

Exercise 5: UDP

Aumrudh Lal Kumar TJ,18BIT034

BTech IT 3rd Year, 5th Sem

1)

Problem: Write a program for reversing given string without inbuilt functions using client, server in UDP.

Aim: To write a program for reversing given string without inbuilt functions using client, server in UDP in C and Java.

Program:

C

Server

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

```
#include<netdb.h>
```

```
#include<arpa/inet.h>
```

```
#include<netinet/in.h>
```

```
#include<stdlib.h>
```

```
#include<string.h>
```

```
#include<sys/socket.h>
```

```
#include<sys/types.h>
```

```
#include<unistd.h>
```

```
int main(){
```

```
    int sd,b,len;
```

```
    char msg[100];
```

```
    struct sockaddr_in server,client;
```

```

server.sin_family=AF_INET;

printf("Enter the port no : ");

int portno;

scanf("%d",&portno);

server.sin_port=htons(portno);

server.sin_addr.s_addr=htonl(INADDR_ANY);


//socket creation

sd=socket(AF_INET,SOCK_DGRAM,0);

if(sd==-1){

    printf("Socket creation failed\n");

    exit(0);

}

else{

    printf("Socket Created\n");

}


//binding

b=bind(sd,(struct sockaddr *)&server,sizeof(server));

if(b==-1){

    printf("Binding failed\n");

    exit(0);

}

else{

    printf("Binded\n");

}

```

```

len=sizeof(client);
do{
    recvfrom(sd,msg,100,0,(struct sockaddr*)&client,&len);
    printf("Client message : %s\n",msg);
    int length=0,i,j=0;
    char temp[100];
    for (i=0;msg[i]!='\0';i++){
        length++;
    }
    printf("Length : %d\n",length);
    for(i=length-1;i>=0;i--){
        temp[j++]=msg[i];
    }
    temp[j]='\0';
    printf("Reversed Word : %s\n",temp);
    sendto(sd,temp,100,0,(struct sockaddr*)&client,sizeof(client));
}while(strcmp(msg,"bye")!=0);
close(sd);
}

```

Client

```

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

```

```
#include<unistd.h>
```

```
int main(){  
    int sd,b,len;  
    char msg[100];  
    struct sockaddr_in server,client;  
    server.sin_family=AF_INET;  
    printf("Enter the port no : ");  
    int portno;  
    scanf("%d",&portno);  
    server.sin_port=htons(portno);  
    server.sin_addr.s_addr=htonl(INADDR_ANY);  
  
    //socket creation  
  
    sd=socket(AF_INET,SOCK_DGRAM,0);  
    if(sd==-1){  
        printf("Socket creation failed\n");  
        exit(0);  
    }  
    else{  
        printf("Socket Created\n");  
    }  
    len=sizeof(server);  
    do{  
        printf("Enter message : ");  
        scanf("%s",msg);  
        sendto(sd,msg,100,0,(struct sockaddr*)&server,sizeof(server));  
        recvfrom(sd,msg,100,0,(struct sockaddr*)&server,&len);  
    }  
}
```

```

printf("Server message : %s\n",msg);

}while(strcmp(msg,"bye")!=0);

close(sd);
}

```

Output

The screenshot shows two terminal windows on a Windows desktop. The left window is the server program, and the right window is the client program. Both are compiled with GCC and executed. The server is listening on port 8953 and has successfully bound to it. The client is also listening on port 8953 and has successfully bound to it.

```

aamrudh@aamrudhlalKumarTJ: ~/cnp/ex5
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ cc server_reverse.c
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ ./a.out
Enter the port no : 8953
Socket Created
bindind

```

```

aamrudh@aamrudhlalKumarTJ: ~/cnp/ex5
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ cc client_reverse.c
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ ./a.out
Enter the port no : 8953
Socket Created
Enter message :

```

The screenshot shows the same two terminal windows as before, but now they are interacting. The client sends messages to the server, and the server responds with the reversed words. The messages and their reversed versions are shown in the output.

```

aamrudh@aamrudhlalKumarTJ: ~/cnp/ex5
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ cc server_reverse.c
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ ./a.out
Enter the port no : 8953
Socket Created
bindind
Client message : great
Reversed Word : taerg
Client message : do
Reversed Word : od
Client message : computer
Reversed Word : retupmoc
Client message : mom
Reversed Word : mom
Client message : bye
Reversed Word : eyb
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$

```

```

aamrudh@aamrudhlalKumarTJ: ~/cnp/ex5
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ cc client_reverse.c
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$ ./a.out
Enter the port no : 8953
Socket Created
Enter message : great
Server message : taerg
Enter message : do
Server message : od
Enter message : computer
Server message : retupmoc
Enter message : mom
Server message : mom
Enter message : bye
Server message : eyb
aamrudh@aamrudhlalKumarTJ:~/cnp/ex5$

```

Java

Server

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.lang.*;

public class Server{
    // Server UDP socket runs at this port

    public static void main(String[] args) throws IOException{
        System.out.print("Enter port no : ");
        Scanner in=new Scanner(System.in);
        int port=in.nextInt();
        try{
            // Instantiate a new DatagramSocket to receive responses from the client
            DatagramSocket sd = new DatagramSocket(port);
            System.out.println("Waiting for a client to connect...");
            /* Create buffers to hold sending and receiving data.
            It temporarily stores data in case of communication delays */
            String receivedData;
            while(true){
                byte[] receivingDataBuffer = new byte[1000];
                //byte[] sendingDataBuffer = new byte[1000];

                /* Instantiate a UDP packet to store the
                client data using the buffer for receiving data*/
```

```
DatagramPacket inputPacket = new DatagramPacket(receivingDataBuffer,  
receivingDataBuffer.length);
```

```
// Receive data from the client and store in inputPacket  
sd.receive(inputPacket);
```

```
// Printing out the client sent data  
receivedData = new String(inputPacket.getData());  
receivedData=receivedData.trim();  
System.out.println("Client Message : "+receivedData);  
if(receivedData.equals("bye")){  
    System.out.println("Closing Connection");  
    sd.close();  
    System.exit(0);  
}
```

```
int length=0;  
String temp="";  
for(char c: receivedData.toCharArray()) {  
    length++;  
}
```

```
//System.out.println("Length : "+length);  
for(int i=length-1;i>=0;i--){  
    temp= temp+ receivedData.charAt(i);  
}
```

```
//System.out.println(temp);
```

```
byte[] sendingDataBuffer = new byte[length];
```

```
/*Convert client sent data string reverse,Convert it to bytes
```

```

        and store it in the corresponding buffer. */
        sendingDataBuffer = temp.getBytes();

        // Obtain client's IP address and the port
        InetAddress senderAddress = inputPacket.getAddress();
        int senderPort = inputPacket.getPort();

        // Create new UDP packet with data to send to the client
        DatagramPacket outputPacket = new DatagramPacket(
            sendingDataBuffer, sendingDataBuffer.length,
            senderAddress, senderPort
        );
        // Send the created packet to client
        sd.send(outputPacket);
    }
}
catch (Exception e){
    System.out.println(e);
}
}
}

```

Client

```

import java.io.*;
import java.net.*;
import java.util.*;
import java.lang.*;

```



```

public class Client{

    public static void main(String[] args) throws IOException{

        try{

            System.out.print("Enter port no : ");

            Scanner in=new Scanner(System.in);

            int port=in.nextInt();

            DatagramSocket sd = new DatagramSocket();


            // Get the IP address of the server

            InetAddress IPAddress = InetAddress.getByName("localhost");


            // Creating corresponding buffers

            String sentence;

            while(true){

                //byte[] sendingDataBuffer = new byte[1000];

                //byte[] receivingDataBuffer = new byte[1000];


                /* Converting data to bytes and
                storing them in the sending buffer */

                System.out.print("Enter message : ");

                sentence = in.next();


                int length=0;

                for(char c: sentence.toCharArray()) {

                    length++;

                }

                byte[] sendingDataBuffer = new byte[length];

                byte[] receivingDataBuffer = new byte[length];

                sendingDataBuffer = sentence.getBytes();

```

```

// Creating a UDP packet

DatagramPacket sendingPacket = new
DatagramPacket(sendingDataBuffer,sendingDataBuffer.length,IPAddress, port);

// sending UDP packet to the server
sd.send(sendingPacket);

if(sentence.equals("bye")){
    System.out.println("Closing Connection");
    sd.close();
    System.exit(0);
}

// Get the server response .i.e. capitalized sentence

DatagramPacket receivingPacket = new
DatagramPacket(receivingDataBuffer,receivingDataBuffer.length);

sd.receive(receivingPacket);

// Printing the received data

String receivedData = new String(receivingPacket.getData());
receivedData.trim();

System.out.println("Sent from the server: "+receivedData);

}

}

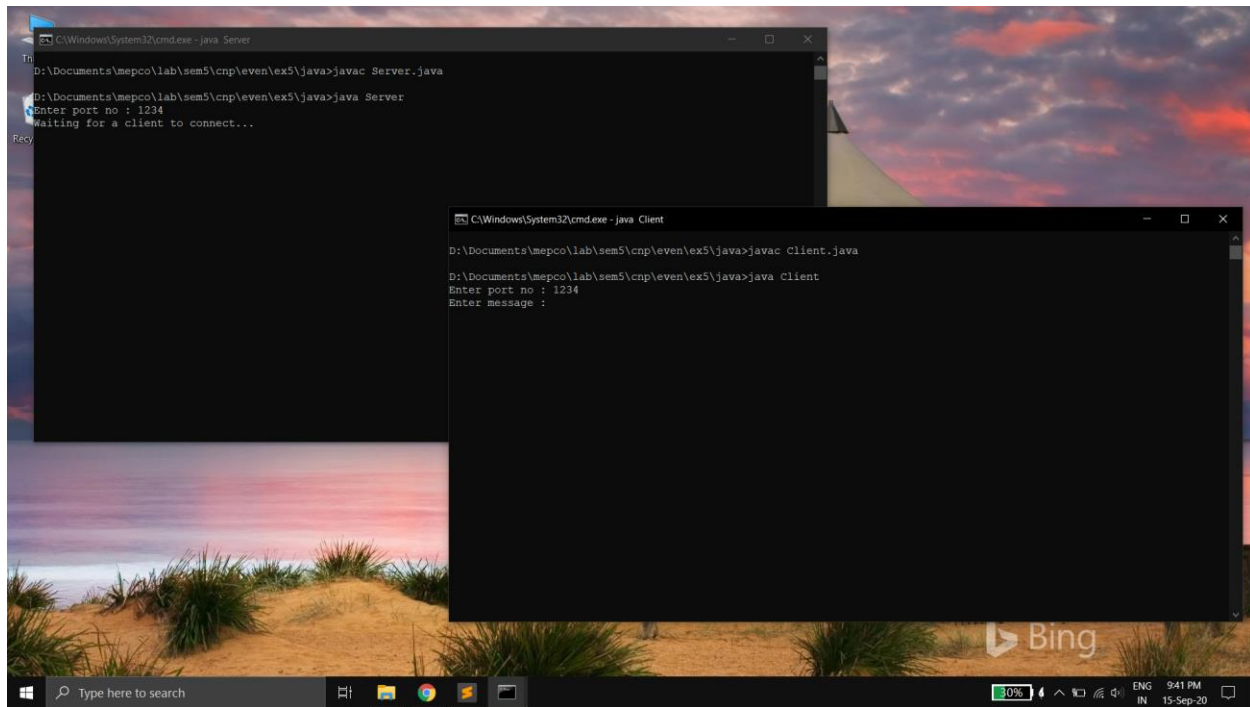
catch(Exception e) {
    System.out.println(e);
}

}

}

```

Output



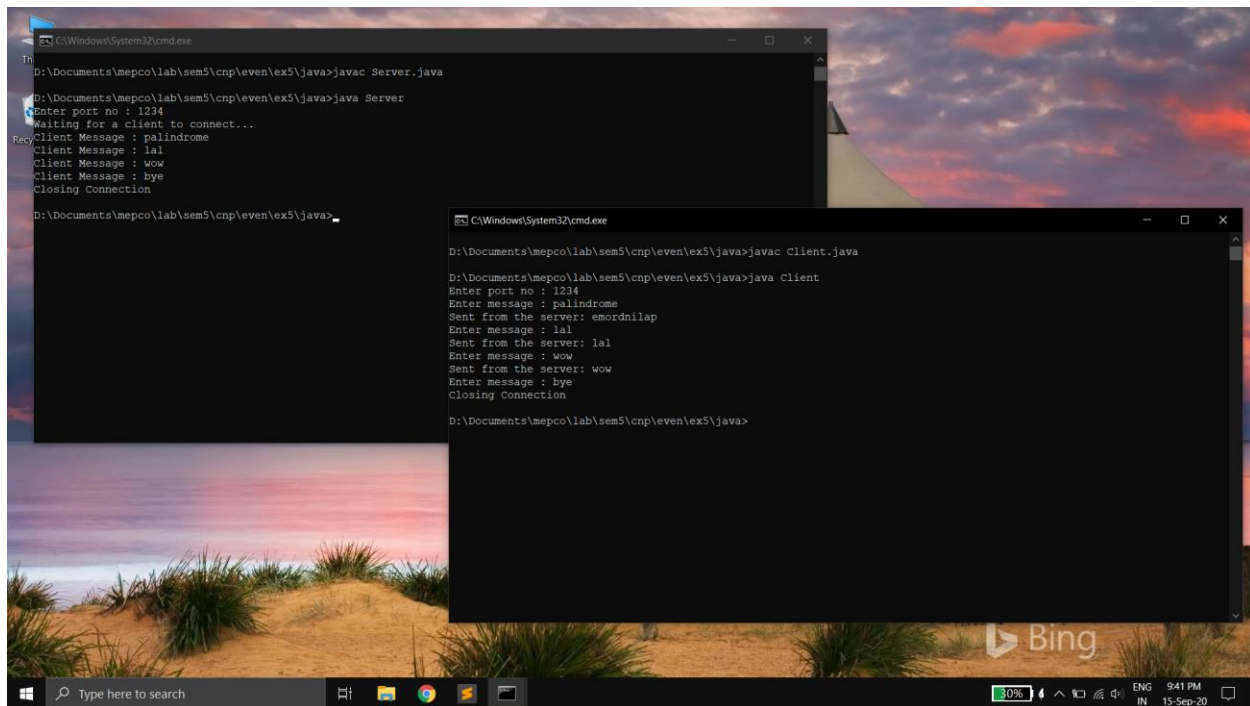
The screenshot shows two overlapping Windows command prompt windows. The background is a desktop wallpaper of a beach at sunset. The taskbar at the bottom shows the search bar, task view, and several application icons. The system tray on the right shows the battery level at 30%, network status, and the date and time as 9:41 PM on 15-Sep-20.

Left Window (C:\Windows\System32\cmd.exe - java Server):

```
Tr
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>javac Server.java
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>java Server
Enter port no : 1234
Waiting for a client to connect...
```

Right Window (C:\Windows\System32\cmd.exe - java Client):

```
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>javac Client.java
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>java Client
Enter port no : 1234
Enter message :
```



The screenshot shows the same two overlapping Windows command prompt windows as above, but now they are displaying the execution of the Java program with messages.

Left Window (C:\Windows\System32\cmd.exe - java Server):

```
Tr
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>javac Server.java
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>java Server
Enter port no : 1234
Waiting for a client to connect...
Recy Client Message : palindrome
Client Message : lol
Client Message : wow
Client Message : bye
Closing Connection
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>
```

Right Window (C:\Windows\System32\cmd.exe - java Client):

```
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>javac Client.java
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>java Client
Enter port no : 1234
Enter message : palindrome
Sent from the server: emordnilap
Enter message : lol
Sent from the server: lol
Enter message : wow
Sent from the server: wow
Enter message : bye
Closing Connection
D:\Documents\mepco\lab\sem5\cnp\even\ex5\java>
```

