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

int main( int argc, char *argv[])
{
struct sockaddr_in server;
int sd ;
char buffer[200];
if((sd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
{
perror("Socket failed:");
exit(1);
}
// server socket address structure initialisation
bzero(&server, sizeof(server) );
server.sin_family = AF_INET;
server.sin_port = htons(atoi(argv[2]));
inet_pton(AF_INET, argv[1], &server.sin_addr);
if(connect(sd, (struct sockaddr *)&server, sizeof(server))< 0)
{
perror("Connection failed:");
exit(1);
}
fgets(buffer, sizeof(buffer), stdin);
buffer[strlen(buffer) - 1] = '\0';
write (sd,buffer, sizeof(buffer));
read(sd,buffer, sizeof(buffer));
printf("%s\n", buffer);
close(fd);
}
```

Server Program

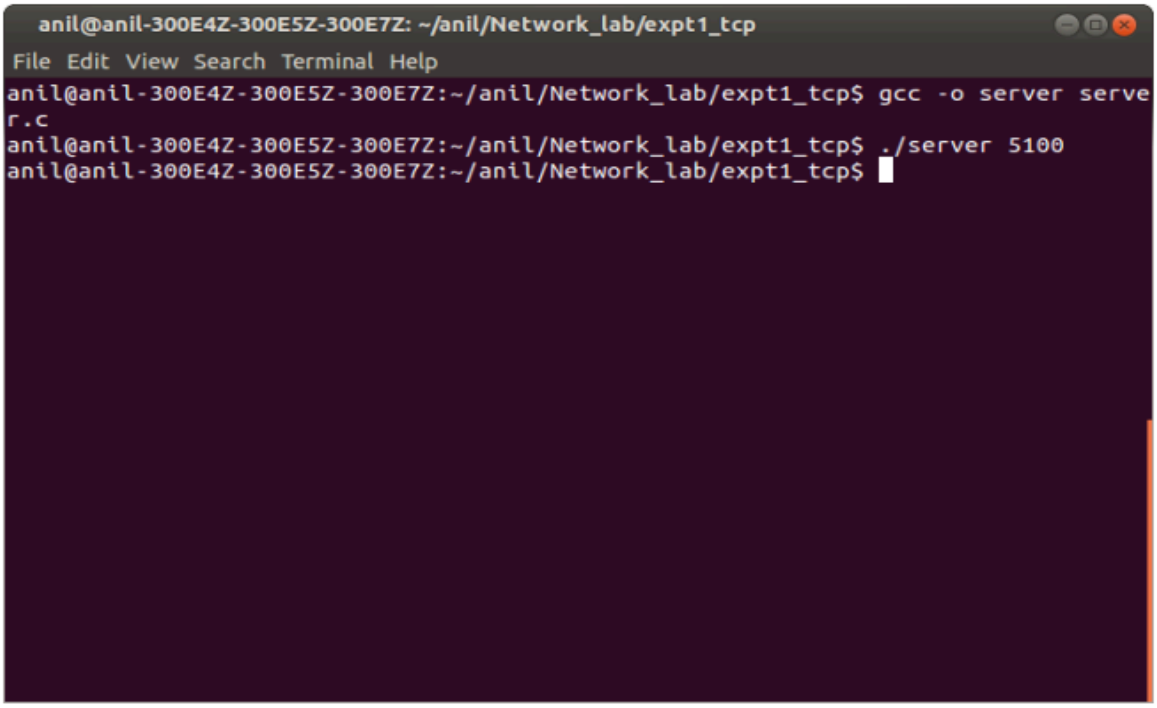
```
#include<stdio.h>
```

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

int main( int argc, char *argv[])
{
struct sockaddr_in server, cli;
int cli_len;
int sd, n, i, len;
int data, temp;
char buffer[100];
if((sd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
{
perror("Socket failed:");
exit(1);
}

// server socket address structure initialisation
bzero(&server, sizeof(server) );
server.sin_family = AF_INET;
server.sin_port = htons(atoi(argv[1]));
server.sin_addr.s_addr = htonl(INADDR_ANY);
if(bind(sd, (struct sockaddr*)&server, sizeof(server)) < 0)
{
perror("bind failed:");
exit(1);
}
listen(sd,5);
if((data = accept(sd , (struct sockaddr *) &cli, &cli_len)) < 0)
{
perror("accept failed:");
exit(1);
}
read(data,buffer, sizeof(buffer));
len = strlen(buffer);
```

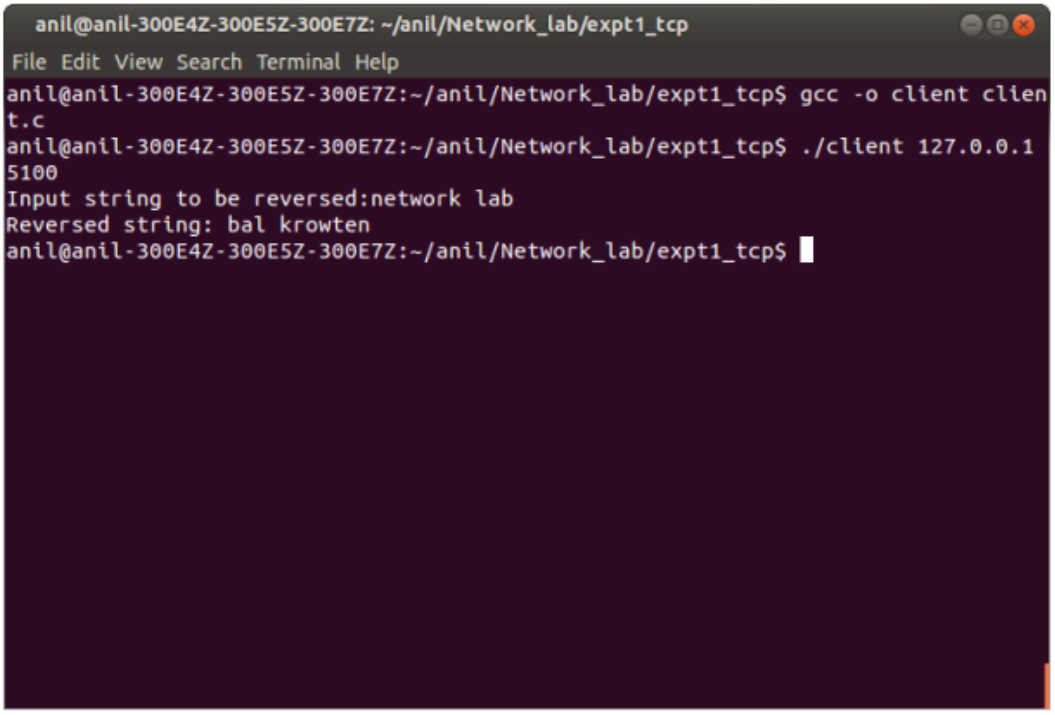
```
for( i =0; i<= len/2; i++)
{
temp = buffer[i];
buffer[i] = buffer[len - 1-i];
buffer[len-1-i] = temp;
}
write (data,buffer, sizeof(buffer));
close(data);
close(sd);
}
```

Output

Server



Client



Experiment 8

Implementation of Client-Server communication using Socket Programming and UDP as transport layer protocol

Aim: Client sends two matrices to the server using udp protocol. The server multiplies the matrices and sends the product to the client, which then displays the product matrix.

Description:

Steps for transfer of data using UDP

1. Creation of UDP socket

The function call for creating a UDP socket is

int socket(int domain, int type, int protocol);

The domain parameter specifies a communication domain; this selects the protocol family which will be used for communication. These families are defined in <sys/socket.h>. In this program,