# MEGHNAD SAHA INSTITUTE OF TECHNOLOGY

*Nazirabad, P.O. - Uchhepota, Near URBANA Complex, Anandapur, Kolkata 700 150*

**BACHELOR OF COMPUTER APPLICATION**



LABORATORY NOTE BOOK

MAKAUT ODD SEMESTER 2023

PAPER NAME: COMPUTER NETWORKING LAB

PAPER CODE: BCAC592

STUDENT NAME: RUPAK SARKAR

ROLL NO: 31001221010

REGN. NO.: 213101001210001

YEAR: 3RD

SEMESTER: 5TH

SECTION: BCA

SESSION: 2021-2024



MEGHNAD SAHA INSTITUTE OF TECHNOLOGY

*Nazirabad, P.O. - Uchhepota, Near URBANA Complex, Anandapur, Kolkata 700 150*

## “LIST OF ASSIGNMENT/EXPERIMENT SUBMISSION DETAILS”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SL.**  **NO.** | **ASSIGNMENT / EXPERIMENT NAME** | **DATE OF ASSIGNMENT**  **/ EXPERIMENT DONE** | **DATE OF SUBMISSION** | **CHECKED BY** | **REMARKS**  **(ANY DEVIATION REGARDING SUBMISSION DATES, CONTENT, FORMAT, ETC)** |
| 1. | Establish a TCP connection between Server and Client. | 30/09/2023 | 30/09/2023 |  |  |
| 2. | Client program will send some echo messages to the server. | 30/09/2023 | 30/09/2023 |  |  |
| 3. | Server will return acknowledgement echo message to the client. | 30/09/2023 | 30/09/2023 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

OBSERVATIONS / COMMENTS ON THE OVERALL PERFORMANCE:

Signature in full with date

Faculty / Technical Assistant

Write and run a TCP client and a TCP server program using C / ‘JAVA’ language in UNIX /LINUX /Windows as per the following details:

I. Establish a TCP connection between a TCP client and a TCP server.

II. Client program will send some messages (echo client) to the server.

III. Server will return the messages/acknowledgement (echo server) to the client.

TCP refers to the Transmission Control Protocol. It is one of the main protocols used for communication over the internet. Some of the features of this protocol are:

* It is a connection-oriented communication protocol.
* It used a three-way handshake to establish reliable connections.
* TCP guarantees the delivery of the data packets.

A client-server architecture is a model in computer networking, where the server provides some service to the client. In this architecture, the client computer sends a request to the server computer through a network (internet). The server accepts this request and sends the required data to the client.

# Server-Side Code

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <unistd.h>  
#include <arpa/inet.h>  
   
int main(){  
   
 char \*ip = "127.0.0.1";  
 int port = 5566;  
   
 int server\_sock, client\_sock;  
 struct sockaddr\_in server\_addr, client\_addr;  
 socklen\_t addr\_size;  
 char buffer[1024];  
 int n;  
   
 server\_sock = socket(AF\_INET, SOCK\_STREAM, 0);  
 if (server\_sock < 0){  
 perror("[-]Socket error");  
 exit(1);  
 }  
 printf("[+]TCP server socket created.\n");  
   
 memset(&server\_addr, '\0', sizeof(server\_addr));  
 server\_addr.sin\_family = AF\_INET;  
 server\_addr.sin\_port = port;  
 server\_addr.sin\_addr.s\_addr = inet\_addr(ip);  
   
 n = bind(server\_sock, (struct sockaddr\*)&server\_addr, sizeof(server\_addr));  
 if (n < 0){  
 perror("[-]Bind error");  
 exit(1);  
 }  
 printf("[+]Bind to the port number: %d\n", port);  
   
 listen(server\_sock, 5);  
 printf("Listening...\n");  
   
 while(1){  
 addr\_size = sizeof(client\_addr);  
 client\_sock = accept(server\_sock, (struct sockaddr\*)&client\_addr, &addr\_size);  
 printf("[+]Client connected.\n");  
   
 bzero(buffer, 1024);  
 recv(client\_sock, buffer, sizeof(buffer), 0);  
 printf("Client: %s\n", buffer);  
   
 bzero(buffer, 1024);  
 strcpy(buffer, "HI, THIS IS SERVER. HAVE A NICE DAY!!!");  
 printf("Server: %s\n", buffer);  
 send(client\_sock, buffer, strlen(buffer), 0);  
   
 close(client\_sock);  
 printf("[+]Client disconnected.\n\n");  
   
 }  
 return 0;  
}

# Client-Side Code

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <unistd.h>  
#include <arpa/inet.h>  
   
int main(){  
   
 char \*ip = "127.0.0.1";  
 int port = 5566;  
   
 int sock;  
 struct sockaddr\_in addr;  
 socklen\_t addr\_size;  
 char buffer[1024];  
 int n;  
   
 sock = socket(AF\_INET, SOCK\_STREAM, 0);  
 if (sock < 0){  
 perror("[-]Socket error");  
 exit(1);  
 }  
 printf("[+]TCP server socket created.\n");  
   
 memset(&addr, '\0', sizeof(addr));  
 addr.sin\_family = AF\_INET;  
 addr.sin\_port = port;  
 addr.sin\_addr.s\_addr = inet\_addr(ip);  
   
 connect(sock, (struct sockaddr\*)&addr, sizeof(addr));  
 printf("Connected to the server.\n");  
   
 bzero(buffer, 1024);  
 strcpy(buffer, "HELLO, THIS IS CLIENT.");  
 printf("Client: %s\n", buffer);  
 send(sock, buffer, strlen(buffer), 0);  
   
 bzero(buffer, 1024);  
 recv(sock, buffer, sizeof(buffer), 0);  
 printf("Server: %s\n", buffer);  
   
 close(sock);  
 printf("Disconnected from the server.\n");  
   
 return 0;  
}

A socket is one endpoint of a two-way communication link between two programs running on the network. The socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent. In java socket programming example tutorial, we will learn how to write **java socket server** and **java socket client** program. We will also learn how server client program read and write data on the socket. **java.net.Socket** and **java.net.ServerSocket** are the java classes that implements Socket and Socket server.

package com.journaldev.socket;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.InetAddress;

import java.net.Socket;

import java.net.UnknownHostException;

/\*\*

\* This class implements java socket client

\* @author pankaj

\*

\*/

public class SocketClientExample {

public static void main(String[] args) throws UnknownHostException, IOException, ClassNotFoundException, InterruptedException{

//get the localhost IP address, if server is running on some other IP, you need to use that

InetAddress host = InetAddress.getLocalHost();

Socket socket = null;

ObjectOutputStream oos = null;

ObjectInputStream ois = null;

for(int i=0; i<5;i++){

//establish socket connection to server

socket = new Socket(host.getHostName(), 9876);

//write to socket using ObjectOutputStream

oos = new ObjectOutputStream(socket.getOutputStream());

System.out.println("Sending request to Socket Server");

if(i==4)oos.writeObject("exit");

else oos.writeObject(""+i);

//read the server response message

ois = new ObjectInputStream(socket.getInputStream());

String message = (String) ois.readObject();

System.out.println("Message: " + message);

//close resources

ois.close();

oos.close();

Thread.sleep(100);

}

}

}

To test java socket programming of server-client communication, first we need to run SocketServerExample class. When you will run socket server, it will just print “Waiting for client request” and then wait for the client request. Now when you will run SocketClientExample class, it will send a request to java socket server and print the response message to console. Here is the output of java socket server SocketServerExample program.

Waiting for the client request

Message Received: 0

Waiting for the client request

Message Received: 1

Waiting for the client request

Message Received: 2

Waiting for the client request

Message Received: 3

Waiting for the client request

Message Received: exit

Shutting down Socket server!!

Sending request to Socket Server

Message: Hi Client 0

Sending request to Socket Server

Message: Hi Client 1

Sending request to Socket Server

Message: Hi Client 2

Sending request to Socket Server

Message: Hi Client 3

Sending request to Socket Server

Message: Hi Client exit