# CSC121 Lab 04: Functions Part 1

## Goals

In this lab assignment, students will demonstrate the ability to:

* Define functions and invoke them
* Pass data to a function when it is invoked
* Pass multiple arguments to a function
* Use Python's standard library module random

NOTE: This lab document is a copyrighted work of Wake Tech Community College and the course instructor. Any posting of this document outside of Wake Tech is considered a copyright violation. Students who post these documents outside of Wake Tech are subject to academic and possible legal actions.

## Instructions

In this lab, you will demonstrate your mastery of void-returning functions (functions which do not return any values) and the built-in Python module random.

Follow the instructions in each problem and submit the specified files. All problems will require that Python code be submitted as well as screenshots that prove the programs have been executed in PyCharm.

Problems 1 and 2 will start with code provided by the instructor that needs to be completed. Problem 3 will consist of a program that you create from scratch that meets the problem specification.

The **main** function: In this lesson you have been learning about functions, and you learned about the convention of using a main function. Starting with this lesson and going forward, **all programs** that you write from scratch should have a main function that contains the top-level logic of your code. Otherwise, you should follow the instructions provided in the lab document.

## Problems

### Problem 1

In this problem, you're given a partially completed program, and you need to update and complete the program to produce the desired output.

A landscaping company has determined that for every 50 square feet of garden space, 1 bag of mulch and 2 hours of labor are required.

The company charges $40.00 per hour for labor plus a flat $100.00 project fee.

This program asks the user to enter the square feet of garden space to be mulched and the price of mulch per bag.

The program will then display:

* The number of bags of mulch required
* The hours of labor required
* The cost of the mulch
* The cost of the labor (per hour charges plus project fee)
* The total cost of the landscaping job.

NOTE: This company charges the customer for quarter days (2 hours) and does not charge for less than that. So, you should base the hours of labor on the number of bags of mulch. For example, if a job requires 6 bags of mulch, it will also require 12 hours of labor.

* The instructor has provided a file called **Lab04P1-FillThisIn.py**. Download that file and rename it **Lab04P1.py**.
* Copy that file into your PyCharm project.
* Change the program header to include your name and the date.
* Replace every instance of **"--Fill this in--"** with correct code that will enable the program to do the required calculations and display the results.
* Note there are GLOBAL CONSTANTS at the top of the program. You should use those constants wherever it makes sense to use them.
* You should eventually be able to run the program with no errors and produce results like the sample output. Note that all monetary values should have 2 digits after the decimal point.

Sample Output:

Enter garden space in square feet: 100

Enter mulch price per bag: 5.00

Bags of mulch: 2

Hours of labor: 4

Mulch charges: $10.00

Labor charges: $260.00

Total cost: $270.00

Run this program. Take a screenshot with the results. Name the screenshot **Lab04P1-ouput.jpg**.

Submit both files, **Lab04P1.py** and **Lab04P1-output.jpg**, to Blackboard for credit.

### Problem 2

In this problem, you are given a partially completed program, and you need to update and fill in the rest of the program to produce the desired output.

This problem is like other programs we've done for Trish's Swap Shop. The program asks the user for the number of books, DVDs, and games being purchased, and then it will output the cost for each item and the total with tax.

Values:

* Books are $2.25 each.
* DVDs are $4.35 each.
* Games are $5.00 each.
* Tax is 6.5%

Sample output:

Enter the number of books: 7

Enter the number of DVDs: 3

Enter the number of games: 4

Books: $15.75

DVDs: $13.05

Games: $20.00

Subtotal: $48.80

Tax: $3.17

Total: $51.97

* The instructor has provided a file called **Lab04P2-FillThisIn.py**. Download that file and rename it **Lab04P2.py**.
* Copy that file into your PyCharm project.
* Change the program header to include your name and the date.
* Replace **# -- FILL THIS IN -- #** with correct code that will enable the program to produce output like what is shown above.
* Use the function name and parameter names indicated in the program comments.
* Use Global Constants within the function where appropriate to make your code more readable.
* All monetary values should be formatted with 2 digits after the decimal point.
* Run this program.
* Take a screenshot of the results.
* Name the screenshot **Lab04P2-ouput.jpg**.

Submit both files, **Lab04P2.py** and **Lab04P2-output.jpg**, to Blackboard for credit.

### Problem 3

In this problem, you will be writing a program from scratch that simulates throwing a special die with faces numbered 1 through 20. Along with a number, each face also has a special symbol or effect associated with it:

* If the number on the die is 20, it's considered a "CRITICAL HIT!"
* Otherwise, each number is associated with a symbol as follows:
  + If the number divides by 4 exactly, the symbol is a Sword.
  + If the number divided by 4 has a remainder of 1, the symbol is a Shield.
  + If the number divided by 4 has a remainder of 2, the symbol is a Spell.
  + If the number divided by 4 has a remainder of 3, the symbol is a Potion.

At the start of the program, the user will be asked how many times they wish to roll the die. They must enter a number between 5 and 15. The program will then simulate rolling the die that number of times.

Sample Output:

How many times do you want to roll the die? 3

Enter a number between 5 and 15.

How many times do you want to roll the die? 7

Roll 1: 19 ==> Potion

Roll 2: 3 ==> Potion

Roll 3: 20 ==> CRITICAL HIT!

Roll 4: 14 ==> Spell

Roll 5: 15 ==> Potion

Roll 6: 13 ==> Shield

Roll 7: 16 ==> Sword

Thanks for playing!

The program should consist of TWO functions:

main – This function will be the main routine, and it should do the following:

* Prompt the user to enter the number of times they want to roll the die. If the user inputs a number less than 5 or greater than 15, the program should continue to ask the user for a valid number within this range.
  + Remember that a while loop can be used for input validation.
* Once the program has a valid number from the user, it should pass this number as an argument when calling the roll\_die function.

roll\_die – This function should have one parameter, rolls, which represents the number of times to roll the die.

* Use a for loop to roll the die the specified number of times.
* Use the randint function from the random module to generate a random number between 1 and 20 inclusive.
* Print the number of the current roll and the result of the roll as demonstrated in the Sample Output.
  + The number of the current roll should start at 1.
  + The result of the roll should include the number and either the associated symbol or "CRITICAL HIT!" in the case of the number 20.
* Once the for loop has finished, print a message thanking the user for playing.

Run this program. Take a screenshot of the results. Name the screenshot **Lab04P3-ouput.jpg**.

Submit both files, **Lab04P3.py** and **Lab04P3-output.jpg**, to Blackboard for credit.

## Grading Rubric

### Grading rubric for Problem 1 (30 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [20 points]
* Screenshot demonstrates student executed the program [5 points]

### Grading rubric for Problem 2 (30 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [20 points]
* Screenshot demonstrates student executed the program [5 points]

### Grading rubric for Problem 3 (40 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [30 points]
* Screenshot demonstrates student executed the program [5 points]