



Date handed out: Friday 6 April 2018
Date submission due: Friday 20 April 2018 23:55

This assignment aims to help you practice the following topics in Python: operating system interface and system programming, multithreading and concurrent programming, network programming, graphical user interface components and database application development in Python. Your main task in this assignment is to develop a client-server communication system based Transmission Control Protocol (TCP). The server should be able to communicate with multiple clients at the same time as illustrated in Figure 1. **You can work individually and in a group of two. If you have worked in a group in the first assignment, then you will need to work with the same person in this assignment.**

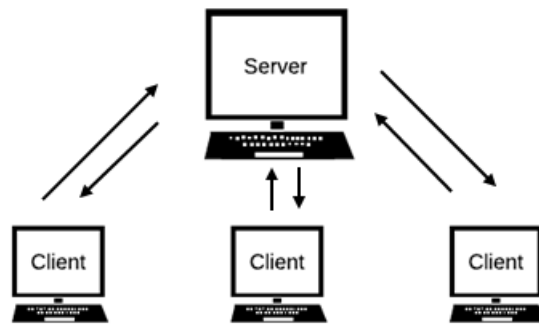


Figure 1: Client-Server Model

Practise Python Coding!

In this assignment, you need to develop a system which allows users to practice their Python programming knowledge. In this system, the server will ask some questions to its clients and automatically evaluate their submissions for the questions based on a number of test cases. You need to implement both the server and client sides.

1. Server Side

The server can be started in two different modes: Administrative Mode & Running Mode. You do not need to develop a graphical user interface for the server side.

- a. Administration Mode:** This mode will allow to create a SQLite database¹ for questions by using the following structure shown in Figure 2 (questionbank.db), and manage the database.

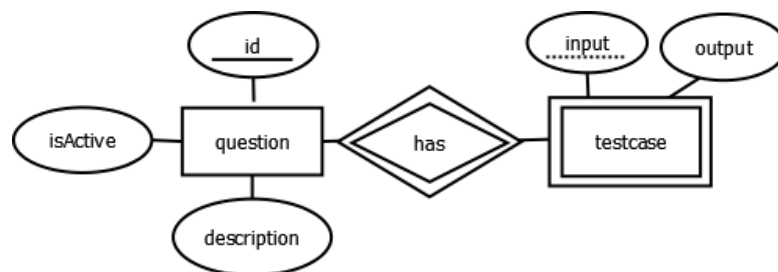


Figure 2: Entity Relationship Diagram (ERD)

¹ <https://docs.python.org/3/library/sqlite3.html>

Table 1 and Table 2 shows some example records for the question and testcase tables respectively.

id	description	isActive
1	Write a function called sum that takes two numbers and returns their sum?	1
2	Write a function called sum that does not have any arguments and prints "Hello World"	0

Table 1: question Table

q_id	input	output
1	(2,3)	5
1	(2.5, 6.5)	9.0
2	()	Hello World

Table 2: testcases Table

If the database has not been created yet, then it should be created and then the following options should be provided. However, if the database has already been created, then the following options should be provided directly.

- Add a new question – These questions should ask to write a function for a specific purpose (see Table 1). The function name and the arguments should be clearly written.
 - Update an existing question (activate/deactivate it or change its description) by using its unique id – The deactivated questions will not be asked to the clients.
 - Add a new test case for a particular question where the input format should be as follows: (argument₁, argument₂, argument₃,...argument_n)
 - Update an existing test case by using its unique id.
 - Delete an existing test case by using its unique id.
- b. Running mode:** This mode will run the server and the server will wait for connections from its clients. When a particular client is connected to the server, the work flow should be as follows:
- The server will randomly select a question from its question bank and sends it to its client.
 - The client will need to write a function for the question and submit it. The client needs to use the given function name and is not allowed to add extra arguments to the function.
 - When the client completes the function, s/he will submit it to the server.
 - The server will evaluate the submission based on its test cases. The score will be computed based on the number of the test cases that the submission provides the desired outputs. For example, if there are 10 test cases and the submission provides the desired output for 4 test cases, then the score will be computed as 40%. The server will send the score to the client.
 - If the client wants to continue with other questions, then s/he will request another question. Otherwise, the connection between the client and the server will be closed.

2. Client Side

The client can connect the server when the server is in its running mode. Multiple clients should be able to communicate with the server at the same time period. There will be two visual screens for the clients. The first screen is the question screen whereas another one is the score screen. For these screens, you need to work with graphical user interface components by using the **tkinter** module².

² <https://docs.python.org/3/library/tk.html>

- a. **Question screen:** There will be four items: one label, one text field, two radio buttons, and three buttons.
- The label will be used to show the question;
 - The text field (multi-lines) will allow to write the answer;
 - The radio buttons will be used to determine whether only the overall score will be shown or the overall score will be shown with the details;
 - The first button (Submit) will be used to submit the answer to the server;
 - The second button (Skip) will be used to skip the question and request a new one;
 - The third button (Exit) will be used to disconnect from the server and close the client application.

Therefore, the question screen should be as follows (see Figure 3):

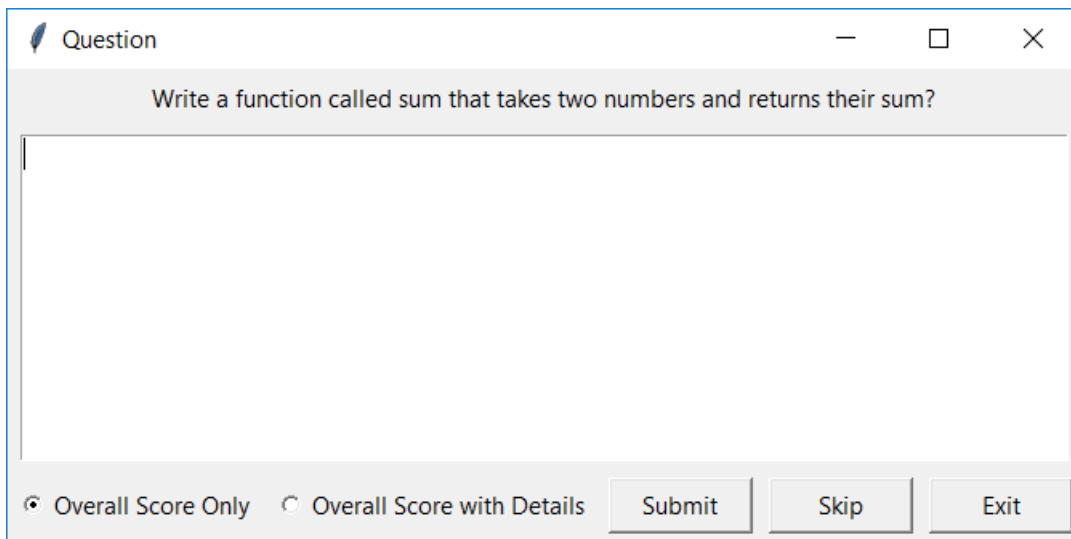


Figure 3: Question Screen

- b. **Score screen:** This screen will be a message box with yes or no options. It will show the overall score only or the overall score with the details. It will then ask the client whether s/he wants to continue. If "Yes" is clicked, then a new question will be requested. However, if "No" is clicked, then the client will be disconnected from the server and the client application will be closed. Therefore, the score screen should be as follows (see Figure 4 and Figure 5).

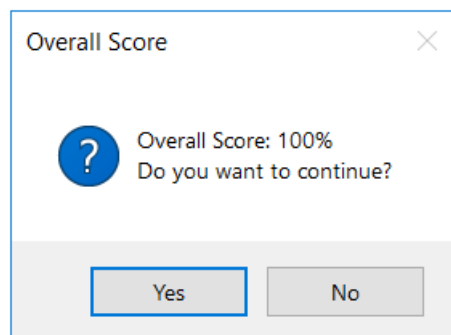


Figure 4: Score Screen – Only Overall Score

When the overall grade is shown with the details, each test case with the desired output will be shown. If the submission provides the desired output for the test case, then "TRUE" will be shown for the test case. Otherwise, "FALSE" will be shown for the test case.

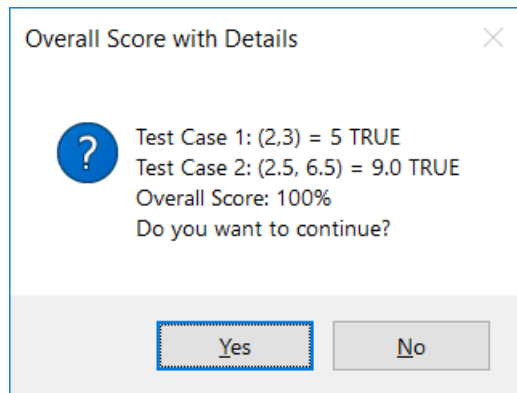


Figure 5: Score Screen – Overall Score with Details

Same Rules!

- You need to write your program by using Python 3.x.
- You can use all built-in functions and modules.
- You need to put all your files into a folder which is named with your student id(s) and submit the compressed version of the folder.
- **The code quality, modularity, efficiency and appropriate comments will be part of the grading.**