Lab 5
Sockets: Master/Slave Voting

Objective
In this lab, students will learn how to perform IP communication and synchronization in a distributed environment. This objective will be achieved by:
- Creating and using sockets for communication
- Creating a Master/Slave setup

Prelab
In the prelab you should briefly explain the functions needed to:
- create a socket
- write to a socket
- read from a socket
- bind a socket
- change socket permissions to allow broadcast
Do not forget to include the pseudo code or flow chart of the program.

Lab Procedure
In a master/slave setup, one device has control over one or more other devices. An example of this is a network of computers, where one computer (master) receives tasks and assigns them to other computers (slaves) based on certain criteria. Usually, if no master device is present, the devices use an election to determine a new master.

For this lab each student will implement a server using their TS-7250 board. Each board will start with a slave status. A client program can ask all the student’s boards which one of them is the master by sending the message “WHOIS”. If no board replies that it is the master, the client program can ask the boards to vote on a new master by sending the message “VOTE”. To vote, each board will send a broadcast message to all of the other boards that begins with a # sign and contains the board’s IP address followed by a space followed by a random number that it generated (e.g.: “# 10.3.52.2 4”). After receiving the votes, each server program must decide if it will become a master or not. This is done by comparing its own vote with the other votes received; the highest vote wins. If a tie occurs, the board with the highest IP wins. If the client program sends another WHOIS message, the board who is the new master will send a message to the client program saying that it is the master. This message should include the student’s name and the board’s IP (e.g. “Peter on board 10.3.52.2 is the master”).

The server should also disregard any invalid message.
**Requirements:**

1.) Each board must determine its IP dynamically (you cannot hard code the IP address in).
2.) The messages will always be character arrays of size 40.
3.) Votes must be randomly generated integers in the range $[1, 10]$.
4.) The port used for communication should be an argument of your program, i.e., when you run your program, you should be able to provide the port number. By default, port 2000 will be used.

**Hints:**

A test program is provided in the `/opt/ts7250/students` directory on the server (nfs1). The name of the binary is `Lab5_client` and it is a client program that broadcasts whatever the user types (including the ‘return’ character when you press Enter) and also prints out messages that are sent to it. This program will be used to test your server program on the post-lab due date. You should use it to test how your board responds to the different messages and the tie breaker.

When you go to run this binary, make sure you log into another board, so you have one board running `Lab5_client` and another one running your server program. You can actually run many instances of your server program on different boards, to test the different cases.