**C] Aim: Implement a program to return the reverse of the string using socket**

**Source Code:**

**ReverseClient1.java**

package com.mycompany.reverseclient1;

import java.net.\*;

import java.io.\*;

public class ReverseClient1 {

Socket soc;

BufferedReader br,br1;

PrintWriter out;

String str;

public ReverseClient1(){

try{

soc=new Socket("127.0.0.1",8765);

br=new BufferedReader(new InputStreamReader(System.in));

br1=new BufferedReader(new InputStreamReader(soc.getInputStream()));

out=new PrintWriter(soc.getOutputStream());

while(true){

System.out.println("Enter the message: ");

str=br.readLine();

out.println(str);

out.flush();

System.out.println("Message from server: ");

str=br1.readLine();

System.out.println(str);

if(str.equals("q"))

break;}

soc.close();}

catch (Exception e){}}

public static void main(String[] args) {

new ReverseClient1();}}

**ReverseServer1.java**

package com.mycompany.reverseclient1;

import java.net.\*;

import java.io.\*;

public class ReverseServer1 {

ServerSocket ss;

Socket soc;

BufferedReader br,br1;

PrintWriter out;

String str;

public ReverseServer1(){

try{

ss=new ServerSocket(8765); System.out.println("Server is listening to port 8765");

soc=ss.accept();

System.out.println("Connection established!!");

br=new BufferedReader(new InputStreamReader(System.in));

br1=new BufferedReader(new InputStreamReader(soc.getInputStream()));

out=new PrintWriter(soc.getOutputStream());

while(true){

System.out.println("Message from client");

str=br1.readLine();

int k=str.length();

System.out.println(str);

String reverse="";

for(int i=k-1;i>=0;i--)

{ reverse=reverse+str.charAt(i);}

System.out.println("Reverse of the string is: "+reverse);

out.println(reverse);

out.flush();

if(str.equals("q"))

break;}

soc.close();}

catch (Exception e){}}

public static void main(String[] args) {

new ReverseServer1(); }}

‘**Practical No:2**

**Remote Procedure Call**

**A] Aim: Implement a Server calculator Add(), Mul(), Sub(), Div() using datagram**

**Source Code:**

**RPCCalClient.java**

package com.mycompany.rpccalclient;

import java.net.\*;

import java.io.\*;

public class RPCCalClient {

RPCCalClient(){

try{

InetAddress ia=InetAddress.getLocalHost();

DatagramSocket ds=new DatagramSocket();

DatagramSocket ds1=new DatagramSocket(1300);

System.out.println("\nRPC Client.\n");

System.out.println("Enter method name and parameter like add 3 4:");

while(true){

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

String str=br.readLine();

byte b[]=str.getBytes();

DatagramPacket dp=new DatagramPacket(b,b.length,ia,1200);

ds.send(dp);

dp=new DatagramPacket(b,b.length);

ds1.receive(dp);

String s=new String(dp.getData(),0,dp.getLength());

System.out.println("\nResult="+s+"\n"); } }

catch(Exception e){}}

public static void main(String[] args) {

new RPCCalClient();}}

**RPCCalServer.java**

package com.mycompany.rpccalserver;

import java.net.\*;

import java.io.\*;

import java.util.\*;

public class RPCCalServer {

DatagramPacket dp;

DatagramSocket ds;

String str,methodName,result;

int val1,val2;

RPCCalServer(){

try{

ds=new DatagramSocket(1200);

byte b[]=new byte[4096];

System.out.println("Server started");

while(true){

dp=new DatagramPacket(b,b.length);

ds.receive(dp);

str=new String(dp.getData(),0,dp.getLength());

if(str.equalsIgnoreCase("q")){

System.exit(1);}

else{

StringTokenizer st=new StringTokenizer(str," ");

int i=0;

while(st.hasMoreElements()){

String token=st.nextToken();

methodName=token;

val1=Integer.parseInt(st.nextToken());

val2=Integer.parseInt(st.nextToken());

} }

System.out.println(str);

if(methodName.equalsIgnoreCase("add")){

result=""+add(val1,val2);}

else if(methodName.equalsIgnoreCase("sub")){

result=""+sub(val1,val2);}

else if(methodName.equalsIgnoreCase("mul")){

result=""+mul(val1,val2);}

else if(methodName.equalsIgnoreCase("div")){

result=""+div(val1,val2);}

else{

System.out.println("Enter a valid operation");}

byte b1[]=result.getBytes();

DatagramSocket ds1=new DatagramSocket();

DatagramPacket dp1=new DatagramPacket(b1,b1.length,InetAddress.getLocalHost(),1300);

System.out.println("Result: "+result+"\n");

ds1.send(dp1);} }

catch(Exception e){} }

public int add(int val1,int val2){

return val1+val2;}

public int sub(int val1,int val2){

return val1-val2;}

public int mul(int val1,int val2){

return val1\*val2;}

public int div(int val1,int val2){

return val1/val2;}

public static void main(String[] args) {

new RPCCalServer(); }}

**B] Aim:Implement a Date Time Server containing date() and time() using datagram**

**Source Code:**

**DateTimeClient.java**

package com.mycompany.datetimeclient;

import java.net.\*;

import java.io.\*;

public class DateTimeClient {

DateTimeClient(){

try{

InetAddress ia=InetAddress.getLocalHost();

DatagramSocket ds=new DatagramSocket();

DatagramSocket ds1=new DatagramSocket(1300);

System.out.println("\nDate Time Client\n");

byte b1[]=new byte[1000]; while(true)

{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

String str=br.readLine();

byte b[]=str.getBytes();

DatagramPacket dp=new DatagramPacket(b,b.length,ia,1200);

ds.send(dp);

dp=new DatagramPacket(b1,b1.length);

ds1.receive(dp);

String s=new String(dp.getData(),0,dp.getLength()); System.out.println("\nResult="+s+"\n");

}}

catch(Exception e){}}

public static void main(String[] args) {

new DateTimeClient();}}

**DateTimeServer.java**

package com.mycompany.datetimeserver;

import java.net.\*;

import java.io.\*;

import java.util.\*;

import java.text.SimpleDateFormat;

public class DateTimeServer {

DatagramPacket dp;

DatagramSocket ds;

String str,methodName,result;

DateTimeServer() {

try{

ds=new DatagramSocket(1200);

byte b[]=new byte[4096];

System.out.println("\n Date Time Server \n");

while(true)

{

dp=new DatagramPacket(b,b.length);

ds.receive(dp);

str=new String(dp.getData(),0,dp.getLength());

if(str.equalsIgnoreCase("q"))

System.exit(1);

else

{

StringTokenizer st=new StringTokenizer(str," ");

int i=0;

while(st.hasMoreTokens())

{

String token=st.nextToken();

methodName=token;}}

Calendar c=Calendar.getInstance();

SimpleDateFormat dateFormat=new SimpleDateFormat("MM/dd/yyyy");

Date d=c.getTime();

InetAddress ia=InetAddress.getLocalHost();

if(methodName.equalsIgnoreCase("date"))

result=""+dateFormat.format(d);

else if(methodName.equalsIgnoreCase("time"))

{

result=""+d.getHours()+":"+d.getMinutes()+":"+d.getSeconds();

}

byte b1[]=result.getBytes();

DatagramSocket ds1=new DatagramSocket();

DatagramPacket dp1=new DatagramPacket(b1,b1.length,ia,1300); System.out.println("result:"+result+"\n");

ds1.send(dp1);

}}

catch(Exception e){}}

public static void main(String[] args) {

new DateTimeServer();}}