CS 111 – Introduction to Computer Science – Fall 2017

Lab Assignment #7

Functions * *(30pts)*

Due Date: at 11:59pm on Friday, March 30.

The purpose of this lab is to give you an opportunity to work with user defined functions.

Before getting started with the lab, copy the entire `lab7` folder from the course folder (`H:\Compsci\givens\cs111`) to your `U:\cs111` folder.

Cylinder

For the first exercise of the lab, you are going to complete the `cylinder.py` program. This program is supposed to prompt the user for the size of a right circular cylinder and then compute and display its volume and total surface area. But the program is incomplete.

You are to complete the program by defining the two missing functions that compute the volume and surface area of the cylinder. The volume of a cylinder is computed as

$$V = \pi r^2 h$$

and the surface area as

$$A = 2\pi r(r + h)$$

Your program should be written to the following specifications:

- Modify the file prolog to include your name.
- Define and implement the two missing functions below the main routine.
- Include a comment immediately above each of the two functions that you write to briefly describe the purpose of the function, its parameters, and return value.
- Your program should have no user input or output other than what is already provided.

After completing the program, be sure to save the file.

■

*Based on the labs of Dr. Rance Necaise

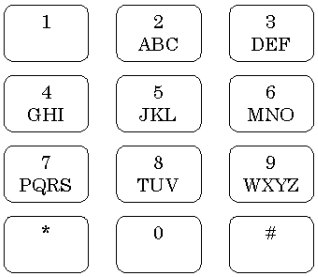
More Functions

In previous labs you completed several tasks using if-statements and loops. For this lab, you will convert some of those tasks into functions, and each function will be stored in the module called `moreFunctions.py`. In order to convert the tasks to functions, you may need make a few changes to your previous code. Pay close attention to each function's parameters, how those parameters are used, and whether the function should output with a print statement or return a value to the user (unless specified, you can assume functions require a return value). You will then create another file, `testMoreFunctions.py` that imports your module and contains a main method to test it.

 Open the `moreFunctions.py` module and complete the following functions. Create another program called `testMoreFunctions.py` that imports everything in `moreFunctions` and tests each of your methods.

- `slope(x1,y1,x2,y2)`
  This function receives two coordinates of a line segment (x_1, y_1) and (x_2, y_2) and computes and returns the slope. If the slope cannot be computed, however, it will return `False` as an error.

- `telephone(letter)`
  This function receives a letter, converts it to its corresponding digit on the telephone, and returns that digit. If an invalid character is entered, the function should return -1 as an error.



- `printSquares(num)`
  Receives an integer `num` and prints all the squares less than n. For example, if `num` is 100, it should print 0 1 4 9 16 25 36 49 64 81 (one per line). Because this function is an output function, it does not require a return value.

- `factorial(num)`
  Receives an integer `num` and if it is greater than, or equal to, 0, *returns* the factorial of that number. (Note that $0! = 1$.) If the function receives a negative number, it should *return* -1 as an error. This function should not print. If you want to print the result, capture the output in a variable from your main method.

`moreFunctions.py` should be written to the following specifications:

- Include an appropriate file prolog and appropriate comments throughout the program, including descriptions for each function.

- The module will not contain a *main* function. Nor should it contain any executable code outside of the functions. To test your functions, use the module `testMoreFunctions.py`.

- No functions should request input from the user. All functions should use their parameters to make all calculations.
- Only the `printSquares` function should output to the console with a print statement. All other functions should *return* their result.
- Include a comment immediately above each of the functions that you write to briefly describe the purpose of the function, its parameters, and return value.

After completing the programs, be sure to save the file.
■

Finishing Up

When you are finished with the lab, you need to show me that your code runs and correctly computes the solution for each part of the lab. Also, you need to submit the source files for grading. To submit the files, find the lab assignment on Canvas and upload the two files:

- `cylinder.py`
- `moreFunctions.py`
- `testMoreFunctions.py`

Remember, all of the files must be named exactly as indicated above, with the same case and with no spaces or special characters.