THE CHINESE UNIVERSITY OF HONG KONG
DEPARTMENT OF INFORMATION ENGINEERING

IERG3310 Computer Networks
(First Semester, 2020-2021)

Socket Programming Project

Due on 11:59pm, Nov 7 (Saturday)

Every Student MUST include the following statement, together with his/her signature in the submitted report.

I understand the University guidelines on academic honesty as shown in the web page http://www.cuhk.edu.hk/policy/academichonesty/ and I agree to comply with the guidelines. I declare that I do not plagiarize or cheat in any way that violates the spirit of the guidelines. I understand the severity of disciplinary action should the guidelines be violated.

Signed (Student _____) Date _____

Name _____ SID _____

Project Requirements

In this project, students learn hands-on experience on both TCP and UDP socket programming. Students are required to write a program (STUDENT) to interact with the provided program (ROBOT). You are allowed to use the ROBOT source codes given (written in C/C++, java and Python) in the project package to complete this project. **All** students should finish **steps 1 to 8**. Steps 1 to 6 give you a taste of basic programming. Steps 7 to 8 ask you to do some simple experiments using the programs. **You must do this project by yourself. However, you are allowed to discuss with your friends or fellow students, but DO NOT copy others' code. We perform code-similarity-checking on your submitted code.**

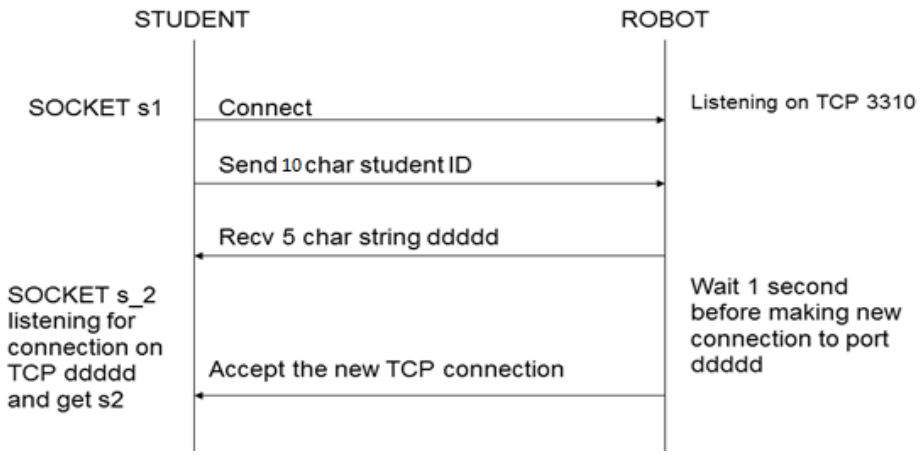
Basic Programming Part:

Your program, STUDENT, has to interact with ROBOT according to the following steps:

1. Start the ROBOT by running "robot.exe" in a command prompt.

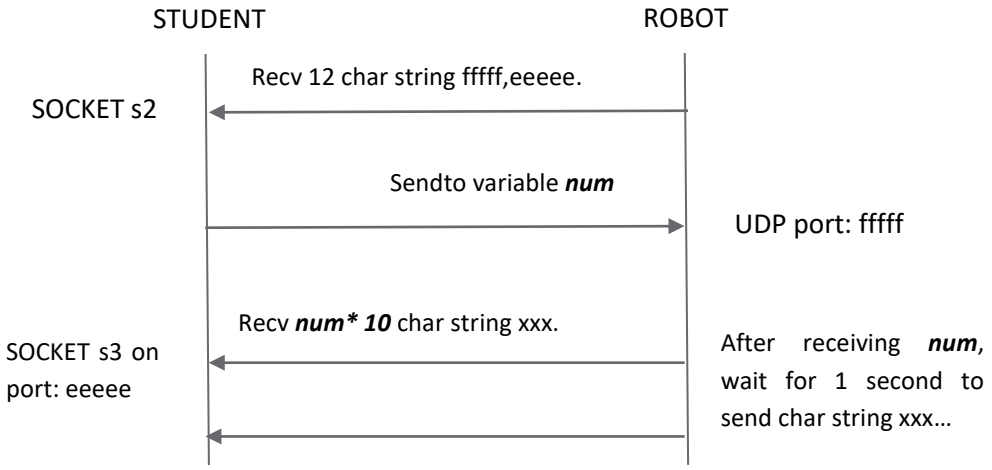
2. When the ROBOT is started, a message "ROBOT IS STARTED" will be printed, indicating that the ROBOT is now listening on TCP Port 3310. STUDENT has to connect to the ROBOT TCP Port 3310 and send your 10 char student ID string via the connection established.

3. The ROBOT will then send a 5 char string *ddddd* to STUDENT. STUDENT will need to create a TCP socket s_2 at port *ddddd* to accept a new connection. The ROBOT will initiate a new connection to port *ddddd* of UDPSTUDENT one second after sending *ddddd* to STUDENT.



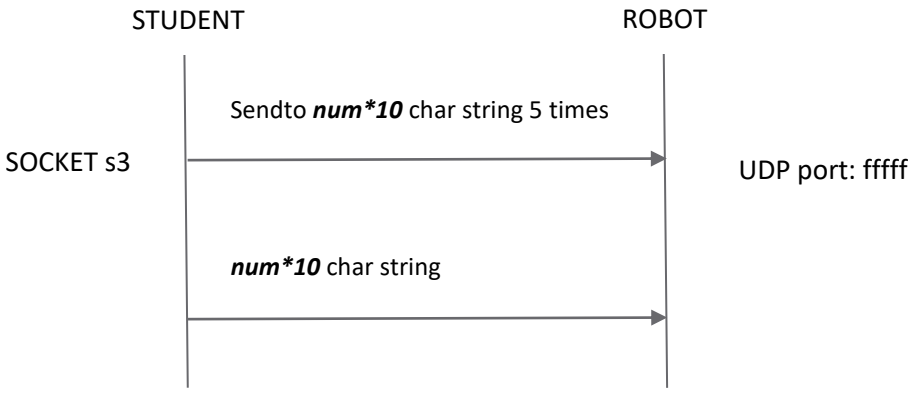
Upon accepting the connection, a new socket s2 will be returned.

4. ROBOT will then send a 12 char string "*fffff,eeeee.*" to the STUDENT using the new connection. STUDENT needs to decode the message and create a UDP socket s3 to send a randomly generated integer variable ***num*** (5 < ***num*** < 10) to ROBOT on port *fffff*. Then ROBOT will send a char string *xxx…* of length ***num * 10*** to STUDENT one second after receiving ***num*** and STUDENT will receive the string using s3 on UDP port *eeeee*.



ROBOT will send the string xxx… 5 times, once every 1 second. STUDENT only needs to receive any one of the strings.

5. When STUDENT receives the char string *xxx*, it will send back the string to the ROBOT at UDP port *fffff*. The string will be sent 5 times, once every 1 second. The ROBOT will check if the two strings are the same.



6. In this step, you need to run ROBOT and STUDENT on 2 different machines and make sure all the steps above can be successfully executed. You may need to use Wireshark on the machines for network-level debugging, e.g., if there is a firewall on either of the machine and/or along their end-to-end path, the programs may fail. It is then your

responsibility to fix it, e.g., by choosing a different pair of machines elsewhere and/or configure them properly.

(For steps 1 to 6, the original provided ROBOT will be used to test your STUDENT program.)

Experiment Part:

7. In this step, you will **study the effect of the receiver buffer size** on the TCP socket s2 on port *ddddd* by modifying the ROBOT and STUDENT codes.

   First, you will get the existing receiver buffer size of s2 on STUDENT and print it out. Then you must inform ROBOT of this receiver buffer size on TCP Port 3310 by modifying both the STUDENT as well as ROBOT code: STUDENT will need to include some keywords in the string to signal ROBOT that the value is the buffer-size information (e.g. the string can be "bsxxx", where xxx is the size of your buffer). Modify the source code of ROBOT, so that when it receives the buffer-size information from the STUDENT, it will send a large number of messages to STUDENT's s2 port as quickly as possible within 30 seconds. STUDENT should be modified so that it can count **the size and number of messages** actually received in this period and print out the total number of packets STUDENT receives, together with total received bytes. In addition, you could either printout the statistics every ten seconds or after receiving all messages. An example printout can be:

   [STUDENT] Number of received messages: xxx, total received bytes: xxx.

   ***Remark*:**

   1) The receiver buffer size here is the **internal buffer size (receive socket buffer size).** The receive socket is allocated for each individual TCP connection. Don't confuse it with the parameter in recv() which specifies the buffer in the user/application space. The receive buffer here is the buffer in the kernel and not in the application space.

   2) The messages above refer to messages at the application layer. The message size can be any value. For example, you can set to have the same size as the receive socket buffer. But this is not absolutely necessary. The assumed message size by the sender and the receiver should be the same to be consistent. You can make your own assumption on the pre-agreed message size between ROBOT and STUDENT.

   3) Note that TCP socket is stream-based. Assuming you use a blocking socket (the default mode), each send() may not completely deliver the intended amount of data from the application-layer buffer to the send socket buffer in each send(). You need to check. An example copied from robot.cc is shown below. Note that byte_left below refers to the message size at the application layer. The while() loop ensures a whole message is delivered to the send socket buffer. If your application wants to send multiple messages, there should be another outer while() loops. When the while() loop below exits without error, that means one

application-layer message has been delivered to the send() socket buffer successfully.

```
int byte_left = (int)strlen(messageBuffer);
// Similar to recv, a loop is required
while (byte_left > 0){
        iResult = send( s1, messageBuffer-
        byte_left+(int)strlen(messageBuffer), byte_left, 0 );

        if (iResult < 0) {
                printf("send() failed with error\n");
                return 1;
        }

        byte_left -= iResult;
}
```

8. Design the code for STUDENT and ROBOT for Step 7 so that they will repeat the entire procedure in Step 7 after STUDENT changes its receiver buffer size over a wide range of values, e.g. [1, 5, 10, 25, 50, 200, 500 or even 1000] % of 1000 bytes. Compare the corresponding TCP throughput from ROBOT to STUDENT as the latter's receiver buffer size changes. The results should be written in the report. The result should contain the **graph** showing the relation between the **receiver buffer size** and the **system throughput**, as well as your interpretation.
   An additional question to be answered in the Lab report: What is the limitation of the experiment set-up since we do it in IE Common Lab? (Hint: What if the network is a low-speed one?)

   Remark: **System throughput = total received bytes / duration**

Programming Language and Platform

Students can use C/C++, Java or Python to complete the project. Students can choose to implement the project on any platform that is available in the IE Computing Lab. The platforms available in the IE Computing Lab are Microsoft Windows and Linux. The submitted source codes have to be compliable and runnable in the IE Computing Lab.

Grading and Demonstration

Steps 1 to 6 in this project contribute **10% points to the course**.

Steps 7 and 8 in this project contributes **5% points to the course**.

The table below contains a comparison:

	achieves
Finished steps 1 to 6	10% of the course
Finished steps 7 to 8	5% of the course
Total	15% of the course

A demo session will be held after the deadline, each student should compile, run and operate his/her program in front of a tutor. The tutor may ask some questions relating to the program, whether or not correctly answering the questions will affect your grading in the project.

Submission and Academic Honesty Policy

Students are required to submit a zip/rar file containing the **executable file**, **source codes** and **report** describing the design and use of your program to Blackboard https://elearn.cuhk.edu.hk/ before the deadline. Late submission within 24 hours suffers penalty of 30% deduction. Further late submission will be given zero point.

Students have to submit the report together with the declaration form to Blackboard for this project. Project without declaration will receive zero mark. You will need to put down the references you used in both the program source code and the report.

Software will be used to check if any submitted program is copying from the others. When scholastic dishonesty is suspected, the matter will be turned over to the University authority for inquiry, with a recommendation of "Fail" for the course.

Project submission checklist:

- The final source codes for STUDENT and ROBOT and a runnable file for each of them.

  (Online)

- A no-more-than-3-page report describing the design of your programs (and discuss the throughput measurement results for Step 7 and 8).

  (**Soft copy**)

  To better describe your code, you need to show the related lines of code, for example:
  "… then I adjust the receiver buffer size by
  s2_server.setsockopt(SOL_SOCKET,SO_RCVBUF,RECEIVER_BUF_SIZE)
  "

  For step 7, provide your screen capture with the printout like
  "[STUDENT] Number of received messages: xxx, total received bytes: xxx."
  in the report.

  For step 8, provide the graph mentioned above in the report.

- Declaration form (Soft copy).