**Practical No. 1**

**Aim:-** To implement LPC(Local Procedure Call) and RPC(Remote Procedure Call)

**Introduction:-**

1. **LPC (Local Procedure Call)**

The Local Procedure Call is an internal, [inter-process communication](https://en.wikipedia.org/wiki/Inter-process_communication) facility provided by the [Microsoft](https://en.wikipedia.org/wiki/Microsoft) [Windows NT](https://en.wikipedia.org/wiki/Windows_NT) [kernel](https://en.wikipedia.org/wiki/Kernel_(computer_science)) for lightweight [IPC](https://en.wikipedia.org/wiki/Inter-process_communication) (Inter Process Communication) between [processes](https://en.wikipedia.org/wiki/Process_(computing)) on the same computer.

1. **RPC (Remote Procedure Call)**

Remote Procedure Call is an Inter Process Communication technology that allows a computer programme to cause a subroutine or procedure to execute in another address space (generally on another computer on a shared network) without the programmer explicitly coding the details for this remote interaction. This is a form of [client–server](https://en.wikipedia.org/wiki/Client%E2%80%93server_model) interaction (caller is client, executor is server), typically implemented via a [request–response](https://en.wikipedia.org/wiki/Request%E2%80%93response) message-passing system.

Differences:

* RPC is slower than LPC since it uses the network to invoke the method.
* With RPC the procedure call can be executed on a remote machine which can be addressed in several ways.
* The parameters and return value need to be serializable (to use java terminology.
* RPC's can fail due to network issues.
* RPC's need to be set up before using them.
* The language used to call the remote procedure and the language implementing the remote procedure are not necessarily the same.

**Implementation:-**

Following steps can be followed to implement a simple RPC where in the client sends a number to a server to calculate it’s factorial:-

1. Write the client and server .java files.
2. Place the client file on the client system and the server file on the server system.
3. Note the server’s IP address and enter the same in the client.java file to facilitate remote connection to the server.
4. Compile both of the files.
5. Run the server.java file on server system first, and then run the client.java file on the client system. Make sure both of the systems are connected to the internet.
6. Now the user can enter a number on the client system which is then sent to the server system for calculating its factorial and the result is sent back. When the number is received at the server system, it’s passed to the method that calculates the factorial and which, in this case is declared in Factorial.java.
7. The server passes the number to the method, thus calling it on the client’s demand (remote calling) and sends the result.

**Program:-**

1. **Server.java**

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.io.OutputStream;

import java.io.OutputStreamWriter;

import java.net.ServerSocket;

import java.net.Socket;

public class Server

{

private static Socket socket;

public static void main(String[] args)

{ Factorial f = new Factorial();

int fact=1;

try

{

int port = 25000;

ServerSocket serverSocket = new ServerSocket(port);

System.out.println("Server Started and listening to the port 25000");

//Server is running always. This is done using this while(true) loop

while(true)

{

//Reading the message from the client

socket = serverSocket.accept();

InputStream is = socket.getInputStream();

InputStreamReader isr = new InputStreamReader(is);

BufferedReader br = new BufferedReader(isr);

String number = br.readLine();

System.out.println("Number received from client is "+number);

fact=f.factorial(Integer.parseInt(number));

//forming the return message

String returnMessage;

try

{

returnMessage = String.valueOf(fact) + "\n";

}

catch(NumberFormatException e)

{

//Input was not a number. Sending proper message back to client.

returnMessage = "Please send a proper number\n";

}

//Sending the response back to the client.

OutputStream os = socket.getOutputStream();

OutputStreamWriter osw = new OutputStreamWriter(os);

BufferedWriter bw = new BufferedWriter(osw);

bw.write(returnMessage);

System.out.println("Factorial sent to the client is "+returnMessage);

bw.flush();

}

}

catch (Exception e)

{

e.printStackTrace();

}

finally

{

try

{

socket.close();

}

catch(Exception e){}

}

}

}

1. **Client.java**

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.io.OutputStream;

import java.io.OutputStreamWriter;

import java.net.InetAddress;

import java.net.Socket;

public class Client

{

private static Socket socket;

public static void main(String args[])

{

try

{

String host = "192.168.1.9";

int port = 25000;

InetAddress address = InetAddress.getByName(host);

socket = new Socket(address, port);

//Send the message to the server

OutputStream os = socket.getOutputStream();

OutputStreamWriter osw = new OutputStreamWriter(os);

BufferedWriter bw = new BufferedWriter(osw);

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

System.out.println("Enter a number to calculate it's factorial: ");

String number = brr.readLine();

String sendMessage = number + "\n";

bw.write(sendMessage);

bw.flush();

System.out.println("Number sent to the server : "+sendMessage);

//Get the return message from the server

InputStream is = socket.getInputStream();

InputStreamReader isr = new InputStreamReader(is);

BufferedReader br = new BufferedReader(isr);

String message = br.readLine();

System.out.println("Factorial received from the server : " +message);

}

catch (Exception exception)

{

exception.printStackTrace();

}

finally

{

//Closing the socket

try

{

socket.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

}

1. **Factorial.java //called from Server.java**

public class Factorial

{

int factorial(int num)

{

int i,fact=1;

for(i=1;i<=num;i++)

fact=fact\*i;

return fact;

}

}