LAB 2L

Description

This exercise gives you practice with simple input, output operations as well as arithmetic expressions using basic arithmetic operators.

Problem

Write a program that prompts the user for three numbers, the side, radius and height (in that order) needed to compute the area and volume of several shapes. The shapes for computing areas are: Square, Circle, Cube Surface, Sphere Surface, Cylinder Surface and Cone Surface. The shapes for computing volumes are: Cube, Sphere, Cylinder, and Cone. The formula for computing each one of these is shown below.

Areas:

Square	side^2
Circle	pi x radius^2
Cube Surface	6 x side^2
Sphere Surface	4 x pi x radius^2
Cylinder Lateral Surface	2 x pi x radius x height
Cone Lateral Surface	pi x radius x side

Volumes:

Cube	Side^3
Sphere	4/3 x pi x radius^3
Cylinder	pi x radius^2 x height
Cone	1 /3 x pi x radius^2 x height

Solving it Step by Step

Here's a recommended way to proceed:

1. Create a directory called 2L under your cs410 directory
2. Copy the file $public/2L/shapes.c to your directory (using the cp command). Two ways to copy the file over:
   ```
   cp $public/2L/shapes.c  .
   cp ~cs410c/public/2L/shapes.c  .
   ```
3. Compile shapes.c (using gcc) and run it (a.out) to see what it does already.
4. Currently the program prompts for just one input, the side. It also computes only one formula i.e. to compute the area of a square.
   Note the way we always use **scanf** to input information: The format specifier always starts with "\n" to avoid the problem of flushing buffer information out to avoid problems while running the program. **scanf** is typically very fussy, any extra spaces or special characters in the format string will cause it behave strangely.
5. First extend it to include taking the 2 other inputs and print the 2 values input by the user.
6. Save, compile and run the program and make sure you can correctly display all 3 values input by the user.
7. Continue to extend the program by writing the appropriate statements to compute and print all other areas and volumes.
   a. Start out by computing just one value at a time and verify the results before proceeding to do the rest.

    b. To use the mathematical constant “pi” used in most of the above formulas you MUST define a constant with the “# define” directive at the top of the program (I use 3.1415 as the value of pi in my program)
    c. Use at least two different types of formatting a floating point number i.e. with different widths and number of decimal digits.
    d. Your output should resemble mine (doesn’t have to be exactly the same, but very similar). Be sure to add your name at the top of the program as a comment

8. Your output values may differ slightly from mine depending on how you are using the value of “pi” as well as how you might be formatting your output as opposed to my solution.
9. Testing your program is as important as writing the program itself. If the program produces incorrect results it could be as bad as not writing one at all. Test your program with different user inputs.
10. To assist you with testing your program you can compare your program output with mine located under the same public folder for 2L. I have created a few files named out1, out2, etc. for showing you the program output for different inputs. Use the same inputs as mine and verify that your program output matches with mine (as closely as possible).
11. Remember to submit your program using the code 2L before you get ready to leave.
    Example (if you file is called shapes.c):
    ```
    submit 2L shapes.c
    ```
12. Remember to logout before you leave.

Grade Key

Name, comments	**3**
Proper inputs and prompts in the correct order	**6**
Constant definition for pi	**3**
Correctly computes each area or volume (7 points each)	**63**
Output in reasonable format (includes at least 2 formats)	**10**
Works correctly for multiple input combinations	**15**