Student Name	Samarth Abhay Kadam
SRN No	202200739
Roll No	10
Program	AI & DS
Year	Third Year
Division	A
Subject	Computer Network Laboratory (BTECCE21506)
Assignment No	Nine

Assignment Number - 09

Title : Socket Programming for TCP Client and TCP Server.

Problem Statement : Write a C /C++ / Java / Python socket program for TCP where TCP Client communicate with TCP server.

Theory :

A **network socket** is an endpoint of an inter-process communication flow across a computer network. Today, most communication between computers is based on the Internet Protocol; therefore most network sockets are **Internet sockets**.

A **socket API** is an application programming interface (API), usually provided by the operating system, that allows application programs to control and use network sockets. Internet socket APIs are usually based on the Berkeley sockets standard.

VISHWAKARMA UNIVERSITY Maximising Human Potential

A **socket address** is the combination of an IP address and a port number, much like one end of a telephone connection is the combination of a phone number and a particular extension. Based on this address, internet sockets deliver incoming data packets to the appropriate application process or thread.

Procedure in Client-Server Communication

There are some procedures that we have to follow to establish client-server communication. These are as follows.

Socket: With the help of a socket, we can create a new communication.

Bind: With the help of this we can, we can attach the local address with the socket.

Listen: With this help; we can accept the connection.

Accept: With this help; we can block the incoming connection until the request arrives.

Connect: With this help; we can attempt to establish the connection.

Send: With the help of this; we can send the data over the network.

Receive: With this help; we can receive the data over the network.

Close: With the help of this, we can release the connection from the network.

A connection is a type of relationship between two machines where the two software are known about each other. These two software know how to establish a connection with each other; in other words, we can say that this two software know how to send the bits over the network. A connection of the socket means the two machines should know all the information between each other, like the phone number, IP address, and the TCP port.

A socket is a type of object which is similar to the file that allows the program to accept the incoming connection and allow them to send or receive the incoming connection. Also, it is a type of resource assigned to the server's process.

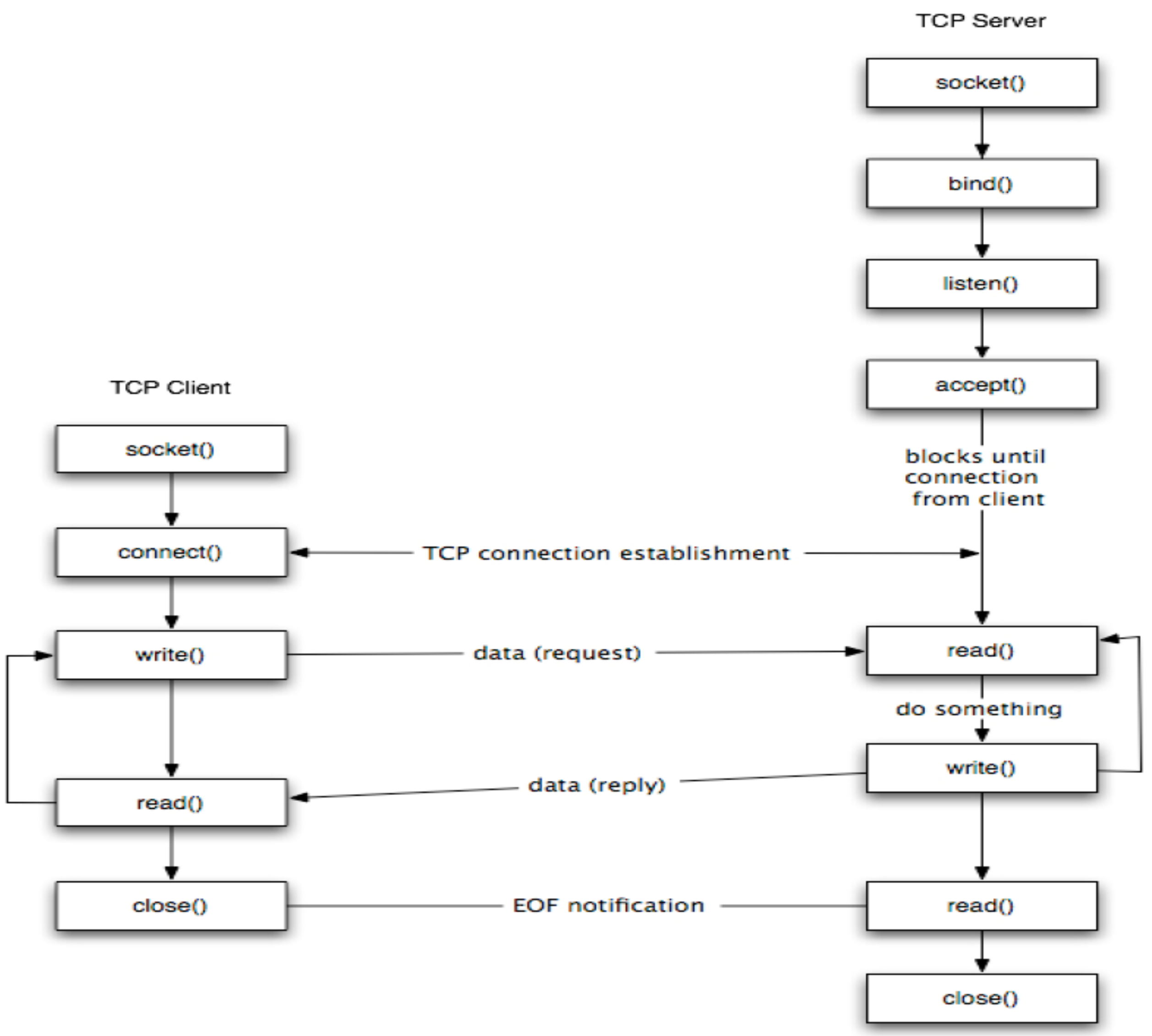


Figure : TCP Client Server Communication

Program :

```
//TCP SERVER PROGRAM
#include<unistd.h>
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<string.h>
#include<stdlib.h>
#include<pthread.h>
#define MAX 30
int ssfd;
int ccfd,res;
int serverlen,clientlen;
struct sockaddr_in serveraddr;
struct sockaddr_in clientaddr;
char result[MAX];
float addition (float a, float b);
typedef struct message
{
        float input[2];
        float output;
        int ch;
}message;
```

VISHWAKARMA UNIVERSITY Maximising Human Potential

```
void main()
{
        int msgLen;
        char *ch;
        ssfd=socket(AF_INET,SOCK_STREAM,0);
        message *msg;
        message *reply;
        msg=(message*)malloc(sizeof(message));
        reply=(message*)malloc(sizeof(message));

        printf("\nWaiting for connection \n SSFD Value:%d",ssfd);
        msgLen=sizeof(message);
        serveraddr.sin_family=AF_INET;
        serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
        serveraddr.sin_port=9734;
        serverlen=sizeof(serveraddr);
        bind(ssfd,(struct sockaddr*)&serveraddr,serverlen);
        printf("\nConnected to socket");
        listen(ssfd,5);
        ccfd=accept(ssfd,(struct sockaddr*)&clientaddr,&clientlen);
        printf("\nConnection accepted successfully \n");
        while(1)
        {
                read(ccfd,reply,msgLen);
                switch(reply->ch)
                {
                        case 1:
                        printf("First Value is %f \n",reply->input[0]);
                        printf("Second Value is %f \n",reply->input[1]);
                        printf("Addition is %f",addition((float)reply->input[0], (float)reply->input[1]));
                        fflush(stdout);
                        reply->output = addition((float)reply->input[0], (float)reply->input[1]) ;
```

VISHWAKARMA UNIVERSITY Maximising Human Potential

```
                        write(ccfd,reply,msgLen);
                        break;
                }

        }
}
float addition (float a, float b)
{
        float result;
        result = a + b   ;
        return result;
}
```

//TCPCLIENT PROGRAM

VISHWAKARMA UNIVERSITY Maximising Human Potential

```
#include<unistd.h>
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<string.h>
#include<stdlib.h>
#define MAX 30

typedef struct message
{
        float input[2];
        float output;
        int ch;
}message;

void main()
```

```
{
        int ssfd;
        int len;
        int msgLen;
        struct sockaddr_in address;
        int result;
        FILE *fp;
        char ch='A';
        message *msg;
        message *reply;
        msg=(message*)malloc(sizeof(message));
        reply=(message*)malloc(sizeof(message));
        msgLen=sizeof(message);
        ssfd=socket(AF_INET,SOCK_STREAM,0);
        address.sin_family=AF_INET;
        address.sin_addr.s_addr=inet_addr("127.0.0.1");
        address.sin_port=9734;
        len=sizeof(address);
        result=connect(ssfd,(struct sockaddr*)&address,len);
        if(result==-1){
        perror("Server not found");
        printf("\nError Result:%d",result);
}
printf("\nConnection successful");
do{
        printf("\n\nSELECT Appropriate Option:\n");
        printf("\n1.\tAddition");
        printf("\n2.\tDisconnect And Exit");
        scanf("%d",&msg->ch);
        printf("Your Choice:%d",msg->ch);
        switch(msg->ch)
        {
                case 1:
```

VISHWAKARMA UNIVERSITY Maximising Human Potential

```
                printf("\nEnter any First value:");
                scanf("%f",&msg->input[0]);
                printf("\nEnter any Second value:");
                scanf("%f",&msg->input[1]);
                write(ssfd,msg,msgLen);
                read(ssfd,reply,msgLen);
                printf("Output from server:%f",reply->output);
                break;

                default:
                printf("Invalid Choice....Select Another");
                break;
        }
}while(msg->ch!=2);
close(ssfd);
}
```

VISHWAKARMA UNIVERSITY
Maximising Human Potential

/*************************************

TCP SERVER TERMINAL

****

```
samarth@Ubuntu:~$ gcc server.c -o server -lpthread
samarth@Ubuntu:~$ ./server
Waiting for connection
 SSFD Value:3
Connected to socket
Connection accepted successfully
First Value is 12.300000
Second Value is 13.200000
Addition is 25.500000
First Value is 12.000000
Second Value is 13.000000
```

```
Addition is 25.000000
```

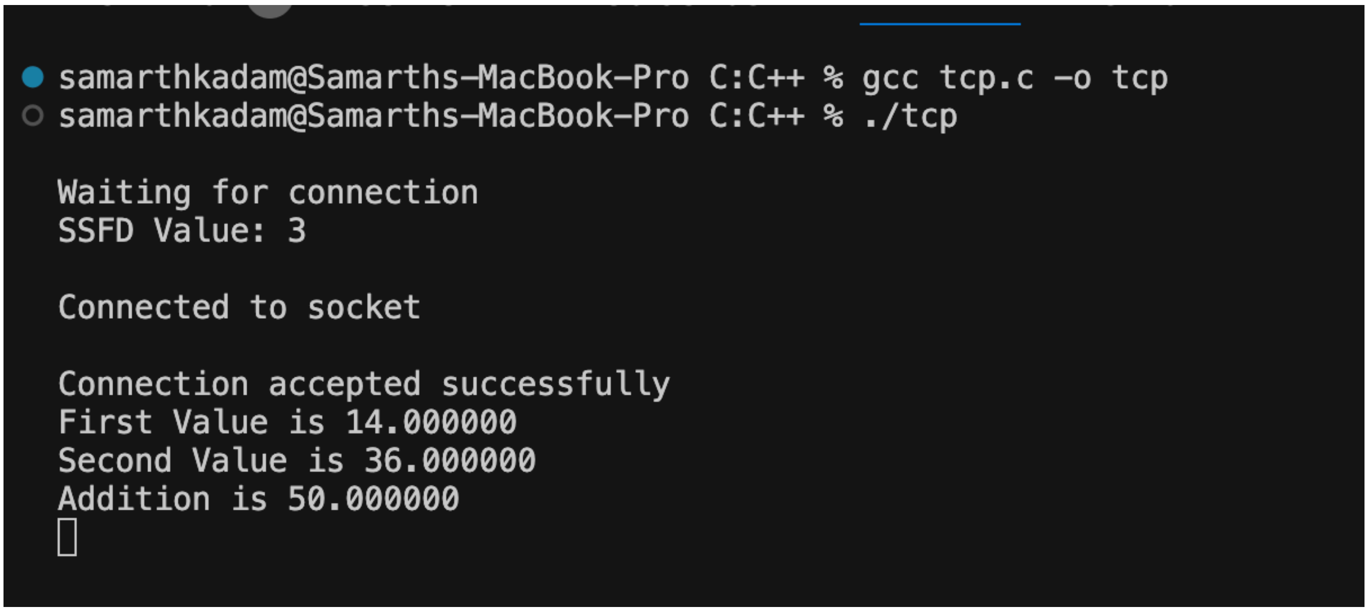
/*************************************

TCP CLIENT TERMINAL

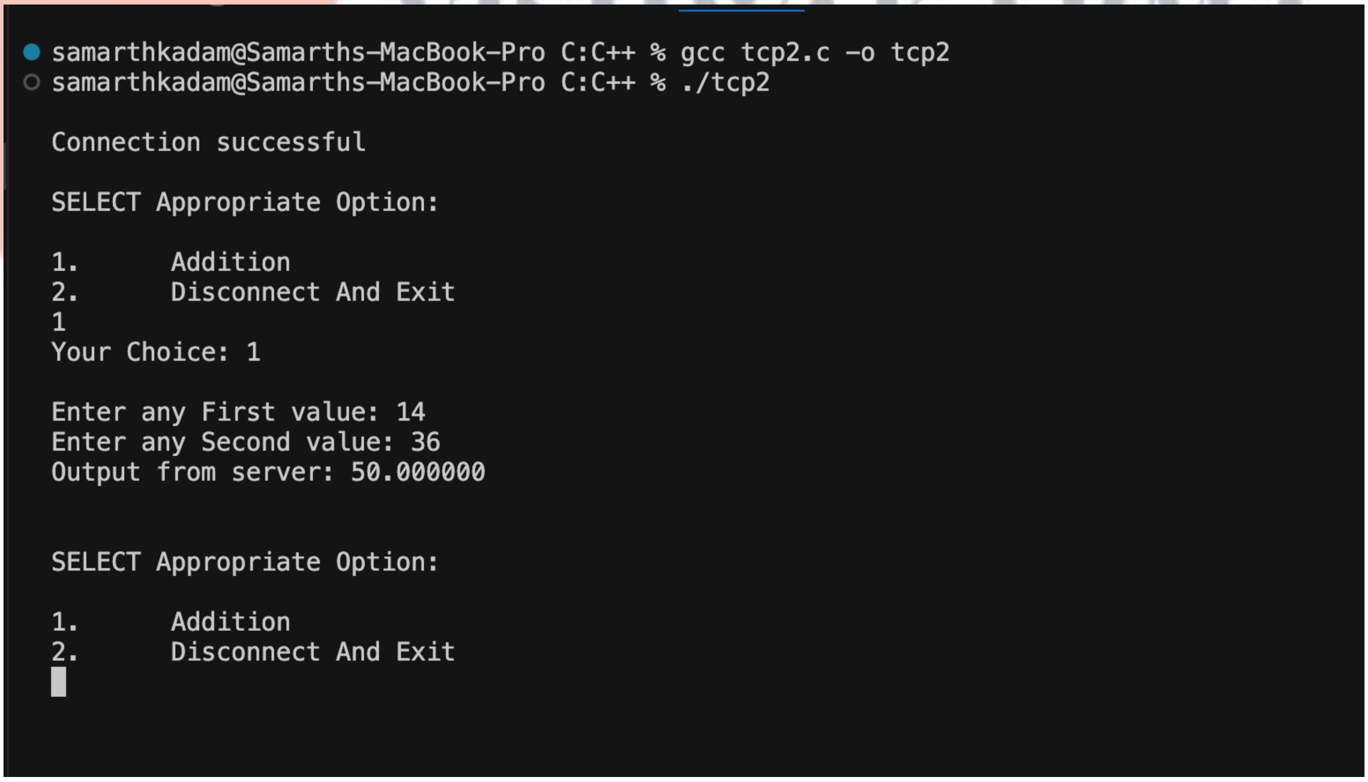
*************************************/

```
samarthkadam@Samarths-MacBook-Pro C:C++ % gcc tcp.c -o tcp
samarthkadam@Samarths-MacBook-Pro C:C++ % ./tcp

Waiting for connection
SSFD Value: 3

Connected to socket

Connection accepted successfully
First Value is 14.000000
Second Value is 36.000000
Addition is 50.000000

samarthkadam@Samarths-MacBook-Pro C:C++ % gcc tcp2.c -o tcp2
samarthkadam@Samarths-MacBook-Pro C:C++ % ./tcp2

Connection successful

SELECT Appropriate Option:

1.     Addition
2.     Disconnect And Exit
1
Your Choice: 1

Enter any First value: 14
Enter any Second value: 36
Output from server: 50.000000


SELECT Appropriate Option:

1.     Addition
2.     Disconnect And Exit
```

VISHWAKARMA UNIVERSITY Maximising Human Potential



```
● samarthkadam@Samarths-MacBook-Pro C:C++ % gcc tcp.c -o tcp
○ samarthkadam@Samarths-MacBook-Pro C:C++ % ./tcp

  Waiting for connection
  SSFD Value: 3

  Connected to socket

  Connection accepted successfully
  First Value is 14.000000
  Second Value is 36.000000
  Addition is 50.000000
```



```
● samarthkadam@Samarths-MacBook-Pro C:C++ % gcc tcp2.c -o tcp2
○ samarthkadam@Samarths-MacBook-Pro C:C++ % ./tcp2

  Connection successful

  SELECT Appropriate Option:

  1.      Addition
  2.      Disconnect And Exit
  1
  Your Choice: 1

  Enter any First value: 14
  Enter any Second value: 36
  Output from server: 50.000000


  SELECT Appropriate Option:

  1.      Addition
  2.      Disconnect And Exit
```

Conclusion :

In this assignment, we explored the implementation of TCP socket programming through a client-server model using C. We learned how sockets enable communication between two processes over a network by creating, binding, and listening on a socket for incoming connections. This practical exercise highlighted key functions such as accept, connect, send, and receive for managing data transmission. The program successfully demonstrated a simple client-server interaction where the client sends data for a mathematical operation, and the server processes and returns the result.

This experience provided a foundational understanding of network socket interactions and the client-server communication process over TCP.

VU VISHWAKARMA UNIVERSITY Maximising Human Potential