# Lab Exercise 1: Using IDLE and Tip Calculator

## Part 1: Opening Debugger

Open up IDLE, and click on "Debug" on the toolbar at the top. From the dropdown, select "Debugger" to open a new window; this is the Debugger window.

If you return to Python shell window and click on the "Debug" toolbar dropdown, you should now see a checkmark next to "Debugger." Additionally, the Python shell window should now have written "[DEBUG ON]".

## Part 2: Using the debugger

Open up a new python file window by using the CTRL+N shortcut, or by going to FILE > New Window. For the purpose of this lab, we will work with a simple python program dealing with arithmetic and simple variables. Using the debugger is very helpful, because it shows you the current value of any and every variable you have.

Copy and paste the following code into your new python window, making sure to remove any tabs or spaces from the beginning of the lines:

```
a = 5
b = 2
c = a + b
c = c + a * 2
c = c + 1
```

Now that you have the debugger open, run the code that you just pasted in by hitting F5, or by going to Run > Run Module. Any changes you have made to a program will need to be saved before the program can be run, and so if you haven't saved already, IDLE will ask you to save your new file.

Once you've run the file, the Python shell window will jump into focus, and the debugger will pop up. Keep clicking on "Step" - Python will run one line of code at a time, showing the values of the variables as your code is executed.

Specifically, notice that, as your program executes, the variables a, b, and c will appear under the section named "Locals." The label "Locals" is a common shortening of "local variables" which is just a collection of every variable that exists at the current location in your code. If you also look at the middle select window, you can see that, as you click the step button, the line number will change:

```
> '__main__'.<module>(), line 3: c = a + b
```

to

```
> '__main__'.<module>(), line 4: c = c + a * 2
```

This will show you where you currently are within your program! Run through your program to make sure that you can follow how each variable is being computed and that each line of the Python code makes sense.

NOTE: When you step to a line of code, that line of code has not yet been executed. For instance, when you are currently on line 3, you should notice that "Locals" does not contain the variable 'c' yet; only when you hit "step" again will the debugger execute that code and create the new variable. In the beginning, it might feel like you are looking at everything one step behind!

## Part 3: Finding Errors

Now that you know how to work the debugger, let's see how it responds to errors.

Go to the python file you were previously working on and replace the code with the following:

```
a = 1
b = 0
c = a / b
d = c + "2"
```

Make sure the debugger is open and once again run the program. Then go step by step through the program.

As you will notice, a highlighted error message will pop up complaining about a "ZeroDivisionError" and the program will not finish running.

Errors like a "ZeroDivisionError" will cause a program to crash, so you won't be able to look at what the rest of the program does until you fix it!

Replace the code with the following and debug it again:

```
a = 1
b = 0
```

```
c = a * b
d = c + "2"
```

Once again the program will crash due to an error, this time because strings (a word) can't be added to an integer (a whole number).

Using the debugger in this way will let you find and correct errors as they appear.

## Part 4: Tip Calculator Application

You will now code your first application in Python. The task of the application is to calculate a tip table for a restaurant bill. You will collect the input values to use in the calculation from the user.

1. Open a new file in IDLE (or the text editor of your choice) and named it **lab1_tip_calculator.py**
2. You need to read in the total price of the bill. Provide an input prompt to the user that says “Please provide the total amount for the bill: “.
    a. Note: You will need to convert the value returned by the **input** function since it will be a string and we need a **float** to perform calculations. Thus, you will need to only type numbers and a decimal point to have the program run properly.
3. Tips can range from 15% for average service, 20% for good service, and 25% for superior service. Store a variable with each of these values in it.
    a. Remember that percentages are actually decimal numbers (15% -> 0.15).
4. After the user has provided their input, calculate the tip for average, good, and superior service and store the values in variables. Then print the values of those variables to the user.
    a. You can use the following format string to provide output to the user:
    b. ‘Your bill was {:.2f}: Avg Tip = ${:3.2f}, Good Tip = ${:3.2f}, Great Tip = ${:3.2f}’
    c. Hint: A format string has placeholders in curly braces { }. You can place the **.format** method after any string and supply it with a series of values corresponding to the number of braces. In this case, your answer might be something like this:
        i. `print('Your bill was {.2f}: Avg Tip = ${3.2f}, Good Tip = ${3.2f}, Great Tip = ${3.2f}'.format(1, 2, 3, 4))`
5. [OPTIONAL]: If time permits, add an additional input prompt and variable to allow a user to split the tip between multiple people. If the user enters 4 for the party size, the tip calculations will be divided by 4 before they are presented.
6. When you are done, show your lab assistant your code, save a copy of it (one for each of you if you are working in pairs), and submit them to CourseWeb. **You must submit a file to CourseWeb under the Lab 1 assignment in order to get credit for attending the lab.**