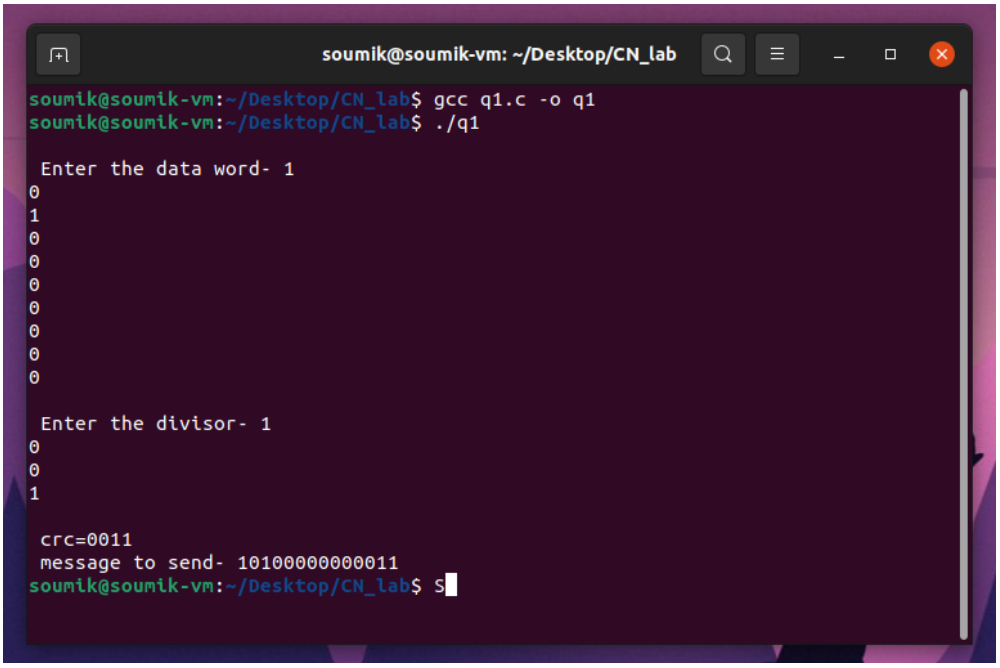
COMPUTER NETWORK LAB ASSIGNMENT

Soumik Roy
12000118029

1.Generate the CRC for a given 10 bit data word and a divisor of 4 bit.

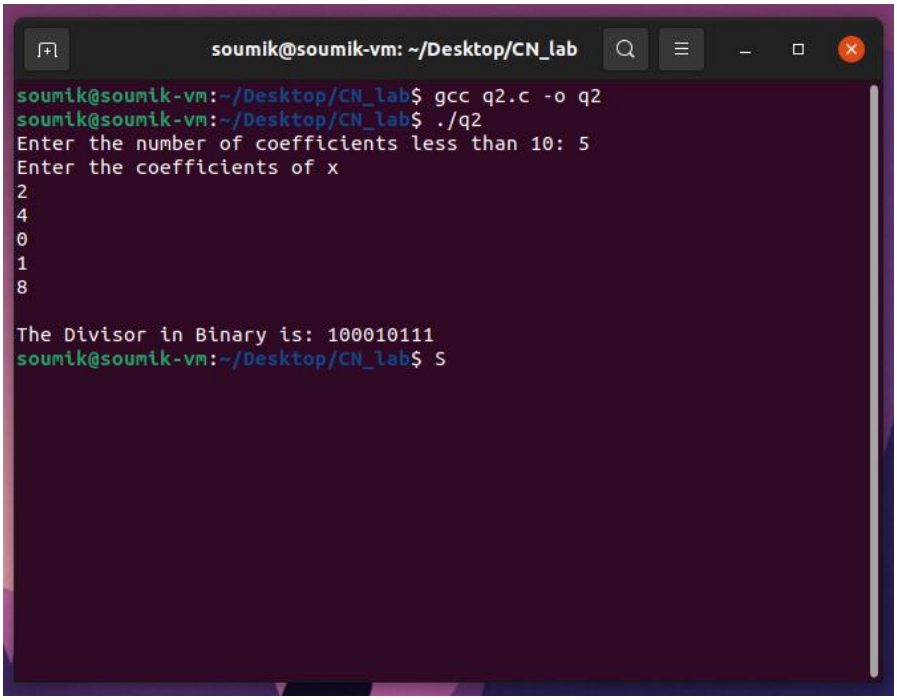
```
#include<stdio.h>
void main()
{
int i,j,k;
int dw[10],div[4],crc[3],dw1[10];
printf("\n Enter the data word- ");
for(i=0;i<10;i++)
{
scanf("%d",&dw[i]);
dw1[i]=dw[i];
}
printf("\n Enter the divisor- ");
for(i=0;i<4;i++)
{
scanf("%d",&div[i]);
}
for(i=10;i<14;i++){
   dw[i]=0;
}
for(i=0;i<10;i++)
{
 k=i;
if(dw[i]==1)
for(j=0;j<4;j++)
{
        if(dw[k]==div[j]){
                dw[k]=0;
                crc[j]=0;
        }
        else{
                dw[k]=1;
                crc[j]=1;
        }
        k++;
}
}
printf("\n crc=");
for(i=0;i<4;i++)
{
printf("%d",crc[i]);
}
printf("\n message to send- ");
for(i=0;i<10;i++)
{
printf("%d",dw1[i]);
}
for(i=0;i<4;i++)
{
printf("%d",crc[i]);
```

```
}
printf("\n");}
```



2.Generate the binary form of divisor to be used for CRC generation, while the divisor polynomial is taken as input

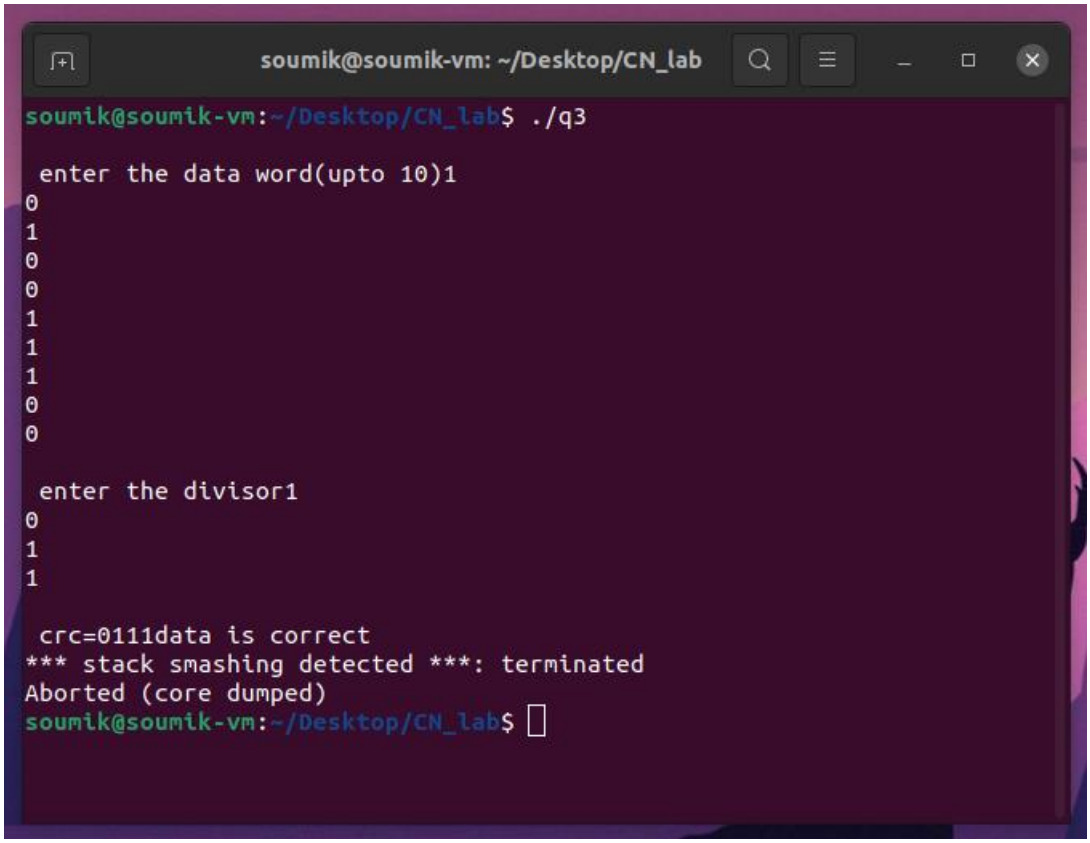
```
#include<stdio.h>
#include<string.h>
void main()
{
   char big=0;
   int i,c[10]={0}, n,pos;

   printf("Enter the number of coefficients less than 10: ");
   scanf("%d",&n);
   printf("Enter the coefficients of x");
   printf("\n");
   for(i=0; i<n; i++){
      scanf("%d",&pos);
      if(big<pos){
         big=pos;
      }
      c[pos]=1;
   }
   printf("\nThe Divisor in Binary is: ");
   for(i=big; i>=0; i--){
      printf("%d",c[i]);
   }
   printf("\n");
   }
```



3.Check the received data is acceptable at receiver end or not using for CRC checking algorithm for a given pair of divisor and received data bit pattern

```
#include<stdio.h>
void main()
{
int i,j,k,flag;
int dw[10],div[4],crc[3];
printf("\n enter the data word(upto 10)");
for(i=0;i<10;i++)
{
scanf("%d",&dw[i]);
}
printf("\n enter the divisor");
for(i=0;i<4;i++)
{
scanf("%d",&div[i]);
}
for(i=10;i<14;i++){
   dw[i]=0;
}
for(i=0;i<10;i++)
{
        k=i;
```

```
if(dw[i]==1)
for(j=0;j<4;j++)
{
        if(dw[k]==div[j]){
                dw[k]=0;
                crc[j]=0;
        }
        else{
                dw[k]=1;
                crc[j]=1;
        }
        k++;
}
}
printf("\n crc=");
for(i=0;i<4;i++)
{
printf("%d",crc[i]);
}
for(i=0;i<4;i++)
{
if(crc[i]==1)
flag=1;
break;
}
if(flag==1)
printf("data word is corrupted");
else
printf("data is correct");
printf("\n");
}
```



COMPUTER NETWORK LAB ASSIGNMENT-2

Soumik Roy
12000118029
CSE-1, 3rd Yr

➔ Write an implementation of a simple TCP connection and display the IP address and PORT of the client connecting

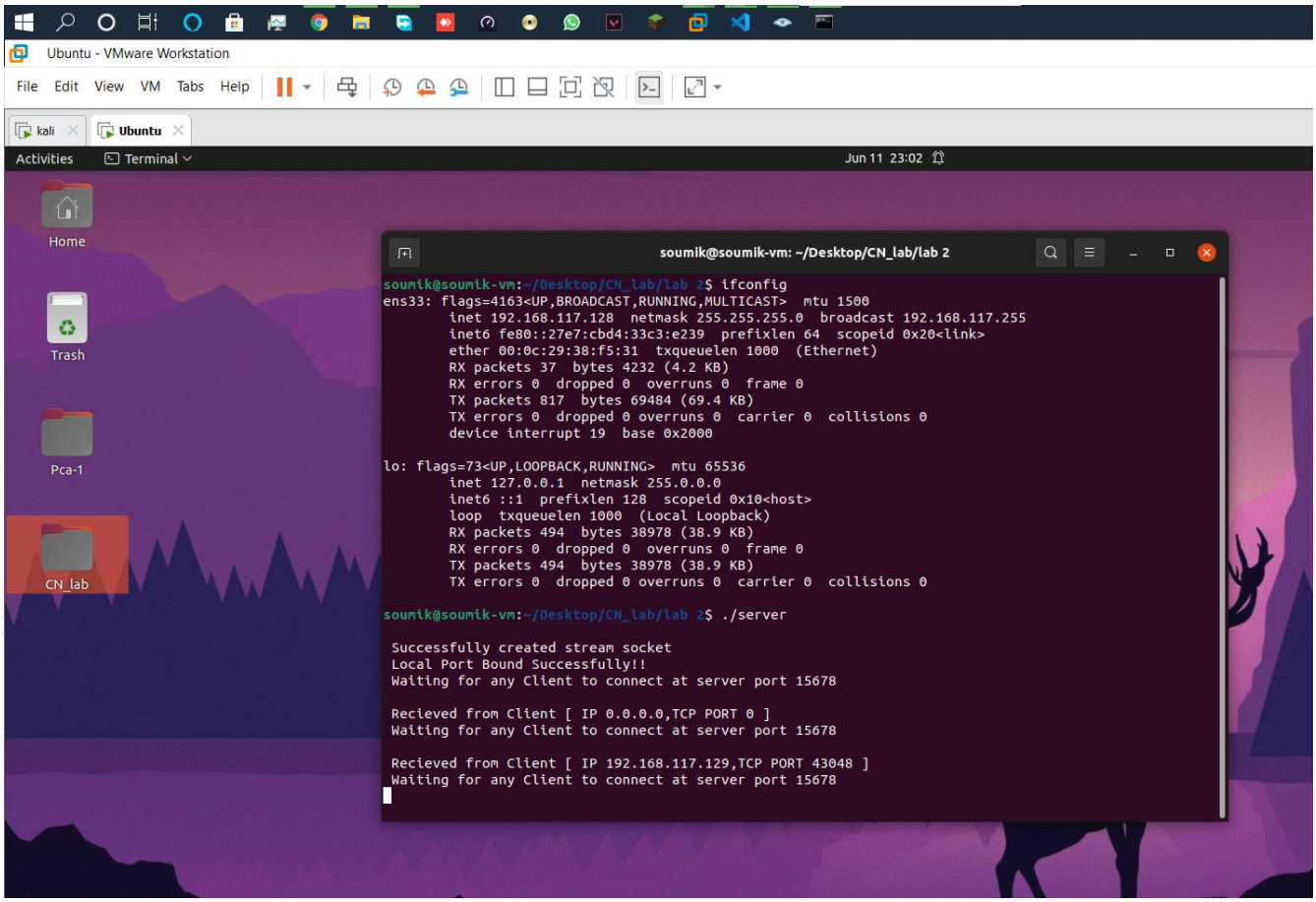
server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>

#define SERVER_ADDR "192.168.117.128"
#define SERVER_PORT 15678

int main()
{
int sd,newsd,cli_len,n;
struct sockaddr_in cli_addr,serv_addr;
bzero((char*) &serv_addr,sizeof(serv_addr));
serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr =inet_addr(SERVER_ADDR);
serv_addr.sin_port = htons(SERVER_PORT);
sd=socket(AF_INET,SOCK_STREAM, 0);

printf("\n Successfully created stream socket");
bind(sd,(struct sockaddr*)&serv_addr,sizeof (cli_addr));
printf("\n Local Port Bound Successfully!!");
listen (sd,2);

while(1)
{
printf("\n Waiting for any Client to connect at server port %u \n",SERVER_PORT);
newsd = accept (sd,(struct sockaddr*)&cli_addr,&cli_len);
printf("\n Recieved from Client [ IP %s,TCP PORT %d ]",inet_ntoa(cli_addr.sin_addr),ntohs(cli_addr.sin_port));
close(newsd);
}

close(sd);
printf("\n\n");

}
```

client.c

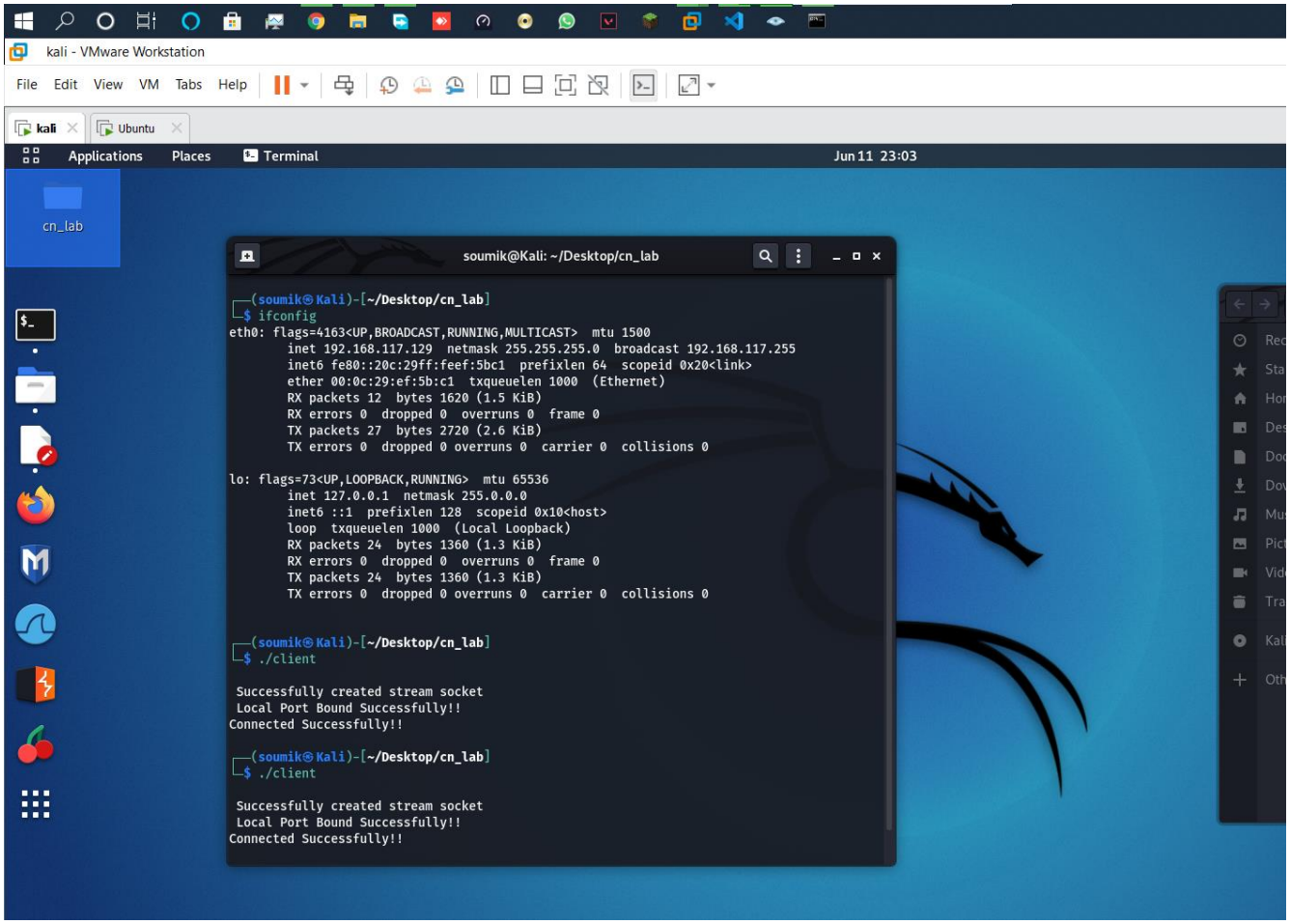
```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>

#define SERVER_ADDR "192.168.117.128"
#define SERVER_PORT 15678
#define CLIENT_ADDR "192.168.117.129"
#define CLIENT_PORT 15678

int main()
{
int sd;
struct sockaddr_in serv_addr,cli_addr;
bzero((char*) &cli_addr,sizeof(cli_addr));
bzero((char*) &serv_addr,sizeof(serv_addr));

serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr =inet_addr(SERVER_ADDR);
serv_addr.sin_port = htons(SERVER_PORT);

cli_addr.sin_family = AF_INET;
cli_addr.sin_addr.s_addr =inet_addr(CLIENT_ADDR);
cli_addr.sin_port = htons(CLIENT_PORT);
sd=socket(AF_INET,SOCK_STREAM, 0);

printf("\n Successfully created stream socket");
bind(sd,(struct sockaddr*)&cli_addr,sizeof (cli_addr));
printf("\n Local Port Bound Successfully!!");
connect(sd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));
printf("\nConnected Successfully!!");
close(sd);
}
```

server.c



---

client.c



COMPUTER NETWORKING LAB ASSIGNMENT-3

Soumik Roy
12000118029
CSE-1, 3rd Yr

➔ Write program using sockets to implement an Echo-Server.

server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<math.h>

#define SERVER_ADDR "192.168.117.128"
#define SERVER_PORT 15678
#define MAX_MSG 100

int main()
{
int sd,newsd,cli_len,n;
char line[MAX_MSG];
struct sockaddr_in cli_addr,serv_addr;
bzero((char*) &serv_addr,sizeof(serv_addr));
serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr =inet_addr(SERVER_ADDR);
serv_addr.sin_port = htons(SERVER_PORT);
sd=socket(AF_INET,SOCK_STREAM, 0);

printf("Successfully created stream socket\n");
bind(sd,(struct sockaddr*)&serv_addr,sizeof (serv_addr));
printf("Local Port Bound Successfully!!\n");
listen (sd,1);

while(1)
{
fflush(stdin);
printf("Waiting for any Client to connect at server port. . . %u \n",SERVER_PORT);
newsd = accept (sd,(struct sockaddr*)&cli_addr,&cli_len);
fflush(stdin);

do
{
fflush(stdin);
memset(line,0x0,MAX_MSG);
n=recv(newsd,line,(strlen(line)+1),0);
line[n]='\n';
```

```
printf("Recieved from HOST : %s\n",line);
send(newsd,line,strlen(line)+1,0);
fflush(stdin);
}while(strcmp(line,"quit"));

printf("CLOSING connection with host. . . .\n");
close(newsd);
printf("\n\n");
return 0;
}
}
```

client.c

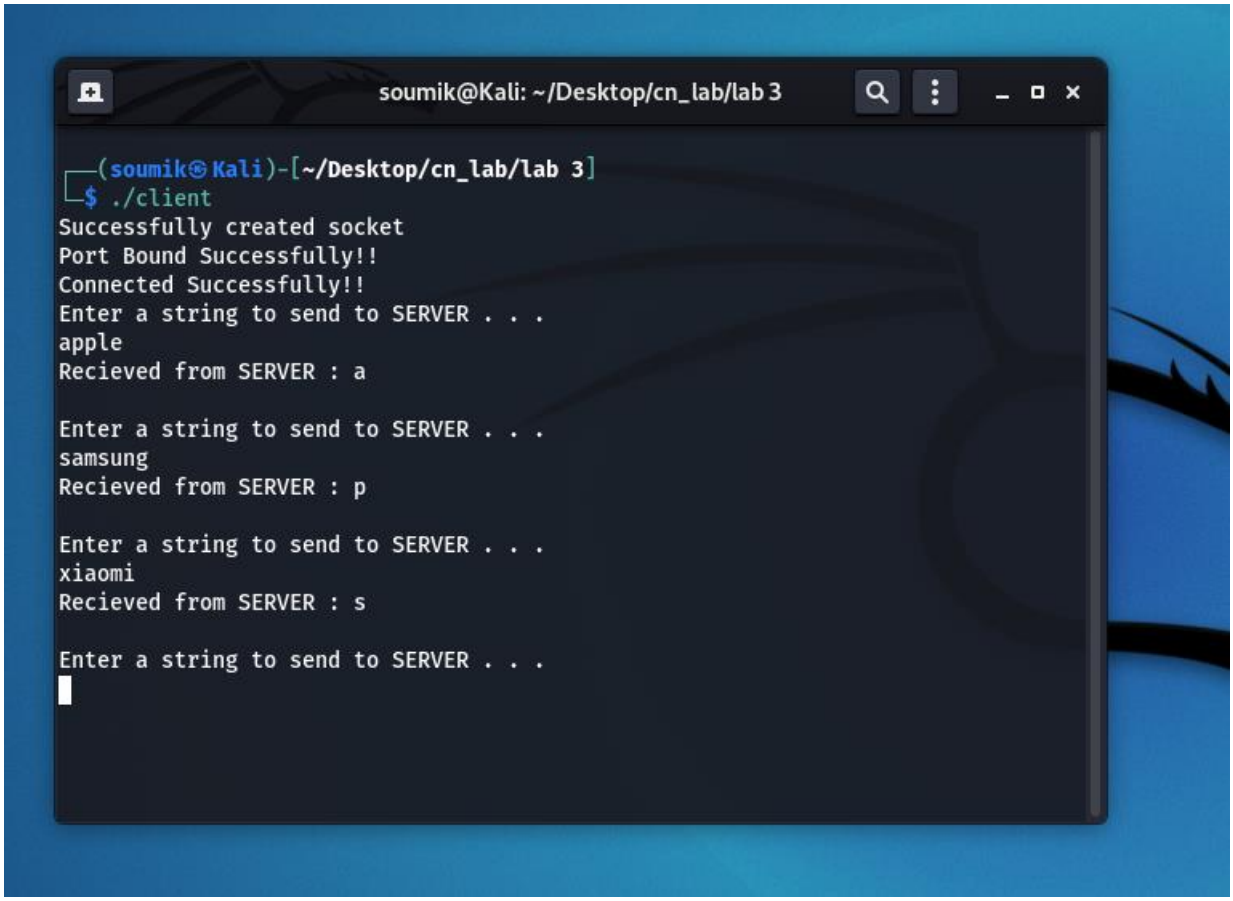
```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>

#define SERVER_ADDR "192.168.117.128"
#define SERVER_PORT 15678
#define CLIENT_ADDR "192.168.117.129"
#define CLIENT_PORT 15678
#define MAX_MSG 100

int main()
{
int sd,n,newsd;
char line[MAX_MSG],line1[MAX_MSG];
struct sockaddr_in serv_addr,cli_addr;

bzero((char*) &serv_addr,sizeof(serv_addr));

serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr =inet_addr(SERVER_ADDR);
serv_addr.sin_port = htons(SERVER_PORT);
bzero((char*) &cli_addr,sizeof(cli_addr));

cli_addr.sin_family = AF_INET;
cli_addr.sin_addr.s_addr =inet_addr(CLIENT_ADDR);
cli_addr.sin_port = htons(CLIENT_PORT);
sd=socket(AF_INET,SOCK_STREAM, 0);
```

```
fflush(stdin);
printf("Successfully created socket\n");
bind(sd,(struct sockaddr*)&cli_addr,sizeof(cli_addr));
printf("Port Bound Successfully!!\n");
connect(sd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));
printf("Connected Successfully!!\n");
fflush(stdin);

do
{
printf("Enter a string to send to SERVER . . .\n");
scanf("%s",line);
send(sd,line,(strlen(line)+1),0);
memset(line1,0x0,MAX_MSG);
n=recv(sd,line1,MAX_MSG,0);
line1[n]='\n';
printf("Recieved from SERVER : %s\n",line1);
fflush(stdin);
}while(strcmp(line1,"quit"));

close(newsd);
printf("\n\n");
return 0;
}
```

server.c

```
soumik@soumik-vm: ~/Desktop/CN_lab/lab 3
soumik@soumik-vm:~/Desktop/CN_lab/lab 3$ gcc server.c -o server
soumik@soumik-vm:~/Desktop/CN_lab/lab 3$ ./server
Successfully created stream socket
Local Port Bound Successfully!!
Waiting for any Client to connect at server port. . . 15678
Recieved from HOST : a

Recieved from HOST : p

Recieved from HOST : p

Recieved from HOST : l

Recieved from HOST : e

Recieved from HOST :
Recieved from HOST : s

Recieved from HOST : a

Recieved from HOST : m

Recieved from HOST : s

Recieved from HOST : u

Recieved from HOST : n

Recieved from HOST : g

Recieved from HOST :
Recieved from HOST : x

Recieved from HOST : i

Recieved from HOST : a

Recieved from HOST : o

Recieved from HOST : m

Recieved from HOST : i

Recieved from HOST :
```

client.c


soumik@Kali: ~/Desktop/cn_lab/lab 3
(soumik㉿Kali)-[~/Desktop/cn_lab/lab 3]
$ ./client
Successfully created socket
Port Bound Successfully!!
Connected Successfully!!
Enter a string to send to SERVER . . .
apple
Recieved from SERVER : a
Enter a string to send to SERVER . . .
samsung
Recieved from SERVER : p
Enter a string to send to SERVER . . .
xiaomi
Recieved from SERVER : s
Enter a string to send to SERVER . . .

COMPUTER NETWORKING LAB ASSIGNMENT-4

Soumik Roy
12000118029
CSE-1, 3rd Yr

➔ Write C programs for both server and client using UDP socket, so that the message sent by the client is received at server's end and displayed .

server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define PORT     8080
#define MAXLINE 100
#define SERVER_ADDR "192.168.92.129
int main()
{
   int sockfd;
   char buffer[MAXLINE];

   struct sockaddr_in servaddr, cliaddr;
   memset(&servaddr, 0, sizeof(servaddr));
   memset(&cliaddr, 0, sizeof(cliaddr));

   servaddr.sin_family    = AF_INET;
   servaddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
   servaddr.sin_port = htons(PORT);

   if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
      perror("socket creation failed");
      exit(EXIT_FAILURE);
   }
   else
   {
         printf("\nSuccessfully created DATAGRAM socket\n");
   }

   if ( bind(sockfd, (const struct sockaddr *)&servaddr,sizeof(servaddr)) < 0 )
   {
      perror("bind failed");
      exit(EXIT_FAILURE);
   }
   else
   {
         printf("\nLocal Port Bound Successfully!!\n");
```

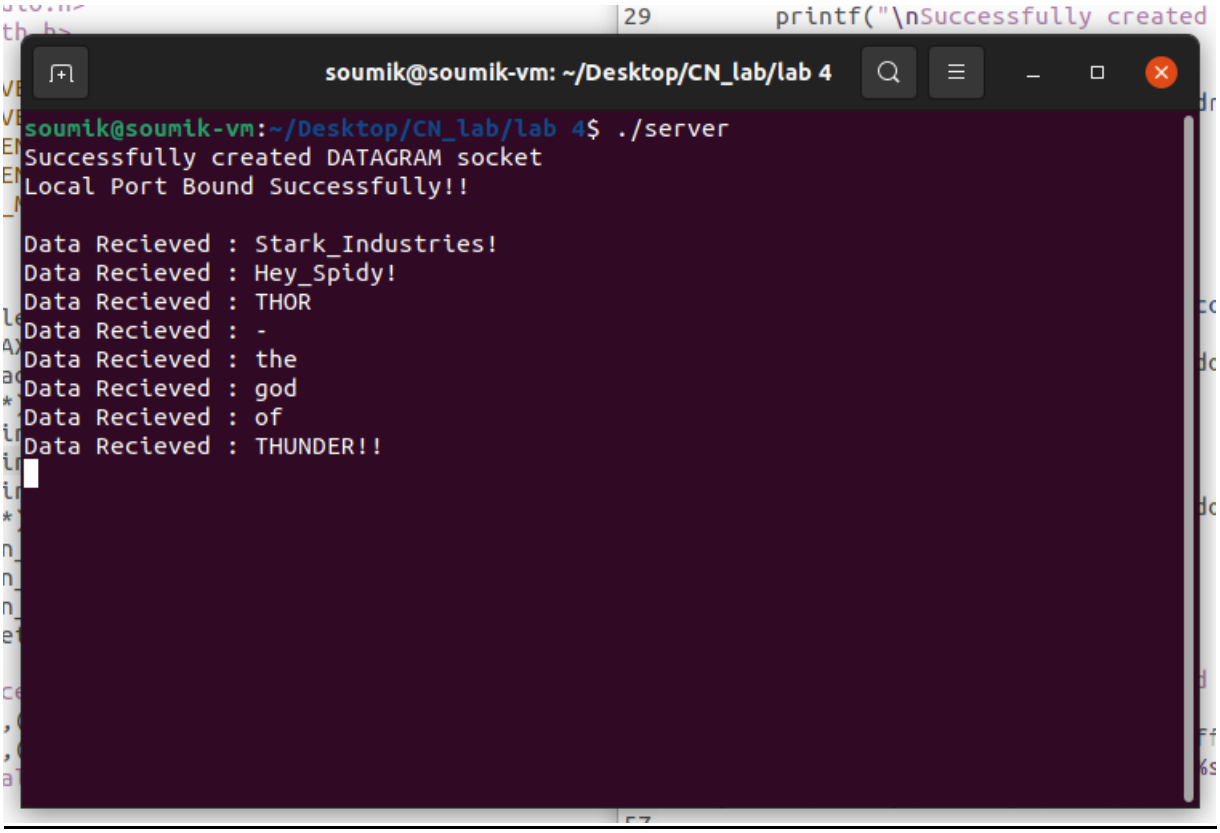
```
    }
    int len, n;
    while(1)
    {
    len = sizeof(cliaddr);
    n = recvfrom(sockfd, (char *)buffer, MAXLINE,0, ( struct sockaddr *) &cliaddr,&len);
    buffer[n] = '\0';
    printf("\nData Recieved : %s",buffer);
    }
     return 0;
}
```

client.c

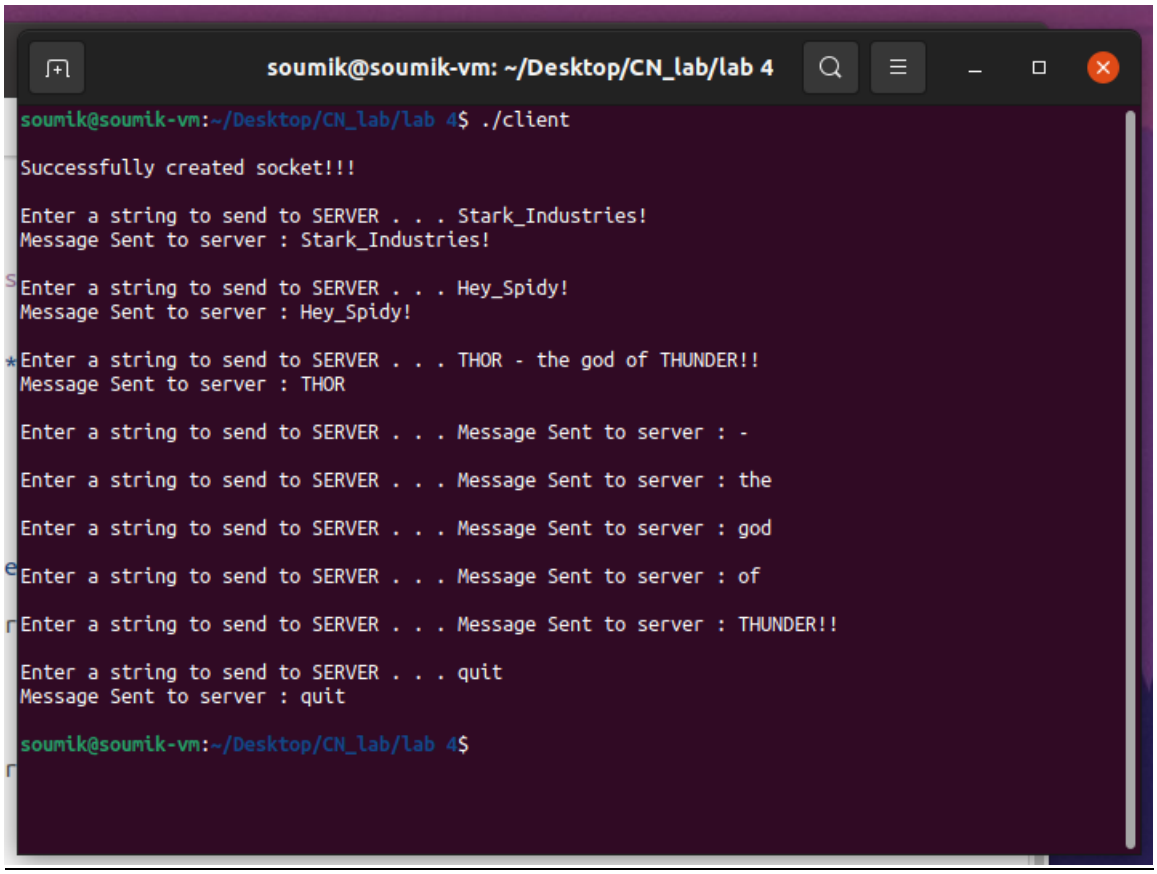
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define PORT     8080
#define MAXLINE 100
#define CLIENT_ADDR "192.168.92.129"

int main()
{
    int sockfd;
    char buffer[MAXLINE];

    struct sockaddr_in servaddr,cli_addr;


    if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 )
    {
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }
    else
    {
            printf("\nSuccessfully created socket!!!");
     }
    memset(&servaddr, 0, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    servaddr.sin_addr.s_addr = inet_addr(CLIENT_ADDR);
    int n, len;
```

```
    do
    {
    printf("\n\nEnter a string to send to SERVER . . . ");
    scanf("%s",buffer);
    sendto(sockfd, buffer, (strlen(buffer)+1),0, (const struct sockaddr *) &servaddr,sizeof(servaddr));
    printf("Message Sent to server : %s",buffer);
    }while(strcmp(buffer,"quit"));

    printf("\n\n");
    close(sockfd);
    return 0;
}
```

SERVER



CLIENT


soumik@soumik-vm: ~/Desktop/CN_lab/lab 4
soumik@soumik-vm:~/Desktop/CN_lab/lab 4$ ./client
Successfully created socket!!!
Enter a string to send to SERVER . . . Stark_Industries!
Message Sent to server : Stark_Industries!
Enter a string to send to SERVER . . . Hey_Spidy!
Message Sent to server : Hey_Spidy!
Enter a string to send to SERVER . . . THOR - the god of THUNDER!!
Message Sent to server : THOR
Enter a string to send to SERVER . . . Message Sent to server : -
Enter a string to send to SERVER . . . Message Sent to server : the
Enter a string to send to SERVER . . . Message Sent to server : god
Enter a string to send to SERVER . . . Message Sent to server : of
Enter a string to send to SERVER . . . Message Sent to server : THUNDER!!
Enter a string to send to SERVER . . . quit
Message Sent to server : quit
soumik@soumik-vm:~/Desktop/CN_lab/lab 4$

COMPUTER NETWORKING LAB ASSIGNMENT-5

Soumik Roy
12000118029
CSE-1, 3rd Yr

Write a Server and a Client Program using TCP socket for implementing chat operation between server and client until the client enters "stop" message to quit the communication. Also attach your output of both server and client end.

server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<math.h>

#define SERVER_ADDR "192.168.117.128"
#define SERVER_PORT 15678
#define MAX_MSG 100

int main()
{
int sd,newsd,cli_len,n;
char line[MAX_MSG];
struct sockaddr_in cli_addr,serv_addr;
bzero((char*) &serv_addr,sizeof(serv_addr));
serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr =inet_addr(SERVER_ADDR);
serv_addr.sin_port = htons(SERVER_PORT);
sd=socket(AF_INET,SOCK_STREAM, 0);

printf("Successfully created stream socket\n");
bind(sd,(struct sockaddr*)&serv_addr,sizeof (serv_addr));
printf("Local Port Bound Successfully!!\n");
listen (sd,1);

while(1)
{
fflush(stdin);
printf("Waiting for any Client to connect at server port. . . %u \n",SERVER_PORT);
newsd = accept (sd,(struct sockaddr*)&cli_addr,&cli_len);
fflush(stdin);

do
{
fflush(stdin);
memset(line,0x0,MAX_MSG);
n=recv(newsd,line,(strlen(line)+1),0);
```

```
line[n]='\n';
printf("Recieved from HOST : %s\n",line);
send(newsd,line,strlen(line)+1,0);
fflush(stdin);
}while(strcmp(line,"quit"));

printf("CLOSING connection with host. . . .\n");
close(newsd);
printf("\n\n");
return 0;
}
}
```

client.c

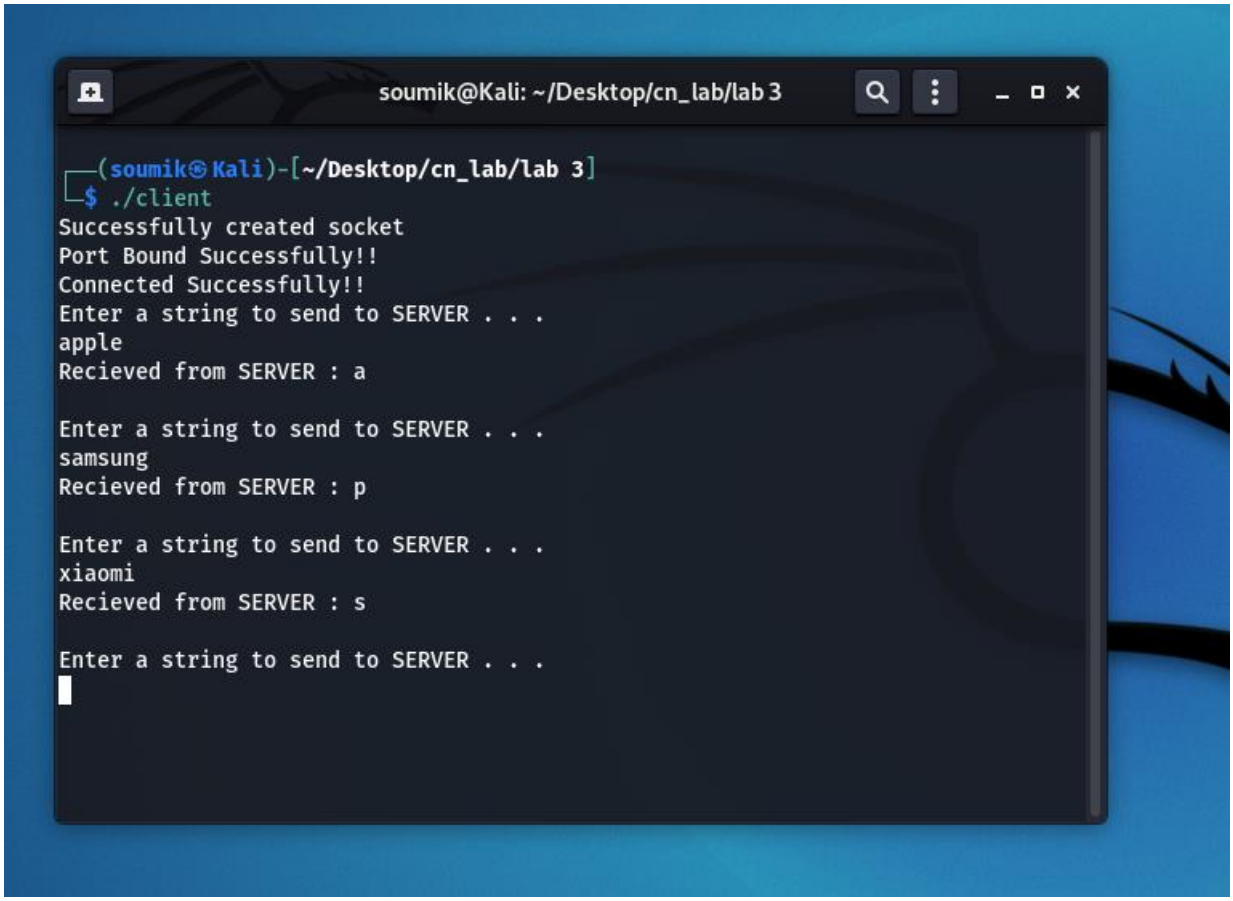
```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>

#define SERVER_ADDR "192.168.117.128"
#define SERVER_PORT 15678
#define CLIENT_ADDR "192.168.117.129"
#define CLIENT_PORT 15678
#define MAX_MSG 100

int main()
{
int sd,n,newsd;
char line[MAX_MSG],line1[MAX_MSG];
struct sockaddr_in serv_addr,cli_addr;

bzero((char*) &serv_addr,sizeof(serv_addr));

serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr =inet_addr(SERVER_ADDR);
serv_addr.sin_port = htons(SERVER_PORT);
bzero((char*) &cli_addr,sizeof(cli_addr));

cli_addr.sin_family = AF_INET;
cli_addr.sin_addr.s_addr =inet_addr(CLIENT_ADDR);
cli_addr.sin_port = htons(CLIENT_PORT);
sd=socket(AF_INET,SOCK_STREAM, 0);
```

```
fflush(stdin);
printf("Successfully created socket\n");
bind(sd,(struct sockaddr*)&cli_addr,sizeof(cli_addr));
printf("Port Bound Successfully!!\n");
connect(sd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));
printf("Connected Successfully!!\n");
fflush(stdin);

do
{
printf("Enter a string to send to SERVER . . .\n");
scanf("%s",line);
send(sd,line,(strlen(line)+1),0);
memset(line1,0x0,MAX_MSG);
n=recv(sd,line1,MAX_MSG,0);
line1[n]='\n';
printf("Recieved from SERVER : %s\n",line1);
fflush(stdin);
}while(strcmp(line1,"quit"));

close(newsd);
printf("\n\n");
return 0;
}
```

server.c

```
soumik@soumik-vm: ~/Desktop/CN_lab/lab 3
soumik@soumik-vm:~/Desktop/CN_lab/lab 3$ gcc server.c -o server
soumik@soumik-vm:~/Desktop/CN_lab/lab 3$ ./server
Successfully created stream socket
Local Port Bound Successfully!!
Waiting for any Client to connect at server port. . . 15678
Recieved from HOST : a

Recieved from HOST : p

Recieved from HOST : p

Recieved from HOST : l

Recieved from HOST : e

Recieved from HOST :
Recieved from HOST : s

Recieved from HOST : a

Recieved from HOST : m

Recieved from HOST : s

Recieved from HOST : u

Recieved from HOST : n

Recieved from HOST : g

Recieved from HOST :
Recieved from HOST : x

Recieved from HOST : i

Recieved from HOST : a

Recieved from HOST : o

Recieved from HOST : m

Recieved from HOST : i

Recieved from HOST :
```

client.c


soumik@Kali: ~/Desktop/cn_lab/lab 3
(soumik㉿Kali)-[~/Desktop/cn_lab/lab 3]
$ ./client
Successfully created socket
Port Bound Successfully!!
Connected Successfully!!
Enter a string to send to SERVER . . .
apple
Recieved from SERVER : a
Enter a string to send to SERVER . . .
samsung
Recieved from SERVER : p
Enter a string to send to SERVER . . .
xiaomi
Recieved from SERVER : s
Enter a string to send to SERVER . . .

COMPUTER NETWORKING LAB ASSIGNMENT-6

Soumik Roy
12000118029
CSE-1, 3rd Yr

➔ Write an echo server program using TCP socket for handling concurrent requests from more than one client. Execute your code and show the output.

Server

server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

#define SERVER_ADDR "192.168.92.129"
#define SERVER_PORT 15555
#define MAX_MSG 100

int main()
{
 int pid, sd, newsd, clilen, n;
 char line[MAX_MSG];
 struct sockaddr_in cli_addr, serv_addr;
 bzero((char*) &serv_addr, sizeof(serv_addr));
 serv_addr.sin_family = AF_INET;
 serv_addr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 serv_addr.sin_port = htons(SERVER_PORT);
 sd=socket(AF_INET, SOCK_STREAM, 0);

 printf("\nSuccessfully created stream socket");
 bind(sd, (struct sockaddr*)&serv_addr, sizeof(serv_addr));
 printf("\nLocal port bound successfully!");
 listen(sd,5);
 while(1)
 {
 printf("\nWaiting for any client to connect at server port %u \n", SERVER_PORT);
 newsd= accept(sd,(struct sockaddr*)&cli_addr, &clilen);
 pid = fork();
 if(pid == 0)
{
 do
 {
 memset(line, 0x0, MAX_MSG);
 n=recv(newsd,line,(strlen(line)+1),0);
 line[n]='\n';
```

```
 printf("\nReceived from host: %s\n", line);
 send(newsd, line, strlen(line)+1, 0);
 }
while(strcmp(line, "quit"));
printf("Closing connection with host\n");
}
else if(pid< 0)
{
printf("\n Connection cannot be created\n");
close(newsd);
}
else
 close(newsd);
 }
 return 0;
}
```

1st Client

client1.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

#define SERVER_ADDR "192.168.92.129"
#define SERVER_PORT 15555
#define CLIENT_ADDR "192.168.92.129"
#define CLIENT_PORT 15555
#define MAX_MSG 100

int main()
{
 int sd, n;
 struct sockaddr_in cliaddr,servaddr ;
 char line[MAX_MSG], line1[MAX_MSG];
 bzero((char*) &servaddr, sizeof(servaddr));
 servaddr.sin_family = AF_INET;
 servaddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 servaddr.sin_port = htons(SERVER_PORT);

 bzero((char*) &cliaddr, sizeof(cliaddr));
 cliaddr.sin_family = AF_INET;
 servaddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 cliaddr.sin_port = htons(CLIENT_PORT);
 sd=socket(AF_INET, SOCK_STREAM, 0);
 printf("\nSuccessfully created stream socket");
```

```
 bind(sd, (struct sockaddr*)&cliaddr, sizeof(cliaddr));
 connect(sd, (struct sockaddr*)&servaddr, sizeof(servaddr));
 printf("\nConnected Successfully!");
 do
 {
 printf("\nEnter a string to send to server:  ");
 scanf("%s",line);
 send(sd, line, (strlen(line)+1), 0);
 memset(line1, 0x0, MAX_MSG);
 n = recv(sd, line1, MAX_MSG,0);
 line1[n]='\n';
 printf("Received from server: %s", line1);
 } while(strcmp(line,"quit"));

 printf("\nClosing connection with server.\n");
 close(sd);
 return 0;
}
```

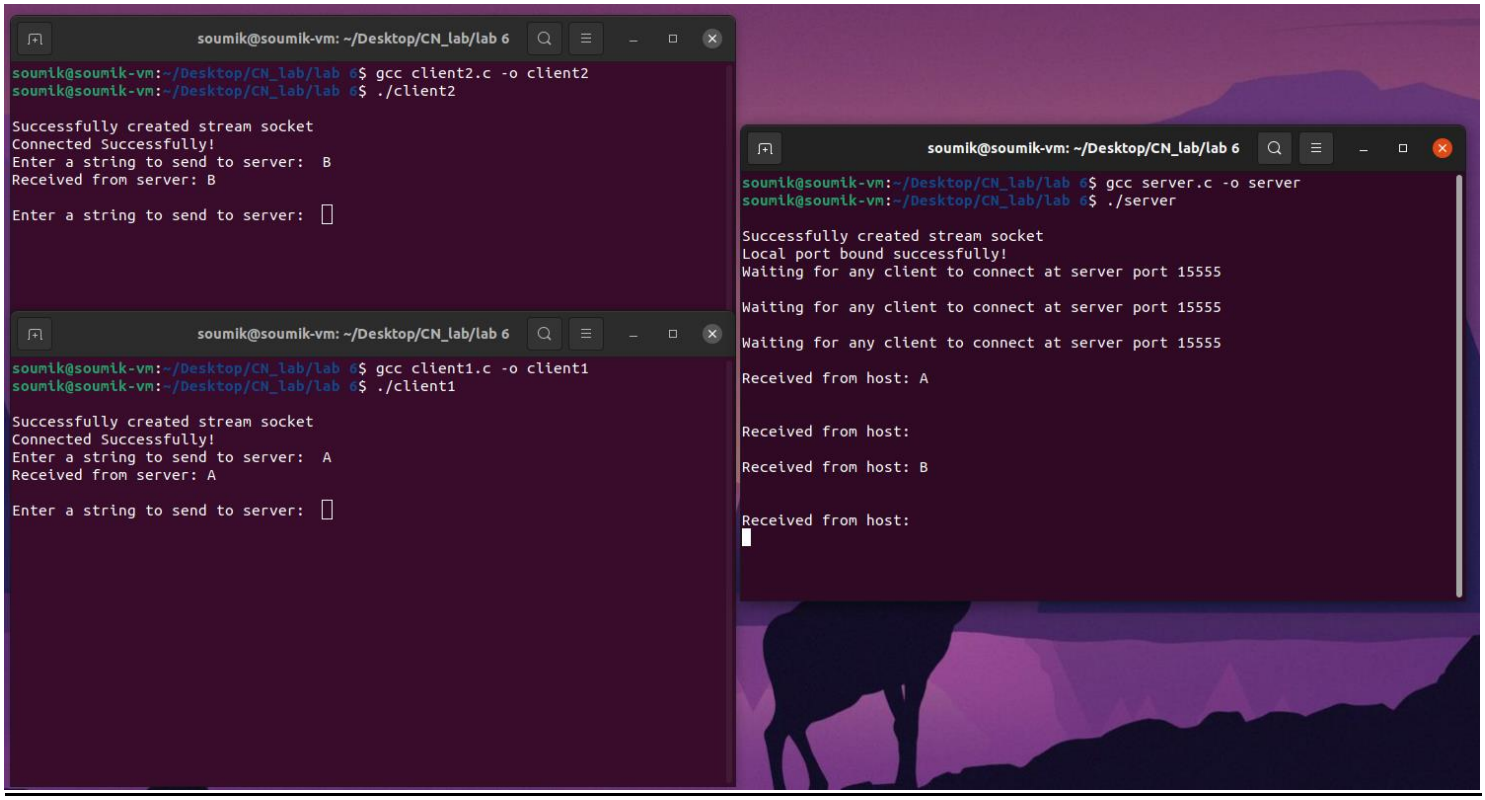
2nd Client

client2.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

#define SERVER_ADDR "192.168.92.129"
#define SERVER_PORT 15555
#define CLIENT_ADDR "192.168.92.129"
#define CLIENT_PORT 15555
#define MAX_MSG 100

int main()
{
 int sd, n;
 struct sockaddr_in cliaddr,servaddr ;
 char line[MAX_MSG], line1[MAX_MSG];
 bzero((char*) &servaddr, sizeof(servaddr));
 servaddr.sin_family = AF_INET;
 servaddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 servaddr.sin_port = htons(SERVER_PORT);

 bzero((char*) &cliaddr, sizeof(cliaddr));
 cliaddr.sin_family = AF_INET;
 servaddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 cliaddr.sin_port = htons(CLIENT_PORT);
```

```
sd=socket(AF_INET, SOCK_STREAM, 0);
printf("\nSuccessfully created stream socket");
bind(sd, (struct sockaddr*)&cliaddr, sizeof(cliaddr));
connect(sd, (struct sockaddr*)&servaddr, sizeof(servaddr));
printf("\nConnected Successfully!");
do
{
printf("\nEnter a string to send to server:  ");
scanf("%s",line);
send(sd, line, (strlen(line)+1), 0);
memset(line1, 0x0, MAX_MSG);
n = recv(sd, line1, MAX_MSG,0);
line1[n]='\n';
printf("Received from server: %s", line1);
} while(strcmp(line,"quit"));

printf("\nClosing connection with server.\n");
close(sd);
return 0;
}
```

Output



COMPUTER NETWORKING LAB ASSIGNMENT-7

Soumik Roy
12000118029
CSE-1, 3rd Yr

WRITE A SERVER AND A CLIENT SOCKET PROGRAM USING C TO SHOW THE SYSTEM TIME OF SERVER AT THE CLIENT END.

Server

server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

#define SERVER_ADDR "192.168.92.129"
#define SERVER_PORT 15555
#define MAX_MSG 100

int main()
{
 int sd, newsd, clilen, n;
 int pts;
 char line[MAX_MSG];
 time_t t;
 struct sockaddr_in cli_addr, serv_addr;
 bzero((char*) &serv_addr, sizeof(serv_addr));
 serv_addr.sin_family = AF_INET;
 serv_addr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 serv_addr.sin_port = htons(SERVER_PORT);
 sd=socket(AF_INET, SOCK_STREAM, 0);

 printf("\nSuccessfully created stream socket");
 bind(sd, (struct sockaddr*)&serv_addr, sizeof(serv_addr));
 printf("\nLocal port bound successfully!");
 listen(sd,5);
 while(1)
 {
 printf("\nWaiting for any client to connect at server port %u \n", SERVER_PORT);
 newsd= accept(sd,(struct sockaddr*)&cli_addr, &clilen);
 t = time(NULL);
 pts=time(&t);
 line[0]=pts;
 printf("%d \n",pts);
 send(newsd,line,strlen(line)+1,0);
 printf("\n closing. . . .");
 }
```

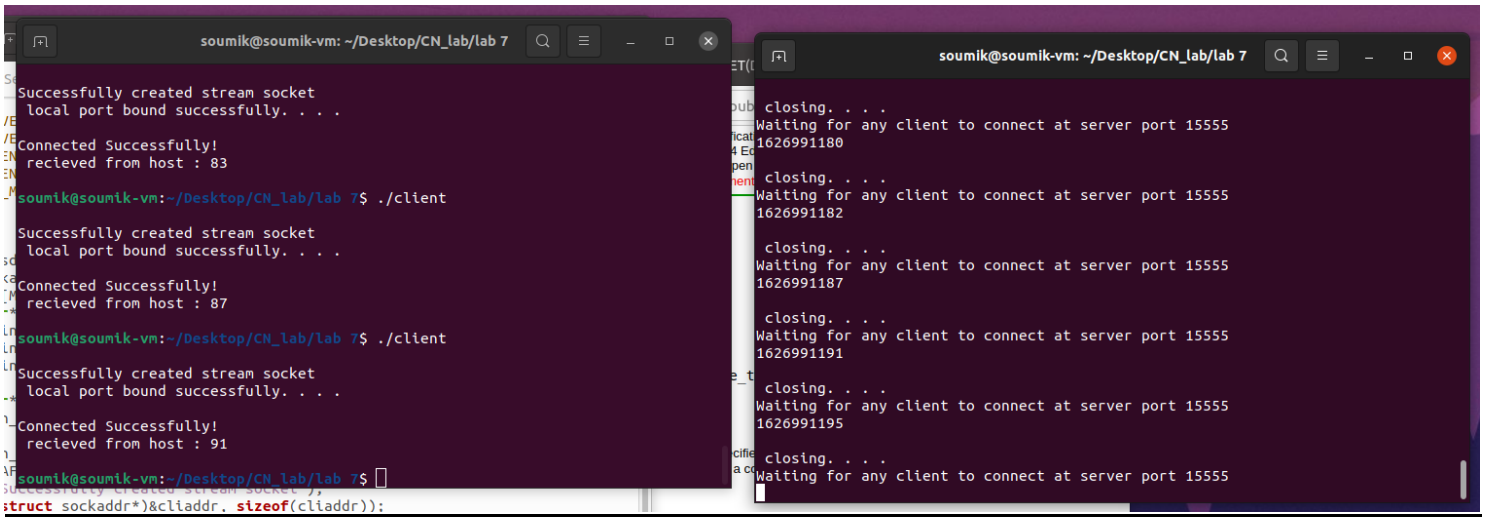
```
    close(newsd);

    return 0;
   }
```

Client

client.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<unistd.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

#define SERVER_ADDR "192.168.92.129"
#define SERVER_PORT 15555
#define CLIENT_ADDR "192.168.92.129"
#define CLIENT_PORT 15555
#define MAX_MSG 100

int main()
{
 int sd,newsd,clilen,n;
 struct sockaddr_in cliaddr,servaddr ;
 short line[MAX_MSG];
 bzero((char*) &servaddr, sizeof(servaddr));
 servaddr.sin_family = AF_INET;
 servaddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
 servaddr.sin_port = htons(SERVER_PORT);

 bzero((char*) &cliaddr, sizeof(cliaddr));
 cliaddr.sin_family = AF_INET;

 cliaddr.sin_port = htons(CLIENT_PORT);
 sd=socket(AF_INET, SOCK_STREAM, 0);
 printf("\nSuccessfully created stream socket");
 bind(sd, (struct sockaddr*)&cliaddr, sizeof(cliaddr));
 printf("\n local port bound successfully. . . . \n");
 connect(sd, (struct sockaddr*)&servaddr, sizeof(servaddr));
 printf("\nConnected Successfully!");
 memset(line,0x0,1);
 n=recv(sd,line,sizeof(line)+1,0);
 line[n]='\n';
 printf("\n recieved from host : %u",line[0]);
 close(sd);
 printf("\n\n");
 return 0;
}
```

Output


soumik@soumik-vm: ~/Desktop/CN_lab/lab 7
Successfully created stream socket
local port bound successfully. . . .
Connected Successfully!
recieved from host : 83
soumik@soumik-vm:~/Desktop/CN_lab/lab 7$ ./client
Successfully created stream socket
local port bound successfully. . . .
Connected Successfully!
recieved from host : 87
soumik@soumik-vm:~/Desktop/CN_lab/lab 7$ ./client
Successfully created stream socket
local port bound successfully. . . .
Connected Successfully!
recieved from host : 91
soumik@soumik-vm:~/Desktop/CN_lab/lab 7$
struct sockaddr*)&cliaddr, sizeof(cliaddr));
soumik@soumik-vm: ~/Desktop/CN_lab/lab 7
closing. . . .
Waiting for any client to connect at server port 15555
1626991180
closing. . . .
Waiting for any client to connect at server port 15555
1626991182
closing. . . .
Waiting for any client to connect at server port 15555
1626991187
closing. . . .
Waiting for any client to connect at server port 15555
1626991191
closing. . . .
Waiting for any client to connect at server port 15555
1626991195
closing. . . .
Waiting for any client to connect at server port 15555