|  |
| --- |
| **Sockets Lab Manual** |

**Question # 1**

You need to conduct election in some areas. For this purpose you have a centralized server which keep track of total number of votes of each party. For instance you have two parties BLUE party and GREEN party and there are 5 areas each area has its id 1, 2, 3, 4 and 5 respectively. Every area has some population size. Let’s population size of each area is 5, 10, 15, 20 and 25 respectively.

Every area has/is a client connected with main server. When they connect with main server they send their id to main server, main server checks whether the same id is connected or not, In case of already connected or finished with election, server will discard the connection request of client, and in case of new connection server will send population size of that particular id to client.

Now client will conduct election, a client has population size of its area from server side, it display a poll to user.

Select you desired party

1. Green
2. Blue

Press 1 for Green and press 2 for Blue.

Now client decrease the population size because one of the voter has been voted successfully. At the end of the each voter choice you should display total number of votes of his desired party. Notice that you should display all votes of that party from every area. When all people of some area/id has been voted successfully then that area/id can’t participate in election anymore.

Server will have a count total\_Blue\_votes and total\_Green\_votes. Server will display final results (who is the winner) of election when all 5 areas done with their election.

For above task you have 2 create only two files (i) server (ii) client run server on one terminal and client on five different terminals. Input area id from user in client program. Use socket programming to solve above problem.

A server and a client file is provided. Use g++ to compile server and use gcc to compile client.