

CS2505 – Python Lab 02

02.03.2017

Let us start with the **Important** stuff first:

1. The grading for these Labs will take into account the **correctness** of your solution, the **approach** taken, and **comments**, which should be clear and concise. We will be checking carefully for plagiarism and penalties will be strictly applied.
2. If you don't understand a question, please ask Alexander or Ahmed, the Lab Demonstrators, who will be only happy to help.
3. All Labs, will consist of some Python programming and some questions. To maximise your Continuous Assessments marks, please answers all sections.
4. The Continuous Assessment labs are worth 20% of your final year mark for this module. Thus each of the five labs is graded out of maximum of 4 marks. To offer students an additional means of maximising their Continuous Assessments marks, each week we will also offer an additional *Coding Assignment*, from which you can receive a maximum of 1 additional mark. This extra assignment is optional and will not affect your ability to receive the maximum 4 marks for the initial part of this lab.

*** Note: irrespective of how many questions you attempt over the five weeks, the maximum number of marks that can be received from the Continuous Assessment aspect of this module is 20 marks ***

Your solutions for this Lab, including the solutions for the additional assignment should you decide to attempt same, must be submitted on Moodle within the specified deadline. Please note that no late submission will be accepted by Moodle. If your solution files cannot run successfully, you will lose marks. So make sure that there is no syntax, compilation or run-time error. You do not need to include your name or UCC ID in the name of the submitted files (Moodle does that automatically). Follow the file naming conventions as mentioned in the description of assignment.😊

We recommend (but not obligate) that you follow the official style guide for Python:
<https://www.python.org/dev/peps/pep-0008/>

Lab 2:

In this Lab you will build on the basics of Python Socket Programming, learned in last weeks Lab.

In this week's lab, we are going to:

1. Run last week's lab with the Server running on another student's machine.
 2. Create a Simple Chat Program, with Command line input on both the Client and Server.
-

1. Remote Server:

Get another student to give you the domain name of his/her lab machine.
Get them to run their implementation of the Server created in last week's lab and then supply the domain name of their machine as input to your Client (Make sure the Client and Server have the same port number).

Questions:

- A. What happens now when you run your client?
 - B. Has the log file to which your server writes changed? Why?
 - C. Has the log file changed on the other student's machine? Why?
-

2. Simple Chat Program:

Create a new project folder called "CS2505_lab2", taking your coding solution for question 2 from last week's lab, you will update the code on the Server side to replicate Command Line input.

A few things you will need to remember:

- a. Make sure to print to the command line what the Client and Server are sending and what the Client and Server are receiving. Just like a normal chat program, you need to see what you said and what the other person said.
- b. In last week's lab, the Client closes when it has received the sentence from the Server, this week, it will need to keep on running, so that you can continue to send and receive sentences to/from the Server.
- c. Due to the way the Sockets work, you will need to wait for the other person to respond, before you can send a message. Anyone want to see if they can fix this?
- d. Once it works on your machine, try it via the Server on another student's machine.

Questions:

- D. Is it possible for a third person to join the Chat? Try it.
- E. Why do we need the Client to keep running?

Submit the solution files as client_solution.py and server_solution.py. Submit the answers to questions as answers.txt file.

Additional Coding Assignment:

The additional assignment this week is to update the chat program to send files (images) from the client to the server and vice versa. To reduce the complexity of the additional assignment, rather than show the images in the terminal, just save the images to a file and output the file location to the local user. Also save the image file

to be sent in your project folder and reference locally, i.e. “file=open”file1.png”,”. Finally you can also use a special text command in the terminal to add the file. That said if you wish, please design a GUI in which you can add the images dynamically, or via a button, as well as view them on the receiving side within the GUI, as is available in most modern chat programs. Submit the solutions as client_solution_additional.py and server_solution_additional.py.