

## CS-119 Chapter 1 Lab

---

### Student Learning Outcomes

---

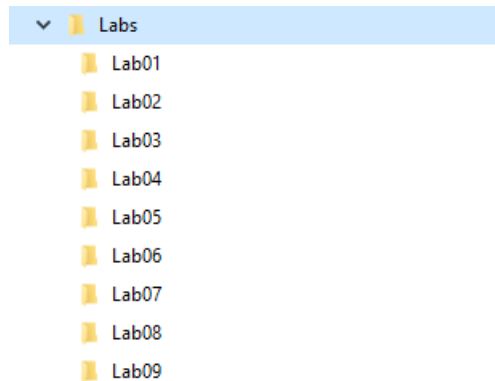
- Introduction to the Python IDLE IDE
- Introduction to Python applications
- Introduction to writing pseudocode
- Variables
- Comments
- Arithmetic calculations
- Sequential code
- Console input and output

### Overview

---

This lab provides an introduction to programming in Python. You will be creating 2 relatively simple Python console applications using Python IDLE. It also gives you an opportunity to apply some of the terminology and concepts presented in chapters 1 & 2 of the PLD text.

Before starting on any lab work for this class, take a few minutes to create a folder structure for your labs like the one shown below. Being organized is a critical survival skill in software development! If you are using the classroom lab computers, be sure to create this on your USB drive and be sure to save to it! Saving on the C: drive of the lab computers is a disaster waiting to happen!



## Exercise 1: A Guided Python Application – Multiply 2 Numbers

---

For most of the labs in this class, the first exercise will be a guided walk through where we will apply the chapter concepts to develop a working Python application.

In this exercise, you will be guided through the steps of analyzing a problem statement, writing pseudocode, and creating a working Python application using Python IDLE (Integrated Development and Learning Environment). Be sure to complete this exercise as you will be repeating these steps and writing the same (or very similar) code for the other exercises. Be aware, **practice is the key here. The more you do, the easier it gets!**

Step one is **always** to analyze the problem and figure out a plan of attack. To save time, we'll continue the problem we looked at in the chapter one lecture notes and the final problem statement is pasted below for reference. Please review the chapter one notes if you need to review the more detailed analysis and pseudocode.

Problem statement from chapter one notes: ~~It's a bright, sunny day and Cuyamaca Learning Center has hired you as a consultant to develop an application that will accept two input numbers from the user and display the product of the two values numbers.~~

Break it down into input, process and output. This is a simple problem and we can easily break it down into “workable pieces” using this approach.

What input do we need? We need 2 numbers

What is our process? We need to multiply the 2 numbers. Our formula is:  $\text{product} = \text{number1} * \text{number2}$

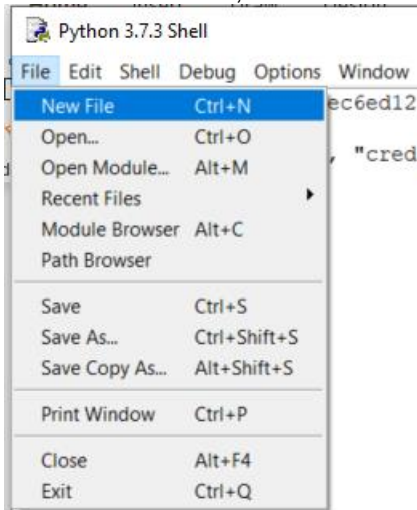
What is our output? We need to display the product

The next step is to write the pseudocode. Being basically a text document, it can be done with any text editor, word processor or even as a spreadsheet. Python IDLE can be used as well. It works out to be a nice way to keep things organized and score valuable points when you turn your work in for a grade.

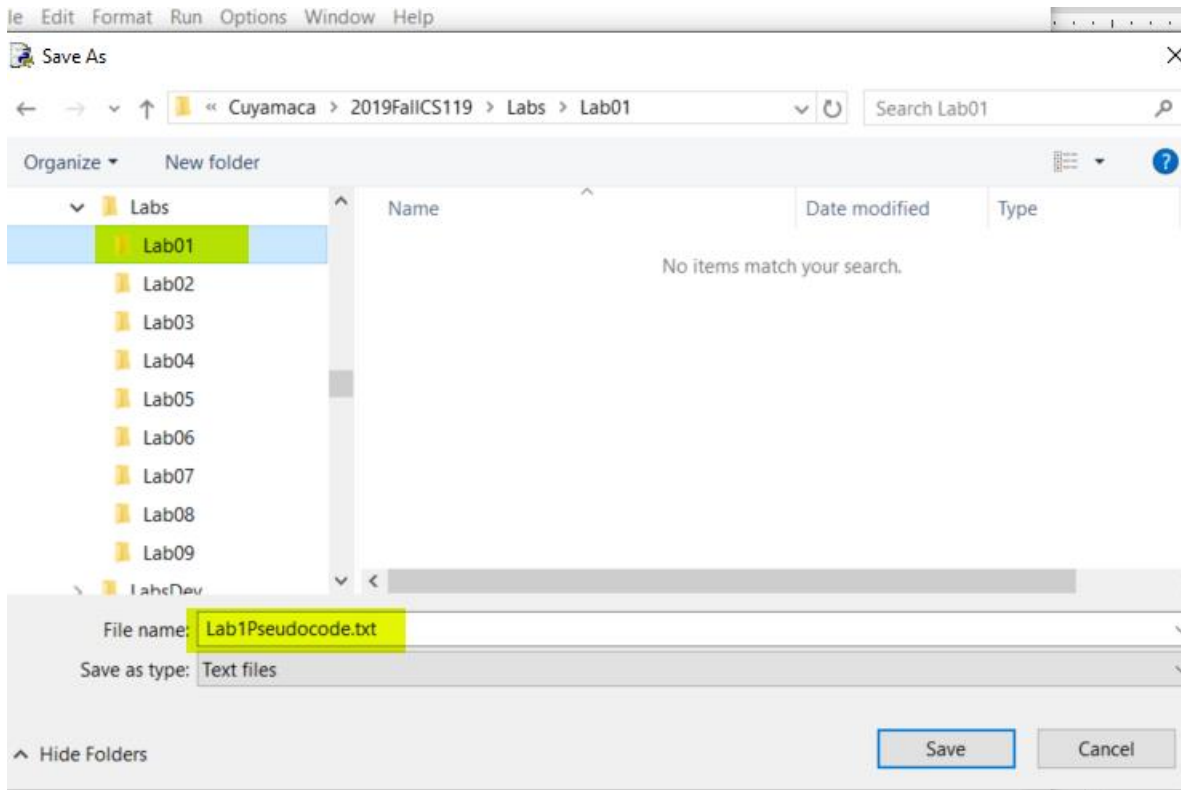
### Steps

---

1. Start Python IDLE.
2. From the File menu, click New File.



3. Once the new file is created, save it right away as a **text file**. Be sure to save it in your Lab 1 folder so you get points for it!



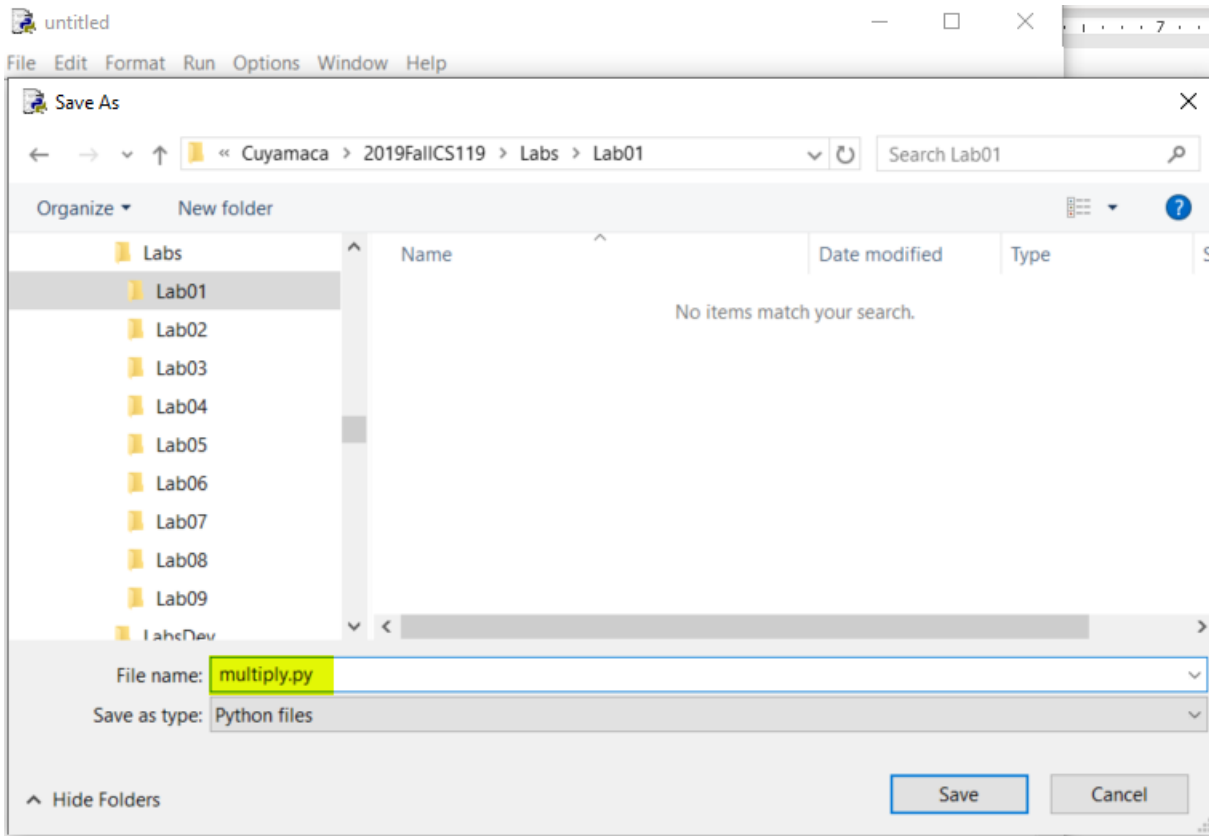
4. Write your pseudocode. Keep in mind there is no single “correct” way of doing it. This is perhaps what makes writing pseudocode confusing and difficult at first. Develop a style that helps *you* understand the problem and how you plan to code the solution. The sample shown below is a *suggested* approach.

```
*Lab1Pseudocode.txt - C:/Cuyamaca/2019FallCS119/Labs/Lab01/Lab1Pseudocode.txt (3.7.3)*
File Edit Format Run Options Window Help
# input
get number1
get number2

# process
product = number1 * number2

# output
display product
```

5. Create a new file and save as multiply.py to your Lab 1 folder. Note that Python source files must have the .py extension.



6. Write the code shown in the Python code screen shot below. Pay close attention to letter case. Note the following things we will be doing for virtually every lab exercise:

Be sure to code the *shebang line* `#!/user/bin/env python3`. You don't need to pseudocode this.  
In comments at the top, put in your name, course number and the lab exercise information. See sample below.

Declare and initialize variables. You don't have to show this in your pseudocode but it may help to do so.  
Use the `input()` method for getting user input  
Use the `print()` method for displaying output.  
Use the `int()` method to convert text to an integer.  
Use the `float()` method to convert text to a float.  
Use the `str()` method to convert an integer value to string.  
The program code should follow the pseudocode!

Here's a sample comment header at the top with your name, the course number, and a brief description of the lab exercise.

In Python, anything after a single hash (#) character is a comment.

\*MPG.py - C:/Cuyamaca/2019FallCS119/LabsDev/Lab01/MPG.py (3.7.3)\*

File Edit Format Run Options Window Help

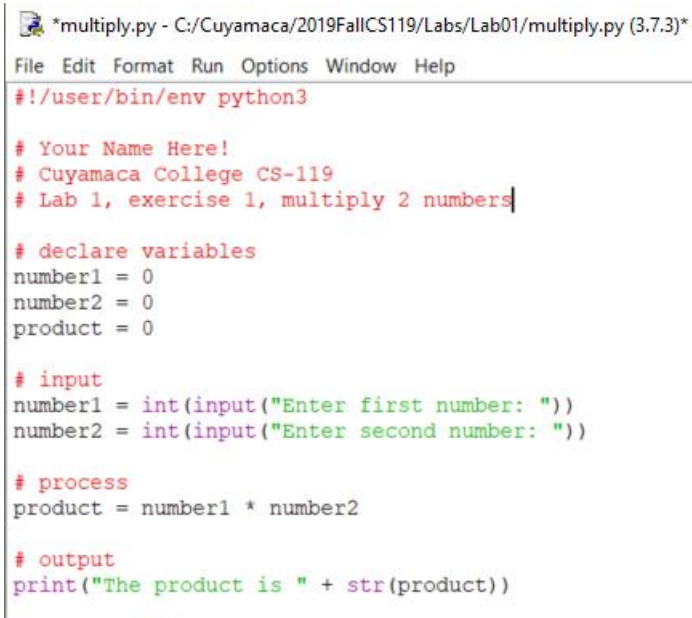
```
#!/user/bin/env python3
```

```
# Your Name Here!
# Cuyamaca College CS-119
# Lab 1, exercise 1, multiply 2 numbers
```

In the code below, a couple things to note:

We declare variables using a concise, yet descriptive name. Python determines the data type by the initial value.

We use a single equal sign (=) for assigning literal values (i.e., 0, etc.) or the result of an expression to a variable.



```
*multiply.py - C:/Cuyamaca/2019FallCS119/Labs/Lab01/multiply.py (3.7.3)*
File Edit Format Run Options Window Help
#!/user/bin/env python3

# Your Name Here!
# Cuyamaca College CS-119
# Lab 1, exercise 1, multiply 2 numbers

# declare variables
number1 = 0
number2 = 0
product = 0

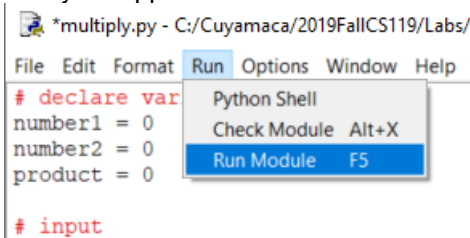
# input
number1 = int(input("Enter first number: "))
number2 = int(input("Enter second number: "))

# process
product = number1 * number2

# output
print("The product is " + str(product))
```

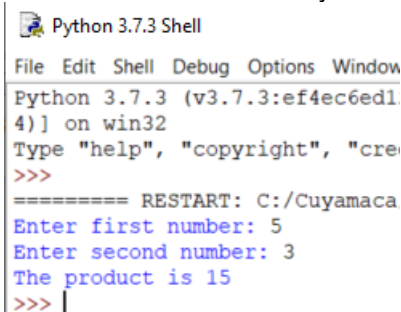
7. Be sure to save your work!

8. Run your application. Select Run Module from the Run menu.



```
*multiply.py - C:/Cuyamaca/2019FallCS119/Labs/
File Edit Format Run Options Window Help
# declare var
number1 = 0
number2 = 0
product = 0
# input
```

9. Choose test values where you can easily verify that your program is producing the correct output.



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window
Python 3.7.3 (v3.7.3:ef4ec6ed1.
4)] on win32
Type "help", "copyright", "cre
>>>
===== RESTART: C:/Cuyamaca
Enter first number: 5
Enter second number: 3
The product is 15
>>> |
```

10. If you get a correct output, congratulations! You have just successfully created your first Python application!

## Exercise 2: Kilometers to Miles Calculator

---

In this exercise, you will create another Python application to convert kilometers to miles. Your application must allow the user to enter the number of kilometers and then convert and display miles. 1 kilometer = 0.62 miles.

Before writing any code, write some pseudocode as it will help you break down the problem into understandable pieces. It's worth points! To save time, you may add your pseudocode for this exercise to the pseudocode file you created in exercise 1.

You will create a new file in the Python IDLE the same way you did in exercise 1. The file name is left to your discretion but choose a reasonably concise, yet descriptive name for your project such as KmToMiles. Make sure you save the file in your Lab 1 folder.

Like exercise 1, be sure to code the shebang line and comments with your name, course No. and lab exercise.

Hint: For your input, you will need a line of code something like this: `km = float(input("Enter kilometers: "))`

### Exercise 3: Miles to Kilometers Calculator

---

In this exercise, you will create another Python application to convert miles to kilometers. This conversion is just the opposite of what you did in the last exercise! Your application must allow the user to enter the number of miles and then convert and display kilometers. 1 kilometer = 0.62 miles.

Before writing any code, write some pseudocode as it will help you break down the problem into understandable pieces. It's worth points! To save time, you may add your pseudocode for this exercise to the pseudocode file you created in exercise 1.

You will create a new file in the Python IDLE the same way you did in exercise 1. The file name is left to your discretion but choose a reasonably concise, yet descriptive name for your project such as MilesToKm. Make sure you save the file in your Lab 1 folder.

Like exercise 1, be sure to code the shebang line and comments with your name, course No. and lab exercise.

Hint: For your input, you will need a line of code something like this: `miles = float(input("Enter miles: "))`

Before writing any code, write some pseudocode as it will help you break down the problem into understandable pieces. Also, it's worth points! To save time, you may add your pseudocode for this exercise to the pseudocode file you created in exercise 1.

You will create the Python file the same way you did in exercise 1.

### Grading Criteria:

---

Deliverable	Points	Breakdown
Guided exercise pseudocode	4	<b>Input, process and output clearly identified. Clear programming logic and calculation formulas.</b>
Guided exercise program code	4	<b>Complete, code is clear, descriptive variable names, appropriate use of comments</b>
Guided exercise run	2	<b>Compiles, runs, produces correct output</b>
Km/Miles pseudocode	5	<b>Input, process and output clearly identified. Clear programming logic and calculation formulas.</b>
Km/miles program code	10	<b>Complete, code is clear, descriptive variable names, appropriate use of comments</b>
Km/miles run	5	<b>Compiles, runs, produces correct output</b>
Miles/Km pseudocode	5	<b>Input, process and output clearly identified. Clear programming logic and calculation formulas.</b>
Miles/Km program code	10	<b>Complete, prompts any inputs are clear, code is clear, descriptive variable names, appropriate use of comments</b>
Miles/Km run	5	<b>Compiles, runs, produces correct output</b>
Lab Total	50	