ASSIGNMENT-1B

Aim and Objective:

To develop any distributed application through implementing client-server communication programs based on Java RMI .

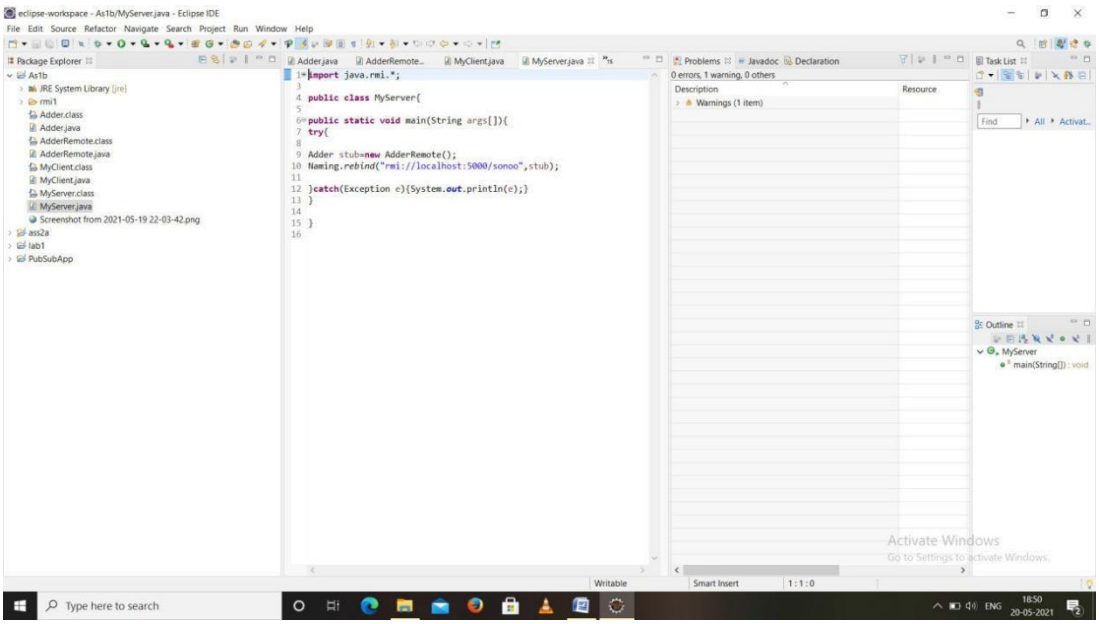
Tools / Environment:

Java Programming Environment, jdk 1.8, rmiregistry

CODES

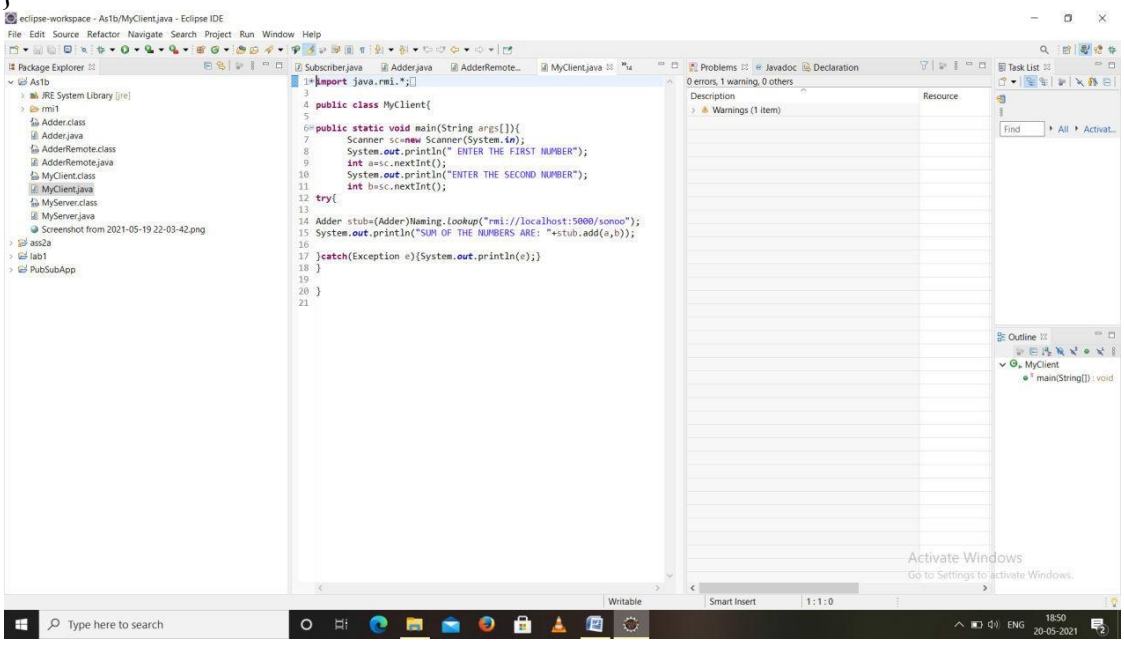
MyServer.java

```
import java.rmi.*;
import java.rmi.registry.*;

public class MyServer{

public static void main(String
args[]){  try{

Adder stub=new AdderRemote();
Naming.rebind("rmi://localhost:5000/sonoo",stub);

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

}
```



MyClient.java

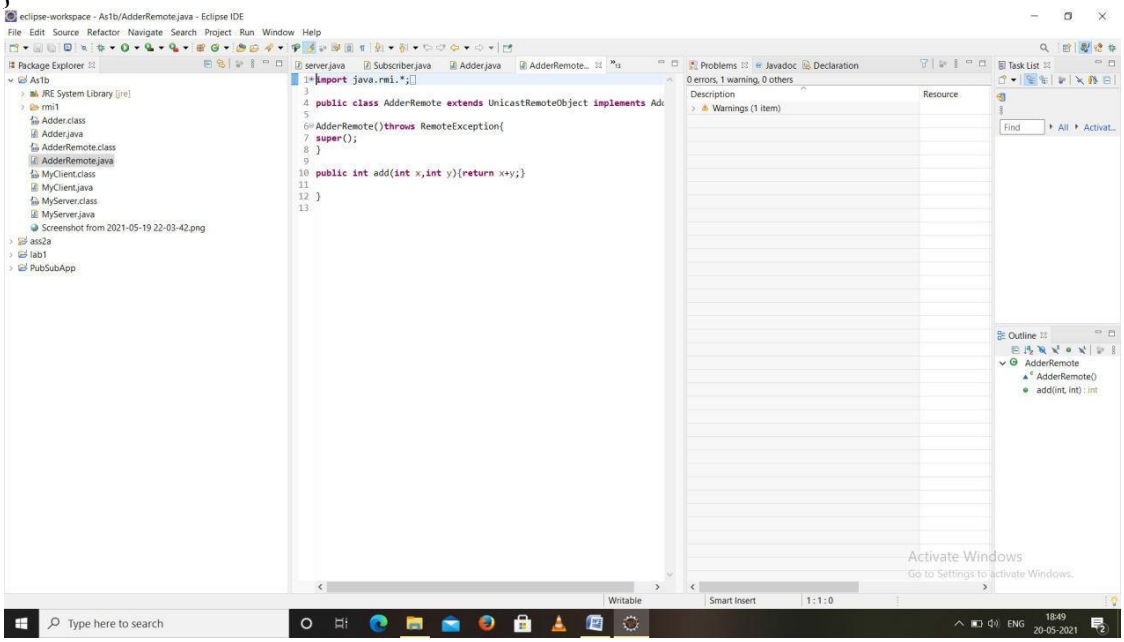
```
import java.rmi.*;
import java.util.*;

public class MyClient{

public static void main(String
        args[]){  Scanner sc=new
        Scanner(System.in);
        System.out.println(" ENTER THE FIRST NUMBER");
        int a=sc.nextInt();
        System.out.println("ENTER THE SECOND NUMBER");
        int b=sc.nextInt();
try{

Adder stub=(Adder)Naming.lookup("rmi://localhost:5000/sonoo");
System.out.println("SUM OF THE NUMBERS ARE: "+stub.add(a,b));

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

}
```



AdderRemote.java

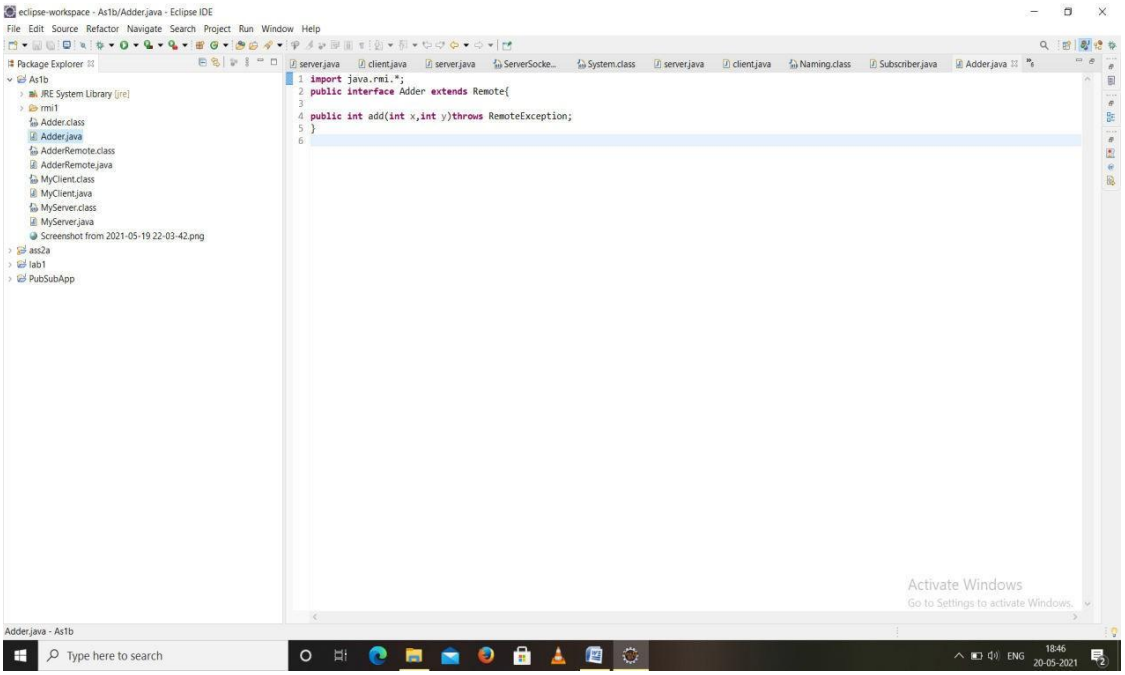
```
import java.rmi.*;
import java.rmi.server.*;

public class AdderRemote extends UnicastRemoteObject implements

Adder{  AdderRemote()throws RemoteException{
```

```
super();
}

public int add(int x,int y){return x+y;}

}
```

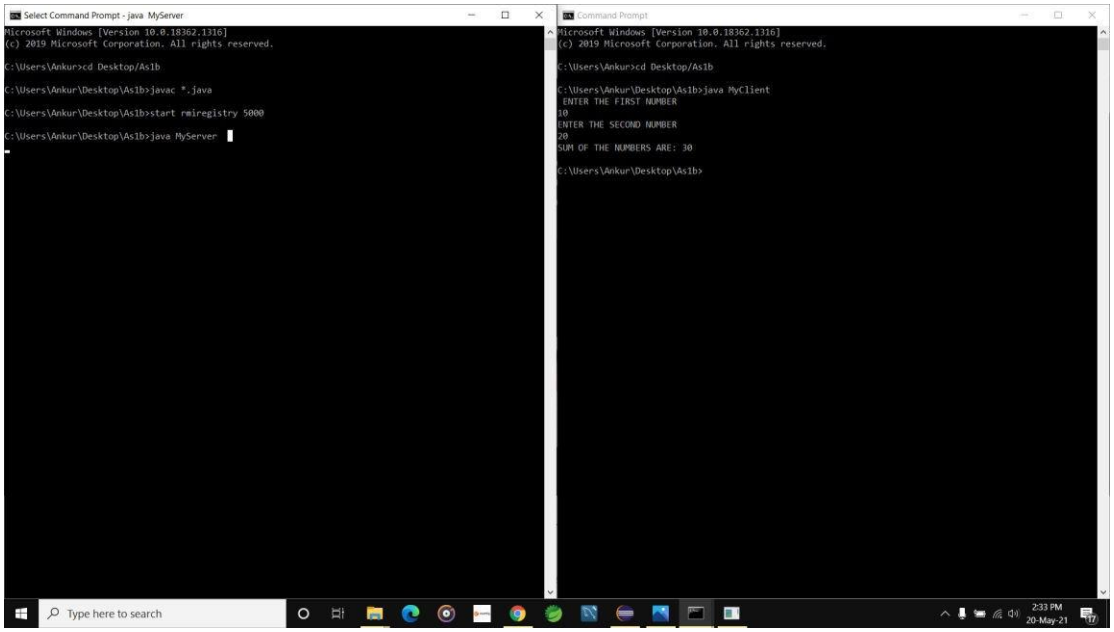


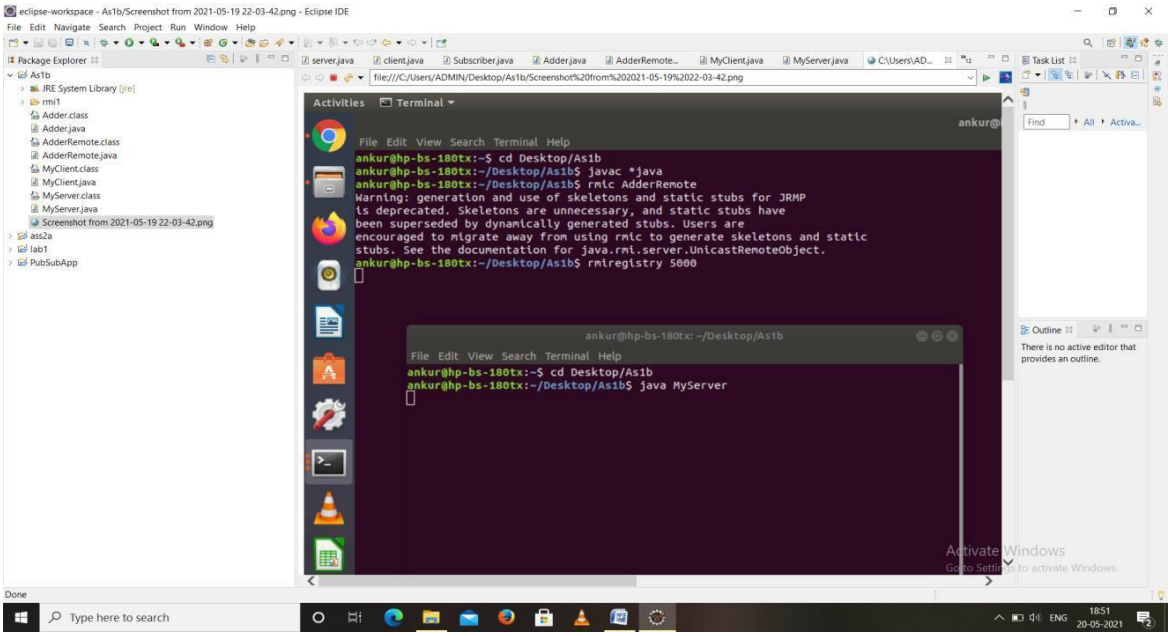
Adder.java

```
import java.rmi.*;
public interface Adder extends Remote{

public int add(int x,int y)throws RemoteException;
}
```


eclipse-workspace - As1b/Adder.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
As1b
JRE System Library [jre]
rmi1
Adder.class
Adder.java
AdderRemote.class
AdderRemote.java
MyClient.class
MyClient.java
MyServer.class
MyServer.java
Screenshot from 2021-05-19 22-03-42.png
ass2a
lab1
PubSubApp
server.java
client.java
server.java
ServerSocke...
System.class
server.java
client.java
Naming.class
Subscriber.java
Adder.java
1 import java.rmi.*;
2 public interface Adder extends Remote{
3
4 public int add(int x,int y)throws RemoteException;
5 }
6
Activate Windows
Go to Settings to activate Windows.
Adder.java - As1b
Type here to search
ENG
18:46
20-05-2021

Outputs


Select Command Prompt - java MyServer
Microsoft Windows [Version 10.0.18362.1316]
(c) 2019 Microsoft Corporation. All rights reserved.
C:\Users\Ankur>cd Desktop/As1b
C:\Users\Ankur\Desktop\As1b>javac *.java
C:\Users\Ankur\Desktop\As1b>start rmiregistry 5000
C:\Users\Ankur\Desktop\As1b>java MyServer
Command Prompt
Microsoft Windows [Version 10.0.18362.1316]
(c) 2019 Microsoft Corporation. All rights reserved.
C:\Users\Ankur>cd Desktop/As1b
C:\Users\Ankur\Desktop\As1b>java MyClient
ENTER THE FIRST NUMBER
10
ENTER THE SECOND NUMBER
20
SUM OF THE NUMBERS ARE: 30
C:\Users\Ankur\Desktop\As1b>
Type here to search
2:33 PM
20-May-21



Related Theory:

RMI provides communication between java applications that are deployed on different servers and connected remotely using objects called **stub** and **skeleton**. This communication architecture makes a distributed application seem like a group of objects communicating acrossa remote connection. These objects are encapsulated by exposing an interface, which h elps access the private state and behavior of an object through its methods.

RMIREGISTRY is a remote object registry, a Bootstrap naming service, that is used by **RMISERVER** on the same host to bind remote objects to names. Clients on local and remote hosts then lookup the remote objects and make remote method in vocations.

Keyterminologies of RMI:

The following are some of the important terminologies used in a Remote Method Invocation. **Remoteobject**: This
Is an object in as pecific JVM whose methods are exposed so they could be invoked by an other program deployed on a different JVM. **Remoteinterface**:This is a Java interface that defines the methods that existing a remote object. A remote object can implement more than one remote interface to adopt multiple remote interface behaviors.

RMI:This is away of invoking an object's methods with the help of a remote interface. It can be carried with a syntax that is similar to the local method invocation.
Stub: This is a Java object that acts as an entry point for the client object to route any outgoing requests. It exists on the client JVM and represents the handle to the remote object.

Conclusion:

Remote Method Invocation(RMI) allows you to build Java applications that are distributed among several machines.
Remote Method Invocation(RMI) allows a Java object that Executes on one machine to invoke a method of a Java object that executes on an other machine. This is an important feature, because it allows you to build distributed applications.