# CSC121 Lab 01: Getting Started/Input, Processing, Output

## Goals

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

* Create a Python project in PyCharm, and create, edit, and run Python files.
* Document Python programs using comments.
* Design computer programs to solve problems.
* Use variables to store data.
* Write statements to get input, perform calculations, and display output.

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

This section provides you an overview of the problems you'll be working on in this lab, and some other important information you need to complete the lab.

For Problem 1, you will need to create a PyCharm project and create a simple Python program which is properly documented. You will run this program in the PyCharm Terminal and capture a screenshot demonstrating that you executed this program.

For Problem 2, you will copy a file into your PyCharm project and demonstrate that you can run the Reformat Code function in PyCharm.

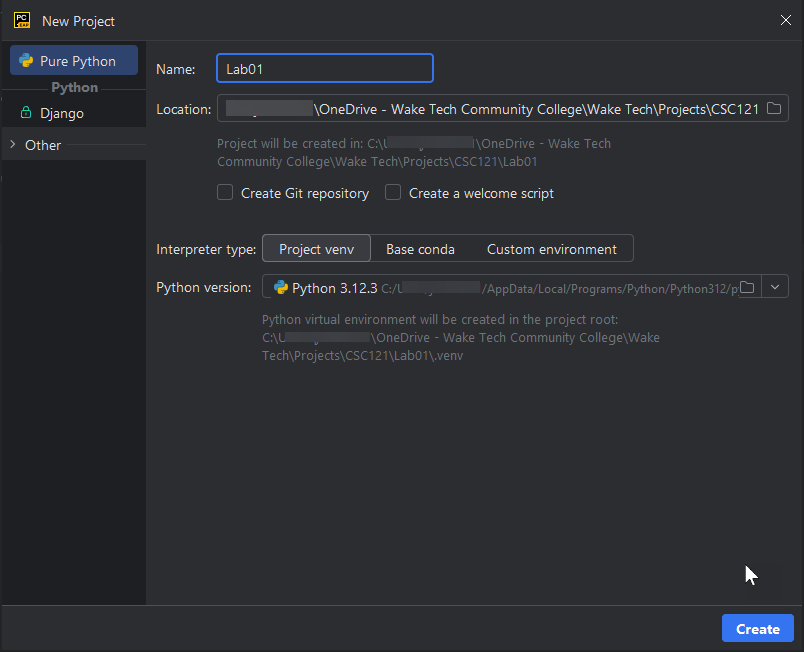
Problems 3 and 4 are more typical of the labs that you have for the rest of the semester. In Problem 3, you are given a partially completed program, and you will supply what's missing to complete the program. In Problem 4, you will write a program from scratch based on the problem specification.

Make sure you have reviewed the Lab Assignment Guidelines before completing this assignment. In particular, every file should have a program comment header at the top, and any string formatting should be done with f-strings.

Remember that you are only allowed to use concepts we have learned in the course up to this point. That means to only use what we have learned in Lesson 1.

## Before You Begin

* You should create a folder somewhere on your computer where you will put all your CSC 121 lab assignments. That will make it easier for you to find files when you need them. A good folder name might be **CSC121**. The folder you create will be the Location when you create a new project.
* For each lab assignments, you should create a new PyCharm project with an appropriate Name. For this assignment, an appropriate Name might be **Lab01**.
* Here’s an example of how the New Project dialog might be filled in:



* Leave the interpreter type as “Project venv”. Although we don’t need virtual environments, it’s easier to just leave this type selected.
* Make sure the Python version is the version you installed. If it isn’t, click on the dropdown on the right-side of the Python version line to select it.

## Problems

### Problem 1

This problem will make sure you understand how to add comment headers to Python programs.

* In your new project, create a Python file called **Lab01P1.py**. If you’re not sure how to do that, review the information in the “Using PyCharm” document.
* Add a comment header to the top of the file and include your name, the date, and a brief description of the program. Creating a comment header was discussed in the Writing and Running Lab Assignments SoftChalk Lesson.
* Add the following code to the file:

import datetime

name = input('What is your name? ')

print(f'Welcome to CSC121 {name}!')

print(f'Today is {datetime.date.today():%B %d, %Y}')

print('I hope you learn a lot of Python this semester!')

* Run this program. Notice that the program is asking for your name (the line with **input)**, so type your name and hit Enter.
* Take a screenshot of the Terminal or Run window that includes the line showing where you started the program run with the results.
* Name the screenshot **Lab01P1-ouput.jpg**.

Submit both files, **Lab01P1.py** and **Lab01P1-output.jpg**, to Blackboard for credit.

### Problem 2

It is important to write well-formatted programs to make it easier for you to debug and for others to review.

* The instructor has provided a file called **Lab01P2-ReformatCode.py**. Download that file and rename it to **Lab01P2.py**.
* Copy that file into your PyCharm project.
* Change the program header to include your name and the date.
* Use PyCharm's **Reformat Code** feature to improve the formatting of that file.

After you have completed the reformat code operation, submit **Lab01P2.py** to Blackboard for credit.

You do not need to execute the code for this problem, however, feel free to execute the code to see what it does. You'll learn how to create similar programs before you reach the end of this course.

### Problem 3

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.

* The instructor has provided a file called **Lab01P3-FillThisIn.py**. Download that file and rename it **Lab01P3.py**.
* Copy that file into your PyCharm project.
* Change the program header to include your name and the date.
* Replace the four instances of "--Fill this in--" with correct code:
  + Ask the user for the number of miles. (This number could be a float.)
  + Ask the user for the speed in MPH. (This number could be a float.)
  + Calculate the travel time.
  + Display the travel time in hours and format the number displayed to 2 decimal places. HINT: Use an f-string!

Sample Output:

Enter number of miles: 150.0

Enter speed in MPH: 70.0

You should cover that distance in 2.14 hours.

* NOTE: Your program should display 2 decimal places of precision. You should use **f-strings** to accomplish that precision. Any other method will result in points being deducted.
* All Fill-this-in problems should ONLY replace the parts indicated by the template and the rest of the template should NOT be changed.
* Run this program using the PyCharm Terminal or using the Run button in PyCharm.
* Take a screenshot of the Terminal or Run window that includes the line showing where you started the program run with the results.
* Name the screenshot **Lab01P3-ouput.jpg**.

Submit both files, **Lab01P3.py** and **Lab01P3-output.jpg**, to Blackboard for credit.

### Problem 4

Trish at Bargain Swap Shop has hired you to write a Python program that will help her calculate what a customer should pay. She has a very simple price structure:

* Books are $2.25 each.
* DVDs are $4.35 each.
* Games are $5.00 each.

With every purchase, Trish also must charge 6.5% sales tax on the total.

Create a file named **Lab01P4.**py. Write a program that calculates the cost for the total purchase:

* Add a comment that indicates if the next section is output, input, or processing.
* Ask the user to enter the number of books, DVDs, and games being purchased.
* Add a comment that indicates if the next section is output, input, or processing.
* Calculate the total before tax.
* Calculate the amount of sales tax on the total.
* Calculate the total after tax.
* Add a comment that indicates if the next section is output, input, or processing.
* Display the total before tax, the sales tax, and the total after tax.

NOTE: Your program code should implement these steps IN THE SAME ORDER as listed above.

All monetary values should be output with a precision of 2 digits after the decimal point. You should use what was presented in this lesson to achieve that result, that is, use **f-strings**. Points will be deducted otherwise.

The prompts for the user should include a space at the end of the prompt so there is a clear separation between the output prompt and where the user types their response.

Sample Output:

Enter the number of books: 7

Enter the number of DVDs: 4

Enter the number of games: 6

Cost before tax: $63.15

Sales tax: $4.10

Cost after tax: $67.25

Run this program. Take a screenshot of the Terminal or Run window that includes the line showing where you started the program run with the results. Name the screenshot **Lab01P4-ouput.jpg**.

Submit both files, **Lab01P4.py** and **Lab01P4-output.jpg**, to Blackboard for credit.

### Bonus Opportunity – 15 Points

* Copy **Lab01P3.py** to another file and name it **Lab01Bonus.py**.
* Rewrite **Lab01Bonus.py** so that it takes it calculates the whole number of hours and the remaining minutes and outputs the results.
* The program does NOT have to change the nouns to account for 1 hour or 1 minute. In other words, the sentence can still read as “1 hours and 1.00 minutes” and does not need to change to singular nouns.
* Submit Lab01Bonus.py to Blackboard for credit. A screenshot is not required.
* To receive the full bonus,
  + The program must execute with no runtime errors.
  + The program must perform the correct calculation.
  + The number of hours should be displayed as a whole number, and the minutes should be displayed with 2 digits after the decimal point.
  + The program comments should also be updated where appropriate.

Sample Output:

Enter number of miles: 150.0

Enter speed in MPH: 70.0

You should cover that distance in 2 hours and 8.57 minutes.

## Grading Rubric

### Grading rubric for Problem 1 (15 points)

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

### Grading rubric for Problem 2 (10 points)

* Program demonstrates correct use of Reformat Code feature [10 points]

### Grading rubric for Problem 3 (25 points)

* Program is well-formatted and has a correct header [5 points]
* Program does execute correctly and produces correct results [15 points]
* Screenshot demonstrates student executed the program [5 points]

### Grading Rubric for Problem 4 (50 points)

* Program has a well-formatted and correct header [5 points]
* Program has correct input statements identified with a comment [14 points]
* Program has correct processing statements identified with a comment [12 points]
* Program has correct output statements identified with a comment [14 points]
* Screenshot demonstrates student executed the program [5 points]

### Grading Rubric for Bonus (15 points)

* The program has been correctly converted as specified above and all comments and text displayed to user has been appropriately updated [15 points]