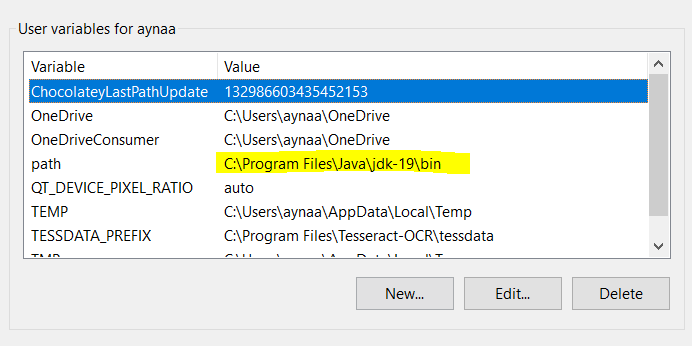
SA Practical

# Things to Download

Java jdk :<https://www.oracle.com/in/java/technologies/downloads/#jdk19-windows>(download x64 MSI Installer version) Set path environment variable for java

Also install Eclipse IDE for Enterprise Java and Web Developers in Eclipse Ide options.



Java Eclipse IDE :<https://www.eclipse.org/downloads/>

Tomcat:<https://tomcat.apache.org/download-80.cgi>( Go to core and under it download zip version 8.5)

# RMI Exp 4 code

**IHello.java**

import java.rmi.\*;

public interface IHello extends Remote{

public String message() throws RemoteException;

}

**HelloImpl.java**

import java.rmi.\*;

import java.rmi.server.\*;

public class HelloImpl extends UnicastRemoteObject implements IHello{

public HelloImpl() throws RemoteException {

//There is no action need in this moment.

}

public String message() throws RemoteException { return ("Hello");

}

}

**HelloServer.java**

import java.rmi.\*;

public class HelloServer {

private static final String host = "localhost";

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

//\*\* Step 1

//\*\* Declare a reference for the object that will be implemented HelloImpl temp = new HelloImpl();

//\*\* Step 2

//\*\* Declare a string variable for holding the URL of the object's name String rmiObjectName = "rmi://" + host + "/Hello";

//Step 3

//Binding the object reference to the object name. Naming.rebind(rmiObjectName, temp);

//Step 4

//Tell to the user that the process is completed. System.out.println("Binding complete...\n");

}

}

**HelloClient.java**

import java.rmi.ConnectException;

import java.rmi.Naming;

public class HelloClient

{

private static final String host = "localhost";

public static void main(String[] args)

{

try

{

//We obtain a reference to the object from the registry and next,

//it will be typecasted into the most appropiate type. IHello greeting\_message = (IHello) Naming.lookup("rmi://"

+ host + "/Hello");

//Next, we will use the above reference to invoke the remote

//object method. System.out.println("Message received: " +

greeting\_message.message());

}

catch (ConnectException conEx)

{

System.out.println("Unable to connect to server!"); System.exit(1);

}

catch (Exception ex)

{

ex.printStackTrace(); System.exit(1);

}

}

}

# RMI Exp 5 code

**Server.java**

public class Server implements interfaceCalculator{ public int add(int a,int b){

return a+b;

}

public int sub(int a,int b){ return a-b;

}

}

**InterfaceCalculator.java**

package middleware;

public interface interfaceCalculator{ public int add(int a,int b); public int sub(int a,int b);

}

**Client.java**

package middleware; public class Client {

public static void main(String [] args)

{

interfaceCalculator i=new Server(); System.out.println(i.add(12,13)); System.out.println(i.sub(12,12));

}

}

# Wrapper Exp 6

**Sender.java**

import java.net.\*; import java.util.\*; public class Sender {

public static void main(String[] args) throws Exception{ Scanner scn=new Scanner(System.in); System.out.println("Enter your Message:");

String str=scn.nextLine(); DatagramSocket ds=new DatagramSocket();

InetAddress ip=InetAddress.getByName("127.0.0.1"); DatagramPacket dp=new DatagramPacket(str.getBytes(),str.length(),ip,3000); ds.send(dp);

ds.close();

System.out.println("Message has been sent to Receiver Class Please Check:"+str);

}

}

**Receiver.java**

import java.net.\*; public class Receiver {

public static void main(String[] args) throws Exception{ System.out.println("Waiting for sender to send message"); DatagramSocket ds=new DatagramSocket(3000);

byte[] buf=new byte[1024];

DatagramPacket dp=new DatagramPacket(buf,1024); ds.receive(dp);

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

ds.close();

System.out.println("Message received Successfully..");

}

}