**Poornima College of Engineering, Jaipur**

**Department of Computer Engineering**

**Advance Java Lab (5CS4-24)**

**B.Tech III Year, V Semester**

**Experiment – 05**

---

**Objective: Create a simple calculator application that demonstrates the use of RMI. You are not required to create GUI.**

We have to run these four classes to make calculator using RMI.

- Calculator.java
- CalculatorImpl.java
- CalculatorClient.java
- CalculatorServer.java

**Calculator.java**

```
public interface Calculator
        extends   java.rmi.Remote   {
   public long add(long a, long b)
      throws java.rmi.RemoteException;

   public long sub(long a, long b) throws
      java.rmi.RemoteException;

   public long mul(long a, long b) throws
      java.rmi.RemoteException;

   public long div(long a, long b)
      throws java.rmi.RemoteException;
```

}

**CalculatorImpl.java**

```
public class CalculatorImpl extends
                java.rmi.server.UnicastRemoteObject
            implements Calculator {
            // Implementations must have an
            //explicit constructor
            // in order to declare the
            //RemoteException exception

            public CalculatorImpl()
                throws  java.rmi.RemoteException  {
                super();
            }
            public long add(long a, long b)
                throws  java.rmi.RemoteException  {
                return a + b;
            }
            public long sub(long a, long b)
                throws  java.rmi.RemoteException  {
                return a - b;
            }
            public long mul(long a, long b)
                throws java.rmi.RemoteException {
```

```
    return a * b;
  }

  public long div(long a, long b)
    throws java.rmi.RemoteException {
    return a / b;
  }}
```

**CalculatorClient.java**

```
import java.rmi.Naming;
import java.rmi.RemoteException;
import      java.net.MalformedURLException;
import java.rmi.NotBoundException;

public class CalculatorClient {

  public static void main(String[] args) {
    try {
      Calculator c = (Calculator)
                Naming.lookup(
        "rmi://localhost/CalculatorService");
      System.out.println( c.sub(4, 3) );
      System.out.println( c.add(4, 5) );
      System.out.println( c.mul(3, 6) );
      System.out.println( c.div(9, 3) );
```

```
}
catch (MalformedURLException murle) {
   System.out.println();
   System.out.println(
   "MalformedURLException");
   System.out.println(murle);
}
catch (RemoteException re) {
   System.out.println();
   System.out.println(
            "RemoteException");
   System.out.println(re);
}
catch (NotBoundException nbe) {
   System.out.println();
   System.out.println(
           "NotBoundException");
   System.out.println(nbe);
}
catch (
   java.lang.ArithmeticException
                        ae)                {
   System.out.println();
   System.out.println(
   "java.lang.ArithmeticException");
   System.out.println(ae);}}
```

**CalculatorServer.java**

```
import java.rmi.Naming;

public class CalculatorServer {

  public CalculatorServer() {
    try {
      Calculator c = new CalculatorImpl();
      Naming.rebind("rmi://localhost:1099/CalculatorService", c);
    } catch (Exception e) {
      System.out.println("Trouble: " + e);
    }
  }

  public static void main(String args[]) {
    new CalculatorServer();
  }
}
```

*//Now use rmic to create the stub and skeleton class files.*

```
 > rmic CalculatorImpl
```

Running the RMI System

*You are now ready to run the system! You need to start three consoles, one for the server, one for the client, and one for the RMIRegistry.*

- *Start with the Registry. You must be in the directory that contains the classes you have written. From there, enter the following:*

  > rmiregistry

- *If all goes well, the registry will start running and you can switch to the next console.*

  *In the second console start the server hosting the CalculatorService, and enter the following*:

  > java CalculatorServer

- *It will start, load the implementation into memory and wait for a client connection.*

  *In the last console, start the client program.*

  > java CalculatorClient//

**OUTPUT –**

1

9

18