**Practical No.3**

**A].Aim**: Implement a Server calculator containing ADD(), MUL(),SUB(),DIV() Using RMI

**CalciClient.java**

package com.mycompany.calciclient;

import java.net.MalformedURLException;

import java.rmi.Naming;

import java.rmi.NotBoundException;

import java.rmi.RemoteException;

import java.util.Scanner;

public class CalciClient {

public static void main(String[] args) throws NotBoundException, MalformedURLException, RemoteException {

Scanner sc=new Scanner(System.in);

try{

CalciInterface c= (CalciInterface)Naming.lookup("rmi://localhost:1099/CalciInterface");

System.out.println("Client is connected to server.");

System.out.println("Please enter your choice: \n1. add\n2. sub\n3. mul\n4. div\n");

int choice=sc.nextInt();

int x,y;

switch(choice){

case 1:

System.out.println("Enter x and y: ");

x=sc.nextInt();

y=sc.nextInt();

System.out.println(c.add(x,y));

break;

case 2:

System.out.println("Enter x and y: ");

x=sc.nextInt();

y=sc.nextInt();

System.out.println(c.sub(x,y));

break;

case 3:

System.out.println("Enter x and y: ");

x=sc.nextInt();

y=sc.nextInt();

System.out.println(c.mul(x,y));

break;

case 4

System.out.println("Enter x and y: ");

x=sc.nextInt();

y=sc.nextInt();

System.out.println(c.div(x,y));

break; }}

catch(Exception e){} }}

**Calciserver.java**

package com.mycompany.calciclient;

import java.rmi.NotBoundException;

import java.rmi.Remote;

import java.rmi.RemoteException;

import java.rmi.registry.Registry;

public class CalciServer {

public static void main(String[] args) throws RemoteException, NotBoundException {

Registry r=java.rmi.registry.LocateRegistry.createRegistry(1099);

r.rebind("CalciInterface", (Remote) new CalciRMI());

System.out.println("server is running");}}

**CalciRMI.java**

package com.mycompany.calciclient;

import java.rmi.RemoteException;

import java.rmi.server.UnicastRemoteObject;

public class CalciRMI extends UnicastRemoteObject implements CalciInterface {

public CalciRMI() throws RemoteException{

int a,b;}

public int add (int a,int b)thro0000ws RemoteException{

return a+b;}

public int sub (int a,int b)throws RemoteException{

return a-b;}

public int mul (int a,int b)throws RemoteException{

return a\*b;}

public int div (int a,int b)throws RemoteException{ return a/b;}

public static void main(String[] args){

}}

**CalciInterface.java**

package com.mycompany.calciclient;

import java.rmi.Remote;

import java.rmi.RemoteException;

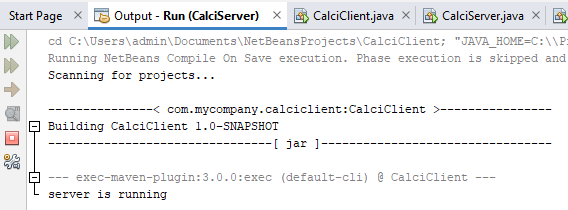
public interface CalciInterface extends Remote{public int add (int x,int y)throws RemoteException;

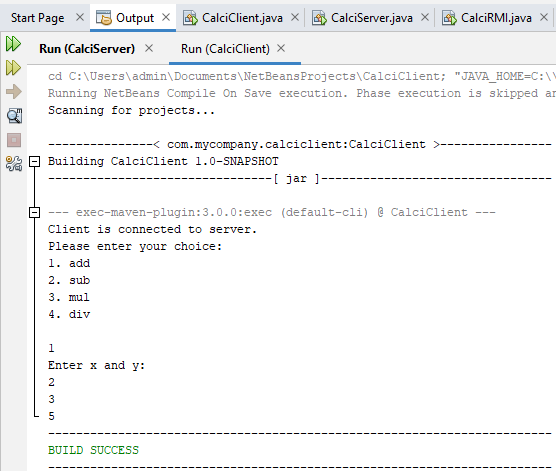
public int sub(int x,int y) throws RemoteException;

public int mul(int x,int y) throws RemoteException;

public int div(int x,int y) throws RemoteException; }

**OUTPUT**





**B].Aim:** Retrieve time and date function from server to client. This program should display server date and time.

**DatetimeClient.java**

package com.mycompany.datetimeclient;

import java.net.MalformedURLException;

import java.rmi.\*;

public class DateTimeClient {

public static void main(String args[]) {

try{

DateTimeInterface intf=(DateTimeInterface)

Naming.lookup("rmi://localhost:1099/DateServer");

System.out.println("The date on the server is: "+intf.getDate()); }

catch(MalformedURLException | NotBoundException | RemoteException e) {

} } }

**Datetimeserver.java**

package com.mycompany.datetimeclient;

import java.net.MalformedURLException;

import java.rmi.\*;

import java.rmi.registry.Registry;

public class DateTimeServer {

public static void main(String args[]) {

try{

Registry r= java.rmi.registry.LocateRegistry.

createRegistry(1099);

DateTimeRMI di=new DateTimeRMI();

Naming.rebind("DateServer", (Remote) di);

System.out.println("Datetime Server is ready"); }

catch(MalformedURLException | RemoteException e){} }}

**Datetimeinterface.java**

package com.mycompany.datetimeclient;

import java.rmi.\*;

public interface DateTimeInterface extends Remote{

String getDate() throws RemoteException;}

**DatetimeRMI.java**

package com.mycompany.datetimeclient;

import java.rmi.\*;

import java.rmi.server.\*;

import java.util.\*;

public class DateTimeRMI extends UnicastRemoteObject implements DateTimeInterface {

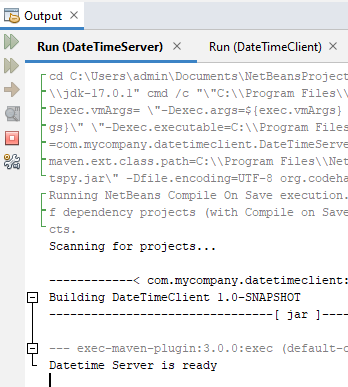
public DateTimeRMI() throws RemoteException {}

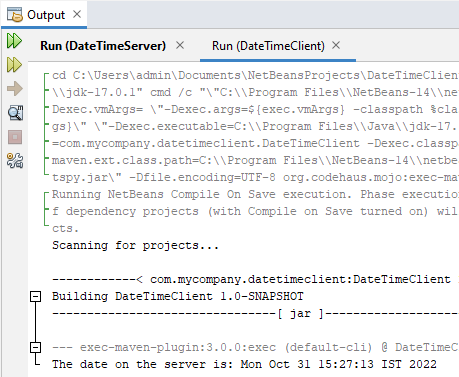
public String getDate() {

Date d=new Date();

return(d.toString()); }}

**output:**





**C]. Aim** Equation solver. The client should provide an equation to the server through an interface. The server will solve the expression given by the client. Such as (a-b)2 = a2 –2ab + b2 , (a+b)2 = a2 +2ab + b2, (a +b)3 = a3+3a2b+3ab2+b3  etc

**EqSolverClient .java**

package com.mycompany.eqsolverclient;

import java.rmi.\*;

import java.io.\*;

public class EqSolverClient {

public static void main(String[] args) {

try {

int num1=0, num2=0, res=0,choice;

EqSolverInterface object=(EqSolverInterface)Naming.lookup("hello");

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

System.out.println("Equations:-");

System.out.println("1. (a-b)2");

System.out.println("2. (a+b)2");

System.out.println("3. (a-b)3");

System.out.println("4. (a+b)3");

System.out.println("5. Exit");

while (true)

{ System.out.println("Choose the equation: ");

choice=Integer.parseInt(br.readLine());

if(choice<=4)

{

System.out.println("Enter the values of a and b");

num1=Integer.parseInt(br.readLine());

num2=Integer.parseInt(br.readLine());

}

switch(choice)

{ case 1:{

res=object.solveEq1(num1,num2);

System.out.println("Result is: "+res);

break;}

case 2:

{res=object.solveEq2(num1,num2);

System.out.println("Result is: "+res);

break;}

case 3:

{res=object.solveEq3(num1,num2);

System.out.println("Result is: "+res); break;} case 4:

{res=object.solveEq4(num1,num2);

System.out.println("Result is: "+res);

break;} case 5:

{ System.exit(0);

break;}

default:

{ System.out.println("Invalid option"); break; }

} } }

catch(Exception e) {} }}

**EqSolverServer.java**

package com.mycompany.eqsolverclient;

import java.rmi.\*;

import java.rmi.registry.Registry;

public class EqSolverServer {

public static void main(String[] args) throws RemoteException, NotBoundException {

try{

Registry r= java.rmi.registry.LocateRegistry.createRegistry(1099);

EquationSolverRMI obj=new EquationSolverRMI();

r.rebind("hello", obj);

System.out.println("Equation Solver Server is ready"); }

catch(Exception e) {} } }

**EqSolverInterface .java**

package com.mycompany.eqsolverclient;

import java.rmi.\*;

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface EqSolverInterface extends Remote{

public int solveEq1(int a, int b) throws RemoteException;

public int solveEq2(int a, int b) throws RemoteException;

public int solveEq3(int a, int b) throws RemoteException;

public int solveEq4(int a, int b) throws RemoteException; }

**EquationSolverRMI.java**

package com.mycompany.eqsolverclient;

import java.rmi.\*;

import java.rmi.server.\*;

import java.util.\*;

public class EquationSolverRMI extends UnicastRemoteObject implements EqSolverInterface {

public EquationSolverRMI() throws RemoteException {}

public int solveEq1(int a,int b) throws RemoteException {

int ans=(a\*a)-(2\*a\*b)+(b\*b);

return ans; }

public int solveEq2(int a,int b) throws RemoteException {

int ans=(a\*a)+(2\*a\*b)+(b\*b);

return ans; }

public int solveEq3(int a,int b) throws RemoteException {

int ans=(a\*a\*a)-(3\*a\*a\*b)+(3\*a\*b\*b)-(b\*b\*b); return ans; }

public int solveEq4(int a,int b) throws RemoteException { int ans=(a\*a\*a)+(3\*a\*a\*b)+(3\*a\*b\*b)+(b\*b\*b);

return ans; } }

output

