Dalhousie University
Faculty of Computer Science

CSCI 3171 **Network Computing** **Summer 2017**

Assignment 4 **[20 marks]**

Submission Deadline: **FRIDAY, JULY 14, at 11:59PM**

ASSIGNMENT TASK

Design a client-server application using TCP sockets, where you use the client to play a game of "Prisoner's Dilemma" with the server.

You may complete this assignment in **groups of 2 students**. Only one submission per group is required.

ASSIGNMENT DETAILS:

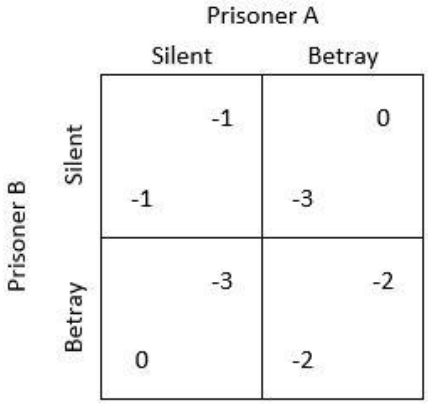
The Prisoner's Dilemma:

The prisoner's dilemma is a standard example of a game analyzed in game theory that shows why two completely "rational" individuals might not cooperate, even if it appears that it is in their best interests to do so.[1]

It is presented as follows:

> "Two members of a criminal gang are arrested and imprisoned. Each prisoner is in solitary confinement with no means of communicating with the other. The prosecutors lack sufficient evidence to convict the pair on the principal charge. They hope to get both sentenced to a year in prison on a lesser charge. Simultaneously, the prosecutors offer each prisoner a bargain. Each prisoner is given the opportunity either to: betray the other by testifying that the other committed the crime, or to cooperate with the other by remaining silent. The offer is:
>
> - If A and B each betray the other, each of them serves 2 years in prison
> - If A betrays B but B remains silent, A will be set free and B will serve 3 years in prison (and vice versa)
> - If A and B both remain silent, both will only serve 1 year in prison (on the lesser charge)"[1]

This is summarized in the figure on the right. The following YouTube video also explains Prisoner's Dilemma (you only need to watch up to 1:50 minutes of this video): https://www.youtube.com/watch?v=t9Lo2fgxWHw



Your Task:

you will write simple client-server application(s) in which the client acts as prisoner A and the server acts as prisoner B. The protocol between the client and server should be as follows:

- The server program is started on a user-defined port.
- The client program is started and connects to the server using the server IP and port number provided on the command line.

[1] https://en.wikipedia.org/wiki/Prisoner%27s_dilemma].

- The client asks the user for input. The input may either be S (for Silent) or B (for Betray). Any other input should result in an error message, asking the user to try again.
- The user's input is sent to the server via the connected socket.
- The server (acting as prisoner B) may decide to remain Silent (S) or Betray (B). How the server decides is completely up to you. For a fair game, you may use a random approach (use **`rand()`** % 2 with 0 indicating S and 1 indicating B for example).
- The server reads the user's input from the client socket, evaluates the outcome (years in prison for both Prisoner A i.e. client, and Prisoner B i.e. server), and sends the result (Prisoner B's decision and the Prison Sentence) back to the client.
- The client should display the server's reply to the user.
- The client should give the user an option to try again or quit.

Socket Programming:
The steps for creating a socket on the client side are:

- Create a socket with the **`socket()`** system call
- Connect the socket to the address of the server using the **`connect()`** system call
- Send and receive data. There are several ways to do this. You may use the **`read()`** and **`write()`** system calls, or the **`send()`** and **`recv()`** system calls

The steps involved in establishing a socket on the server side are as follows:

- Create a socket with the **`socket()`** system call
- Bind the socket to an address using the **`bind()`** system call. For a server socket on the Internet, an address consists of a port number on the host machine
- Listen for connections with the **`listen()`** system call
- Accept a connection with the **`accept()`** system call. This call typically blocks until a client connects with the server
- Send and receive data

Additional Help and Reading:
Following are some helpful tutorials on socket programming using TCP including code samples. Feel free to study and understand any or all of these before starting on your assignment.

- http://www.linuxhowtos.org/C_C++/socket.htm
- https://www.codeproject.com/Articles/586000/Networking-and-Socket-programming-tutorial-in-C
- http://www.csd.uoc.gr/~hy556/material/tutorials/cs556-3rd-tutorial.pdf

SUBMISSION DETAILS:
Upload a single zip file containing:

1. your code (consisting of your c or c++ file(s)),
2. an image file (.jpg or .png) showing a screen capture of your program in action, and
3. a readme.txt file containing names of the group members and instructions to compile your program

to the Assignment 4 dropbox on Brightspace. Do not submit the executable.

SUBMISSION DEADLINE:

Friday, 14-July-2017, 11:59PM

GRADING CRITERIA:

- Code for both the client and the server compiles using gcc. Assignment instructions are properly followed.
- Client can connect to the server running on either a local or a remote host. Be sure to test your client's connectivity by connecting to the server running on a different machine.
- Client's message is received by the server and server correctly finds the outcome (i.e. the Prison sentence).
- Server's message is received by the client and the client can display this information to the user.
- Code is well written, properly formatted and commented.

[Note: You do not need to write a multi-threaded server. Connecting a single client is enough]