Name: Preyash Date: 20-01-2022

Registration Number: 20BPS1022

LAB-02

Socket Programming

AIM: To print a message from the server to the client utilizing the socket functions, and vice versa

ALGORITHM

Server-side Algorithm

1. Start.

2. Declare variables.

3. Take the port number as input from the user.

4. Using socket(), create TCP socket for server.

5. Using bind(), bind the socket to server address.

6. Using listen(), put the server socket in a passive mode, where it waits for the client to approach the server to make a connection

7. Using accept(), at this point, connection is established between client and server, and they are ready to transfer data.

8. Sending a message to client side.

9. Printing the message received from client side.

Client-Side Algorithm

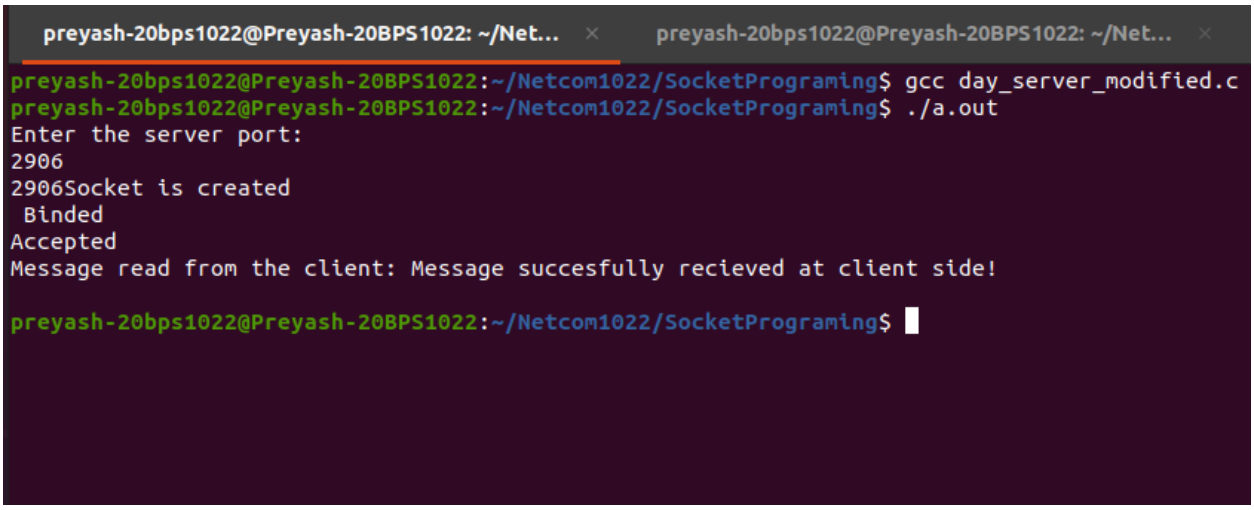
1. Start.
2. Declare variables.
3. Take port as input from user.
4. Using socket(), create TCP socket for client.
5. Using connect(),connect the client to server to exchange information.
6. Using recv() ,receive the data from server side.
7. If the connection will be successful data will be revived.
8. The message from server side is printed at client side.
9. The client will acknowledge by sending a message.

Server Program Source Code:

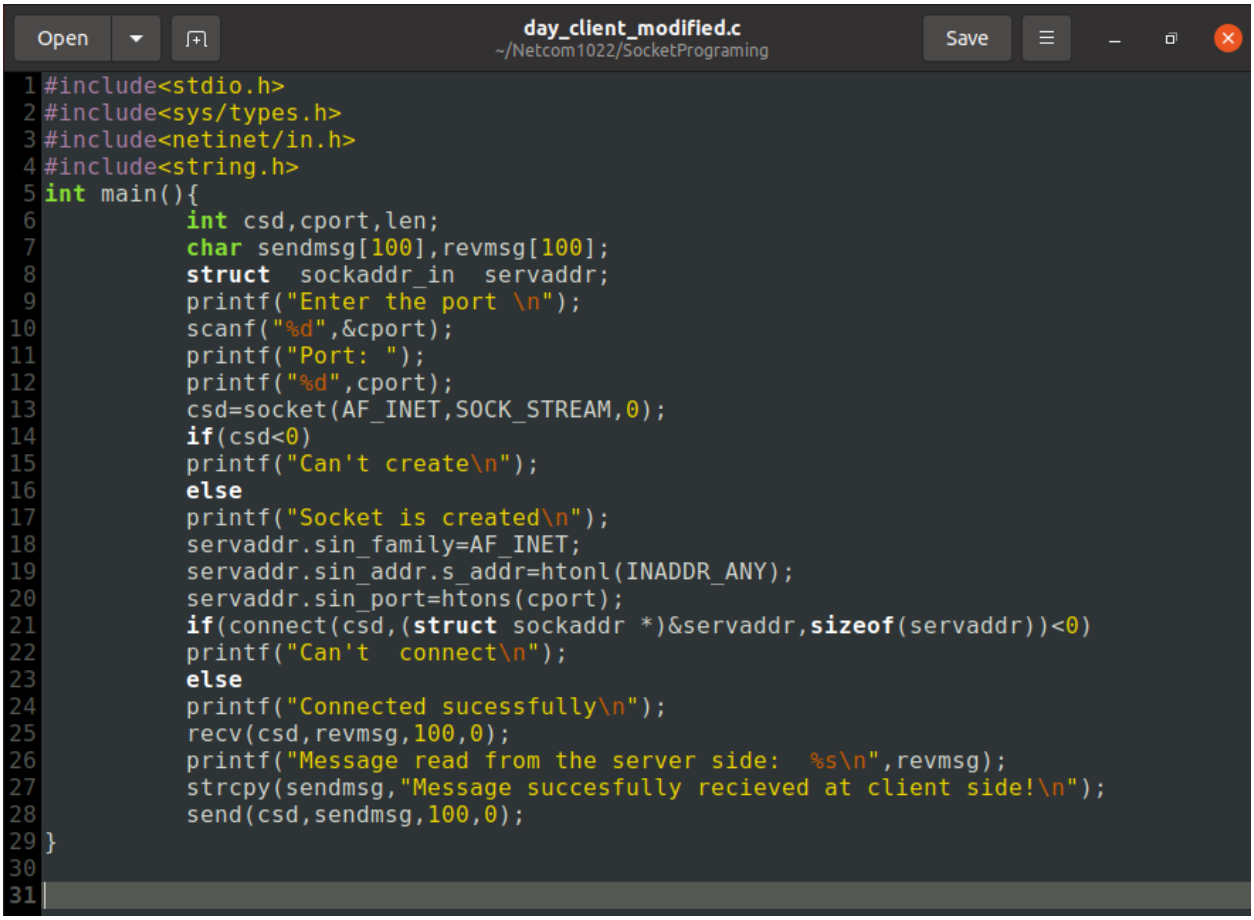
Open | day_server_modified.c ~/Netcom1022/SocketPrograming | Save

```c
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>
#include<time.h>
int main(){
        int sd,sd2,nsd,clilen,sport,len;
        int port;
        time_t  ticks;
        char sendmsg[100],rcvmsg[100];
        struct  sockaddr_in servaddr,cliaddr;
        printf("Enter the server port:\n");
        scanf("%d",&sport);
        printf("%d",sport);
        sd=socket(AF_INET,SOCK_STREAM,0);
        ticks=time(NULL);
        strcpy(sendmsg,ctime(&ticks));
        strcpy(sendmsg,"Namaste, this is Preyash! \n");
        if(sd<0)
        printf("Can't create \n");
        else
        printf("Socket is created\n");
        servaddr.sin_family=AF_INET;
        servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
        servaddr.sin_port=htons(sport);
        sd2=bind(sd,(struct  sockaddr*) &servaddr,sizeof(servaddr));
        if(sd2<0)
        printf("Can't bind\n");
        else
        printf(" Binded \n");
        listen(sd,5);
        clilen=sizeof(cliaddr);
        nsd=accept(sd,(struct  sockaddr *)&cliaddr,&clilen);
        if(nsd<0)
        printf("Can't accept\n");
        else
        printf("Accepted\n");
        send(nsd.sendmsg,100,0);
        recv(nsd,rcvmsg,100,0);
        printf("Message read from the client: %s\n,rcvmsg);
}
```

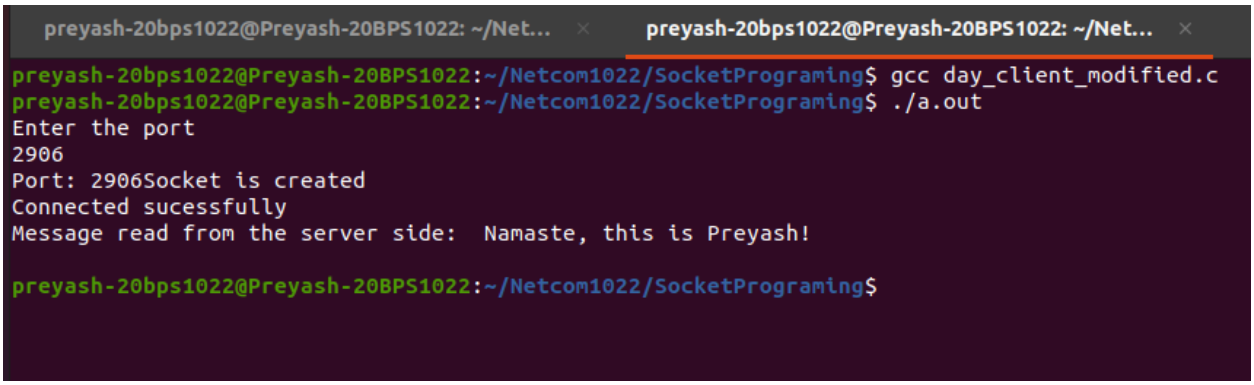
Output:



Client Program Source Code:



Output:



Result: A successful connection between the client and server is established.