

**Name:** Harsh Chaudhari  
**Batch:** P-10

**Class:** BE-10  
**Roll no:** 43215

**Code:**

**AddClient.java**

```
import java.rmi.*;

public class AddClient {

    public static void main(String args[]) {

        try {

            // Get reference to the remote object

            String addServerURL = "rmi://" + args[0] + "/AddServer";

            AddServerIntf addServerIntf =

            (AddServerIntf) Naming.lookup(addServerURL);

            System.out.println("The first number is: " + args[1]);

            double d1 = Double.parseDouble(args[1]);

            System.out.println("The second number is: " + args[2]);

            double d2 = Double.parseDouble(args[2]);

            // Invoke remote method to add numbers

            System.out.println("The sum is: " + addServerIntf.add(d1, d2));

        }

        catch (Exception e) { System.out.println("Exception: "+ e);

        }}}


```

**AddServer.java**

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

```

public class AddServer {

    public static void main(String args[]) {

        try {

            //create remote object

            AddServerImpl addServerImpl = new AddServerImpl();

            //bind the remote object

            Naming.rebind("AddServer", addServerImpl);

        }

        catch (Exception e) {

            System.out.println("Exception: "+ e);

        }

    }

}

```

#### **AddserverImpl.java**

```

import java.rmi.*;

import java.rmi.server.*;

//class that implements the remote interface

public class AddServerImpl extends UnicastRemoteObject

implements AddServerIntf {

    //constructor

    public AddServerImpl() throws RemoteException {

    }

    //implement method declared in the interface

    public double add(double d1, double d2) throws RemoteException {

        return d1 + d2;

    }

}

```

### AddServerIntf.java

```
import java.rmi.*;

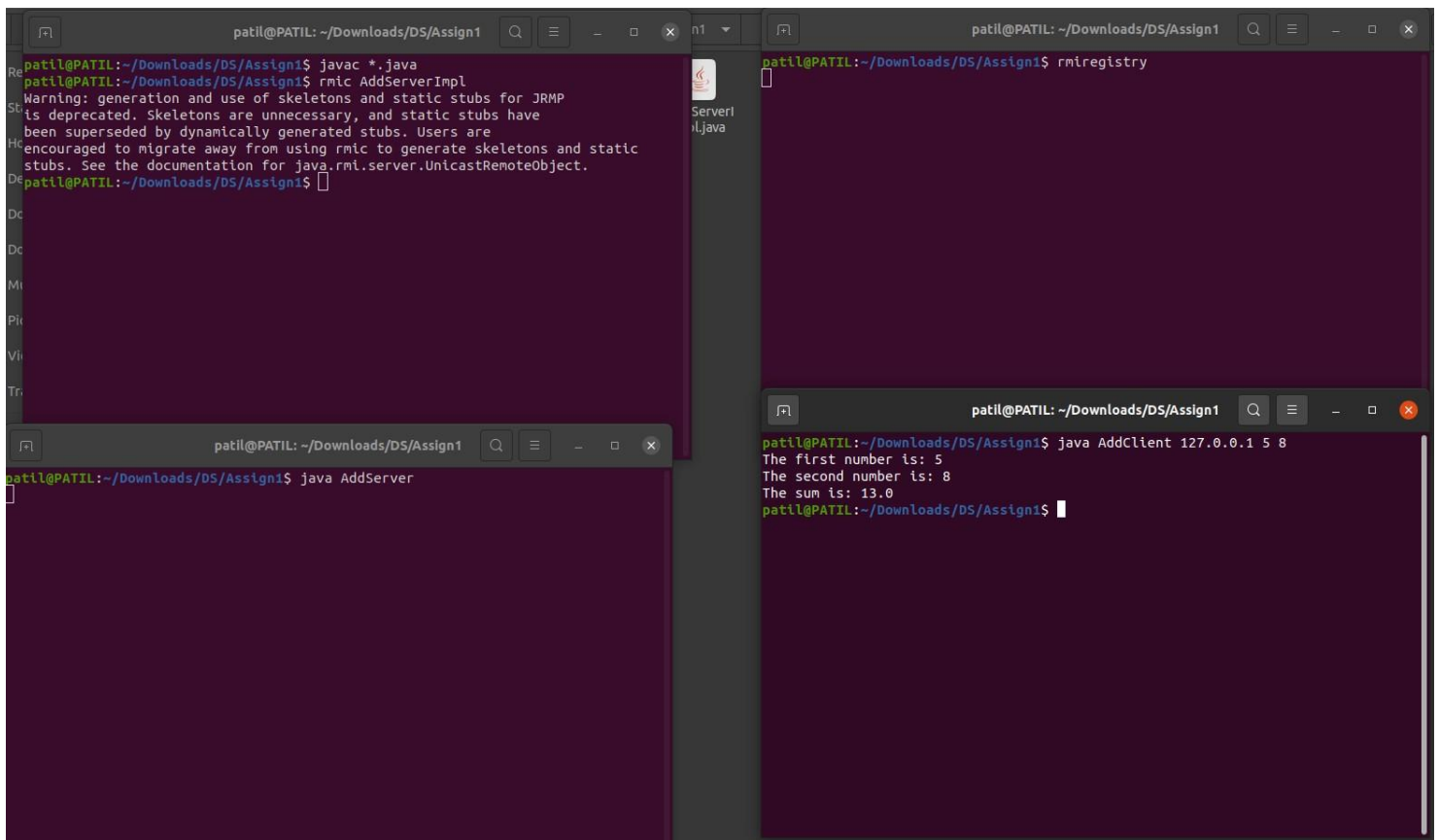
public interface AddServerIntf extends Remote {

    //method declaration

    double add(double d1, double d2) throws RemoteException;

}
```

### Output:



The image displays three terminal windows from a Linux environment, showing the steps to compile, register, and run a Java RMI application.

**Terminal 1 (Top Left):** Shows the compilation of the code.

```
patil@PATIL: ~/Downloads/DS/Assign1
patil@PATIL:~/Downloads/DS/Assign1$ javac *.java
patil@PATIL:~/Downloads/DS/Assign1$ rmic AddServerImpl
Warning: generation and use of skeletons and static stubs for JRMP
is deprecated. Skeletons are unnecessary, and static stubs have
been superseded by dynamically generated stubs. Users are
encouraged to migrate away from using rmic to generate skeletons and static
stubs. See the documentation for java.rmi.server.UnicastRemoteObject.
patil@PATIL:~/Downloads/DS/Assign1$
```

**Terminal 2 (Top Right):** Shows the registration of the code with the RMI registry.

```
patil@PATIL:~/Downloads/DS/Assign1$ rmiregistry
```

**Terminal 3 (Bottom):** Shows the execution of the client.

```
patil@PATIL:~/Downloads/DS/Assign1$ java AddClient 127.0.0.1 5 8
The first number is: 5
The second number is: 8
The sum is: 13.0
patil@PATIL:~/Downloads/DS/Assign1$
```