yourself, describing the program, and describing key portions of the code.
2. Assignments are to be done individually and must **be your own work**. Software may be used to detect cheating.
3. Use Python coding conventions and good programming techniques, for example:
 - Meaningful variable and function names,
 - Conventions for naming variables and constants,
 - Use of constants where appropriate,
 - Readability: indentation, white space, consistency.

Submission requirements: What to submit:

You should submit your files **volumes.py** and **main.py** as well as the three files **Asgn2TestCase1**, **Asgn2TestCase2** and **Asgn2TestCase3** with the results of your testing. Make sure you upload (attach) the files; **DO NOT** put the code inline in the textbox.

What You Will Be Marked On:

- Functional specifications:
 - Are there modules volumes.py and main.py and are they defined according to specifications?
 - Is the function **summarize** used correctly?
 - Does the program compute according to the specifications?
 - Does the program handle invalid input for the type of object?
 - Is there an effective use of functions?
 - Is the output according to specifications?
 - Have the files with the test cases been created and submitted?
- Non-functional specifications: as described above
- Assignment submission: via the OWL, though the assignment submission in OWL.