

CSCI 1411: Fundamentals of Computing
Lab 8
Due Date: October 20, 2023

Name: Brandon Perez

Goals:

- Learn about debugging your Python code using IDLE debugger.

Development Environment: IDLE

Deliverables:

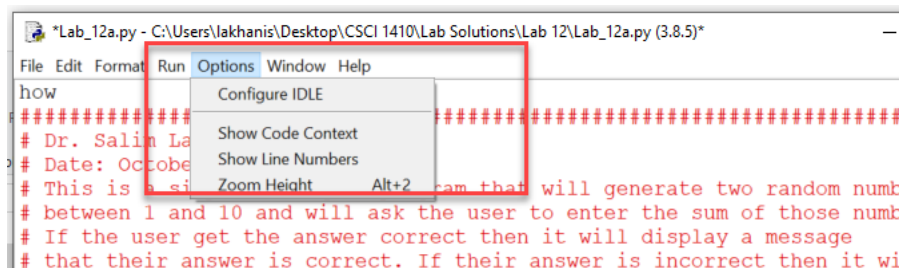
- 1) This completed document with required screen shots and algorithms.
- 2) Python file created for the first and second parts of the lab. Name the file using the following format: lastnameLab08Part1.py and lastnameLab08Part2.py.

How to take a screen shot:

- For a Windows 10: Use Snipping Tool to copy and press CTRL + V to paste screen shot.
- For Mac: Press Shift + Command (⌘) + 4 to copy and press Command (⌘) + V to paste screen shot.

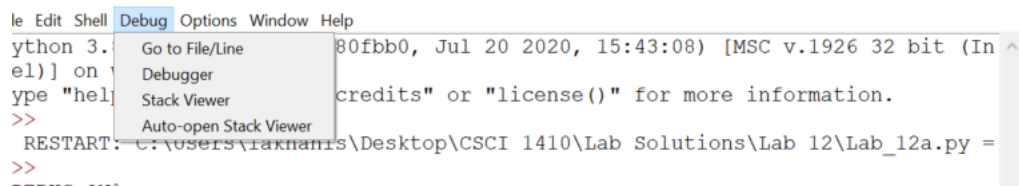
Part I – Skill Practice

- **Note: Take two screen shots at two different points as you follow the steps given below to run your debugger.**
- In this portion of the lab you will follow along with your instructor. Please do not get too far ahead of your instructor as they explain the use of debugger.
- Start IDLE
- Open the file Lab_8a.py and save it as lastnameLab08Part1.py
- Select “Show Line Numbers” in the Options menu of your Editor window to display the line numbers. This will help you with following this tutorial.

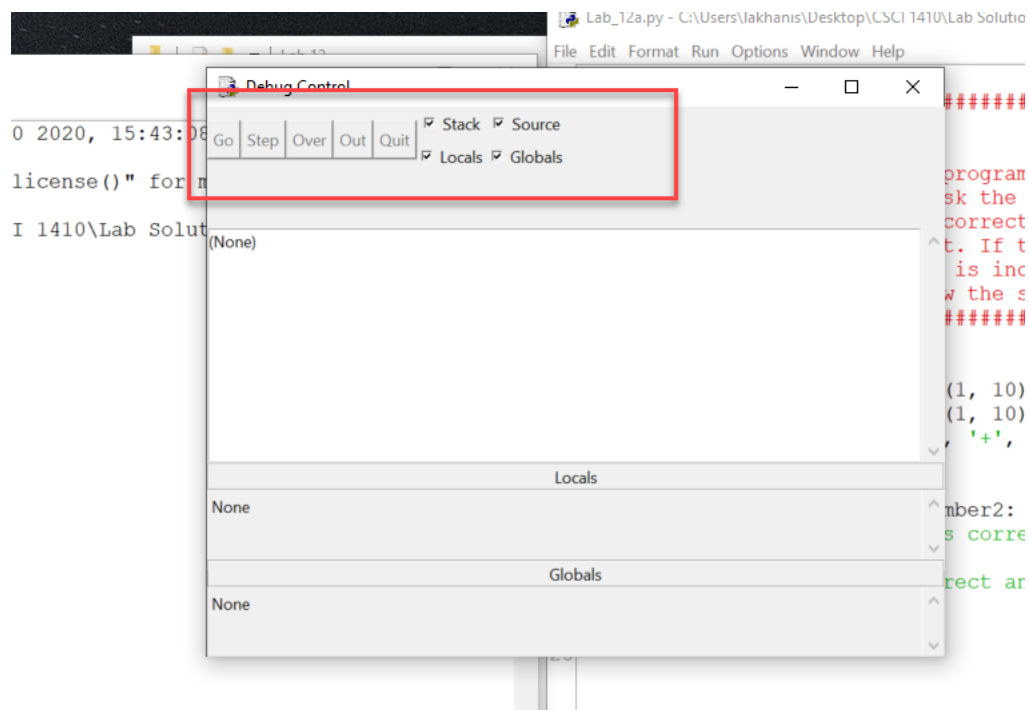


- This is a simple math quiz program that will generate two random numbers between 1 and 10, it will display those numbers and ask the user to enter the sum of those numbers. If the user answers the question correctly then it will display the message that the answer is correct. If the user answers the question incorrectly then it will display the message Nope with the correct answer.

- Run the program. This program has a bug and it will display Nope every time you run this program. It will display Nope even if you get the answer correct. We will use a debugger to find the error.
- Start the debugger by selecting Debugger from the Debug menu:

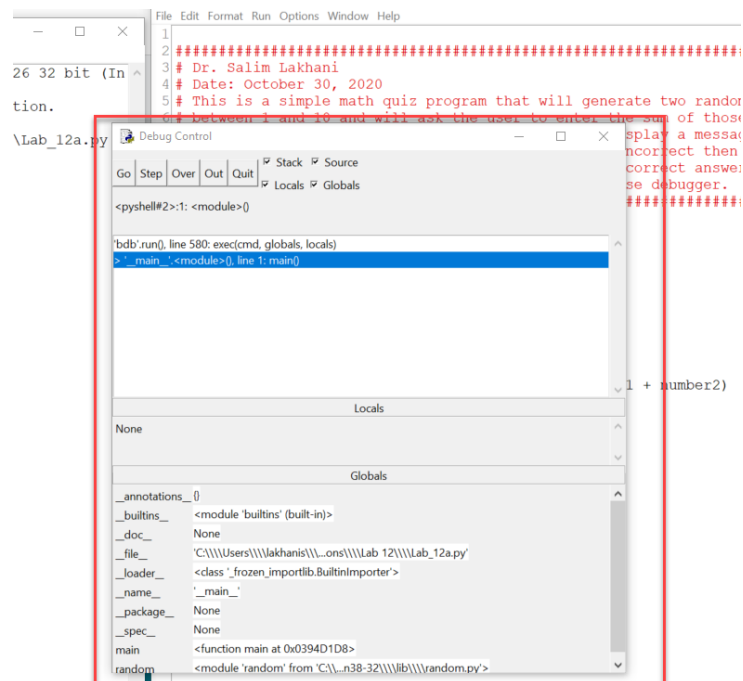


- This will open the Debug Control window.

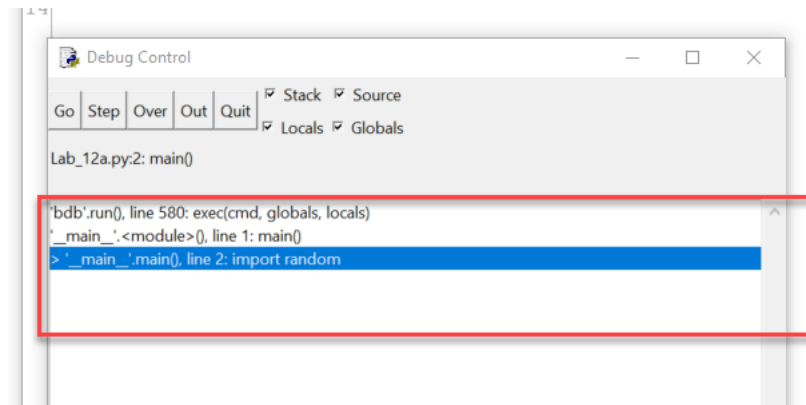


- Make sure that you select all the following check boxes: Stack, Locals, Source, and Globals.
- The debugger lets you execute one instruction at a time. This is called stepping. There are two options: you can either click on Step or Over to execute the current Python instruction. If your statement is going to call a function then Step will step into the function and Over will execute the function but will not take you into the function. You can also click on the Out button to finish execution of current function and jump back to the calling function. We will look at some examples later in this lab. The Go button will execute the code from the current location either to the end of the program or a break point.

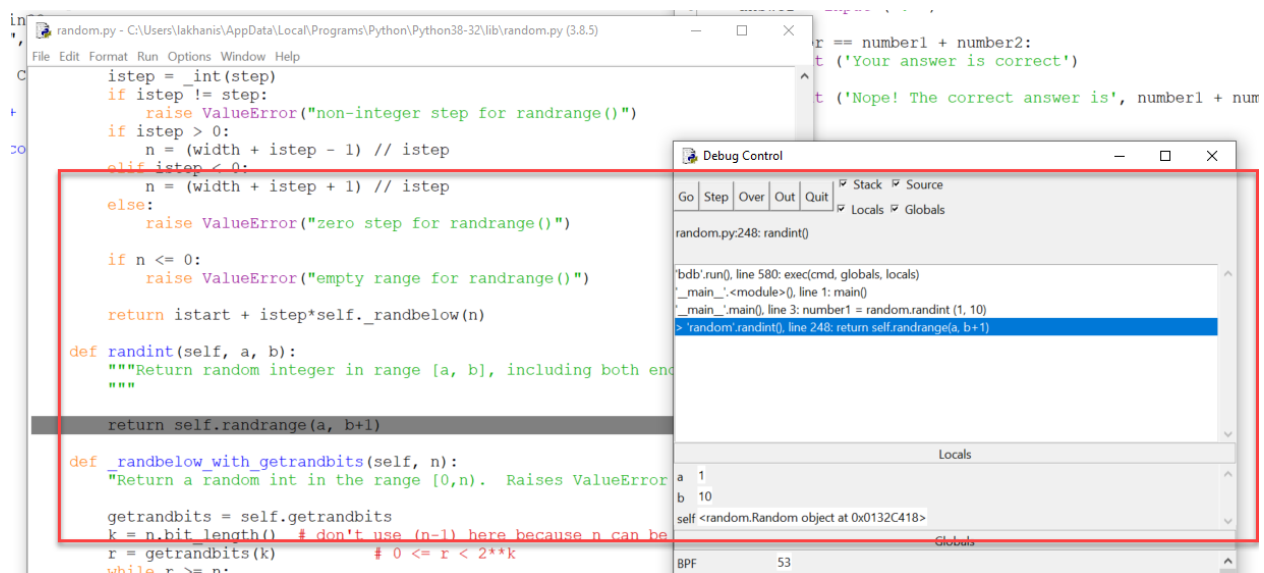
- You can insert a break point by right clicking on a line of code and selecting “Set Breakpoint” from short cut menu. You can clear the break point by right clicking on a line of code and selecting “Clear Breakpoint” from the short cut menu. IDLE will highlight the line with breakpoint. If you think that there is an error in certain part of the program (like loop) then you can set a breakpoint at start of that loop, click on Go button to execute the program. Your debugger will stop at the line of code with breakpoint.
- You can use the Quit button to quit the debugger.
- Globals area in the Debug control window will display all the global variables. We will not use it for this lab.
- Locals area in the Debug control window will display all the local variables and their values (as the variables are created during the execution of your program).
- Type `main()` in the shell to start the execution of your program. Your debugger Control will show that it is stopped at the first line (line 1) of your code (`def main()`)



- Now click on the Step button to complete the execution of the first line and move to the second line of your code. The Debug Control window will display line 2 of your code.

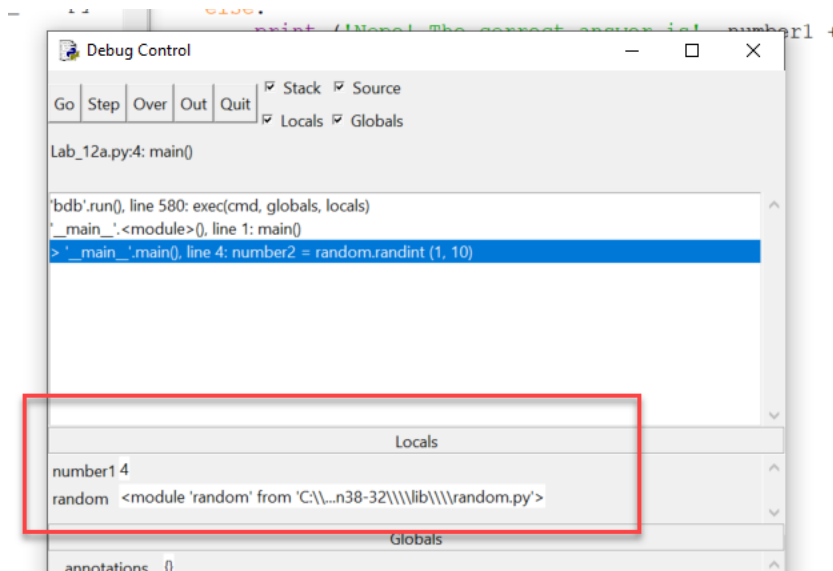


- Line 2 of your code will import the random module. Click on Step Button to complete the execution of line 2 and move to the next statement (line 3) in your Python code. Now you can either click on Step button which will take you into randint function or you can click on the Over button to complete the execution of randint function and line of your code. Go ahead and click on the Step button. This will open a new code window displaying the random.py file with line highlighted in the randint function. The Locals area will display the value of variables a and b (which are local variables for randint function.)

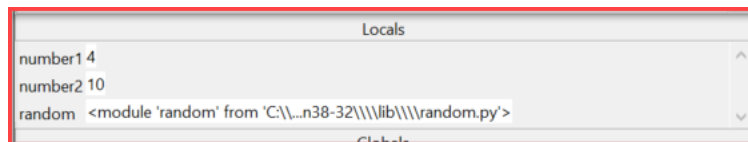


- Now click on the Out button to get out of the randint function and go back to your Python code. This will finish execution of the line 3 of your Python code. You can close the random.py window if you want. Now the Locals area will display the value of

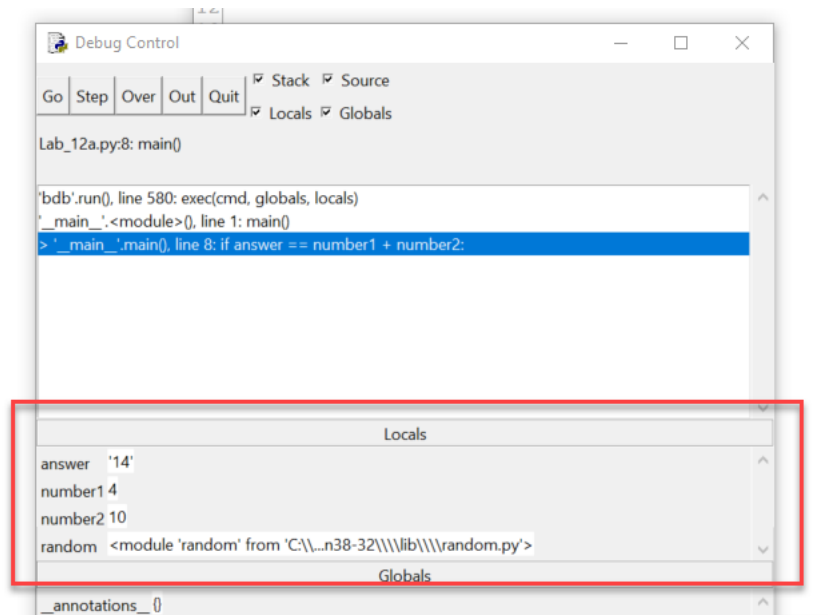
`number1` (variable in your Python code). Note that the Locals area also show that you have imported `Random` module into your Python code.



- Click on **Over** button to execute the next line of code (line 4) in your Python code. This will create another local variable `number2` and its value will be displayed in the Locals area.



- Click on the **Over** button to execute the print statement which will display the message in the shell.
- Click on **Over** again. It will execute the input statement and wait for you to type your input in the shell. Go ahead and type the correct answer. Your Python code will read your input complete the execution of line 6 of your code which is input statement. Now the Locals area will display three variables `answer`, `number1`, and `number2` (variables are displayed in alphabetical order).



- Click on the Step button. Debugger will execute the `if` statement and will jump to the `else` part (even if you inputted the correct answer). Look at the values of `answer`, `number1`, and `number2` in your Debug Control window. If you think that you know what is wrong with the code then click on the Quit button to quit the debugger, close the Debug Control window, fix the code and run it to make sure it does what it is supposed to do.

Briefly Describe the bug in the program and how you fixed it:

Problem and Its Solution – Part 1

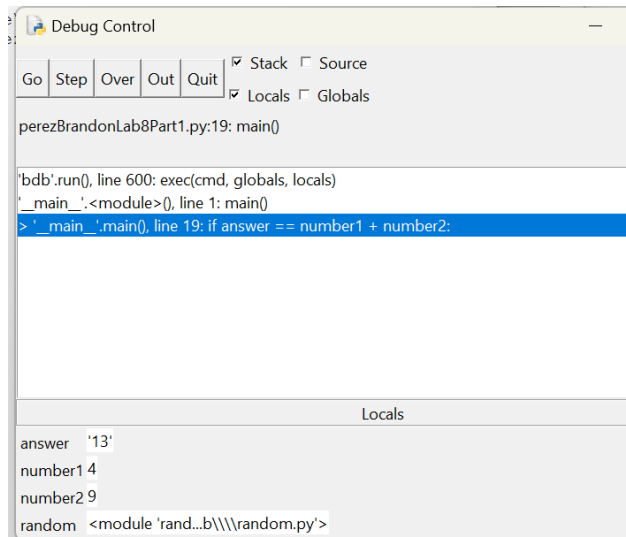
When inputting the answer, the input is never converted to an int.

This causes the program to always say "Nope! The correct answer is ___"

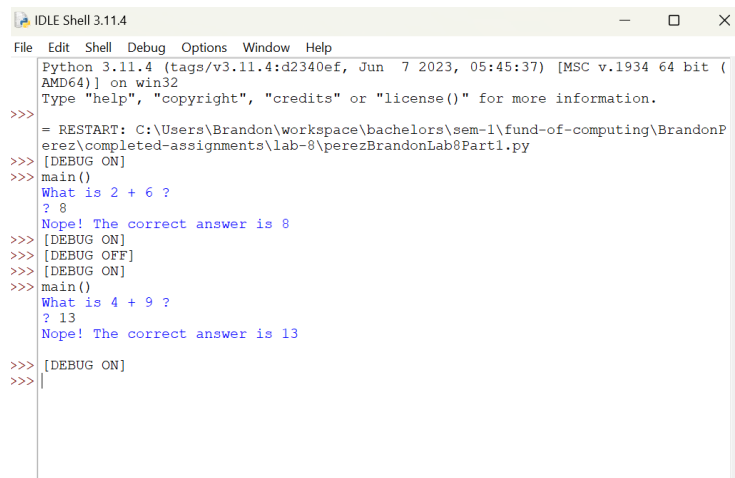
I just wrapped the input statement in `int()`.

Once you are satisfied with your results then run your program and take a screen shot of your output.

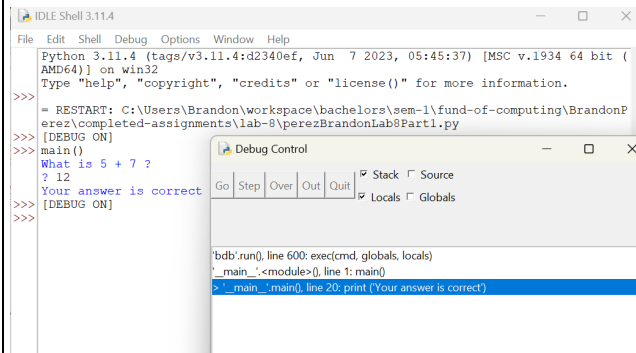
Screen Shot 1 – While Debugging



Screen Shot 2 – While Debugging



Screen Shot 3 – After Debugging



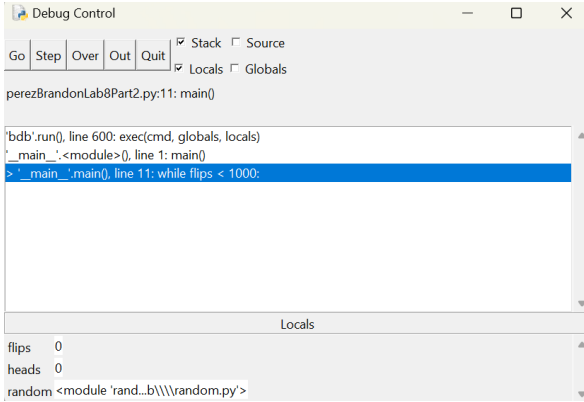
Part II – Debug a Python Program

- **Note: Take two screen shots at two different points as you run the debugger to debug the given Python code.**
- Open the file Lab_12b.py and save it as lastnameLab08Part2.py
- Run the program and you will see that it is stuck in an infinite loop. Press CTRL-C in shell to kill the execution of your code. Use the debugger to find the bug, fix the code and run it to make sure it does what it is supposed to do.

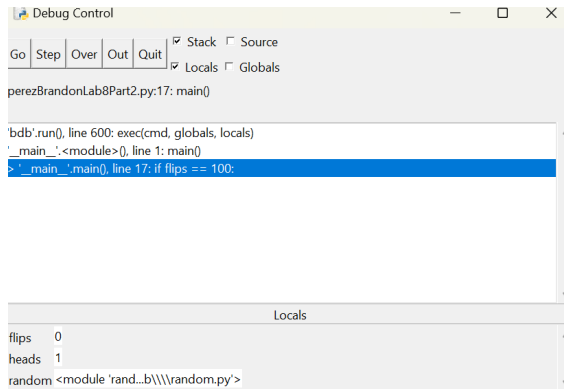
Briefly Describe the bug in the program and how you fixed it:

Problem and Its Solution – Part 2
flips never increments up. The program repeats until flip equals 1000, but flip stays at 0 forever. I just added the line: flip = flip+1 as the first line of the while loop. each iteration is a "flip"

Once you are satisfied with your results then run your program and take a screen shot of your output.

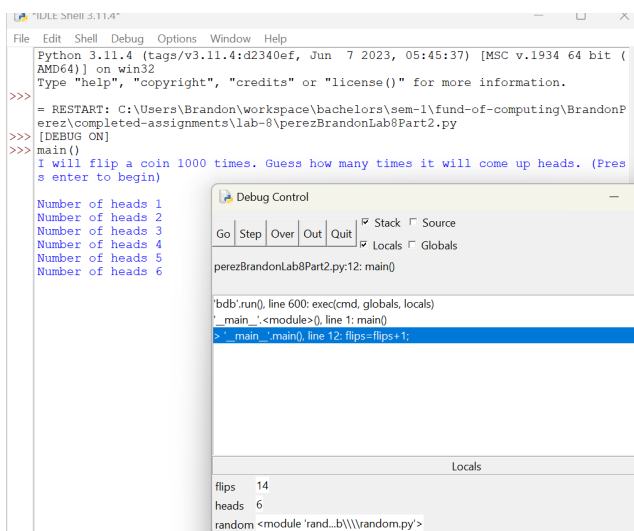
Screen Shot 4 – While Debugging	
	so good so far...

Screen Shot 5 – While Debugging



how can heads be 1 when flips = 0

Screen Shot 6 – After Debugging



Every program should have the following comment block at the top. Make sure to fill in your name, class with section number, due date, brief description of your program, and status of your program:

```
#  
# Name:  
# Class: CSCI 1411-00X  
# Due Date:  
# Description:  
# Status:
```

Rubric for Lab 8:

Criteria	Rating
Part I (Problem and Its Solution – Part 1)	Correctly identified the problem and fixed it – 20 points Correctly identified the problem but can't find the solution – 5 points Cannot correctly identify the problem – 0 points
Part I (Screen shot 1)	Screen shot included – 5 points No screen shot included – 0 points
Part I (Screen shot 2)	Screen shot included – 5 points No screen shot included – 0 points
Part I (Screen shot 3)	Screen shot included – 5 points No screen shot included – 0 points
Part 2 (Problem and Its Solution – Part 2)	Correctly identified the problem and fixed it – 20 points Correctly identified the problem but can't find the solution – 5 points Cannot correctly identify the problem – 0 points
Part 2 (Screen shot 4)	Screen shot included – 5 points No screen shot included – 0 points
Part 2 (Screen shot 5)	Screen shot included – 5 points No screen shot included – 0 points
Part 2 (Screen shot 6)	Screen shot included – 5 points No screen shot included – 0 points
Total Points	70