PRACTICAL NO. 3

Remote Method Invocation

Exercise:

1. Design a Graphical User Interface (GUI) based calculator. (scientific or standard). Operations should be performed using both mouse and keyboard.

Calculator.java

```
package mypackage;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface Calculator extends Remote{
	public void calculate() throws RemoteException;

}
```

Main.java

```
package mypackage;

import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class Main extends UnicastRemoteObject implements Calculator{

	protected Main() throws RemoteException {
		super();
	}

	private static final long serialVersionUID = 1L;

	@Override
	public void calculate() throws RemoteException {
		new calculator();

	}


}
```

calculator.java

```
package mypackage;

import java.awt.event.*;
import java.awt.*;
import javax.swing.*;
```

PRACTICAL NO. 3

```
public class calculator extends JFrame implements ActionListener
{
   JButton b10,b11,b12,b13,b14,b15;
   JButton b[]=new JButton[10];
    int i,r,n1,n2;
    JTextField res;
    char op;
   public calculator()
  {
     super("calulator");
      setLayout(new BorderLayout());
      JPanel p=new JPanel();
      p.setLayout(new GridLayout(4,4));
      for(int i=0;i<=9;i++)
      {
        b[i]=new JButton(i+"");
        p.add(b[i]);
        b[i].addActionListener(this);
      }
      b10=new JButton("+");
      p.add(b10);
      b10.addActionListener(this);

      b11=new JButton("-");
      p.add(b11);
      b11.addActionListener(this);

      b12=new JButton("*");
      p.add(b12);
      b12.addActionListener(this);

      b13=new JButton("/");
      p.add(b13);
      b13.addActionListener(this);

      b14=new JButton("=");
      p.add(b14);
      b14.addActionListener(this);

      b15=new JButton("C");
      p.add(b15);
      b15.addActionListener(this);

      res=new JTextField(10);
      add(p,BorderLayout.CENTER);
      add(res,BorderLayout.NORTH);
      setVisible(true);
      setSize(200,200);
     }
public void actionPerformed(ActionEvent ae)
```

```
{
  JButton pb=(JButton)ae.getSource();
          if(pb==b15)
          {
           r=n1=n2=0;
           res.setText("");
          }
          else
                    if(pb==b14)
                    {
                     n2=Integer.parseInt(res.getText());
                     eval();
                     res.setText(""+r);
                    }

                    else
                    {
                      boolean opf=false;
                      if(pb==b10)
                              { op='+';
                                opf=true;
                              }
                      if(pb==b11)
                              { op='-';opf=true;}
                      if(pb==b12)
                              { op='*';opf=true;}
                      if(pb==b13)
                              { op='/';opf=true;}

                      if(opf==false)
                      {
                       for(i=0;i<10;i++)
                       {
                              if(pb==b[i])
                              {
                              String t=res.getText();
                              t+=i;
                              res.setText(t);
                              }
                       }
                      }
                      else
                      {
                              n1=Integer.parseInt(res.getText());
                              res.setText("");
                      }
                    }
}
int eval()
{
	switch(op)
	{
	case '+': r=n1+n2; break;
	case '-':	r=n1-n2;   break;
	case '*':	r=n1*n2; break;
	case '/':	r=n1/n2; break;

	}
	return 0;
}
}
```

Server.java

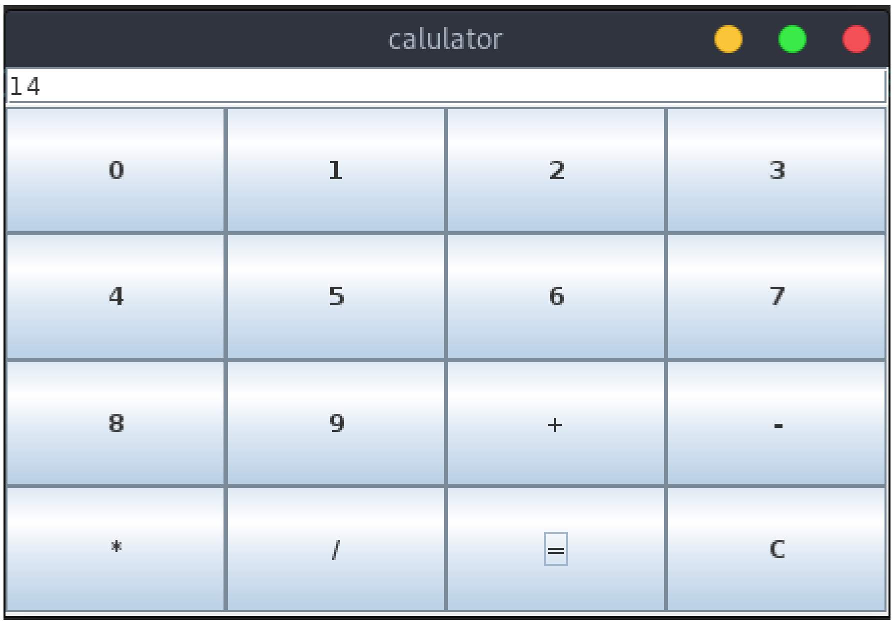
```
package mypackage;

import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;


public class Server {
	public static void main(String[] args) {
		try
		{
			Calculator cal=new Main();
			LocateRegistry.createRegistry(1900);
			Naming.rebind("rmi://localhost:1900/calculator", cal);
		}
		catch(Exception ex)
		{
			System.out.println(ex);
		}

	}
}
```

Client.java

```
package mypackage;

import java.rmi.Naming;

public class Client {

	public static void main(String[] args) {
		try
		{
			Calculator
access=(Calculator)Naming.lookup("rmi://localhost:1900/calculator");
			access.calculate();
```

```
                }
                catch(Exception ex)
                {
                        System.out.println(ex);
                }

        }

}
```

Output:



2. Retrieve day, time and date function from server to client. This program should display server day, date and time.

Dater.java

```
package datetime;

import java.rmi.Remote;
import java.rmi.RemoteException;
import java.sql.Date;
import java.time.LocalDateTime;

public interface Dater extends Remote {
        public LocalDateTime getDate() throws RemoteException;
}
```

Main.java

```
package datetime;
```

```
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.sql.Date;
import java.time.LocalDate;
import java.time.LocalDateTime;

public class Main extends UnicastRemoteObject implements Dater{
        Main() throws RemoteException
        {
                super();
        }
        @Override
        public LocalDateTime getDate() throws RemoteException {
                return java.time.LocalDateTime.now();
        }


}
```

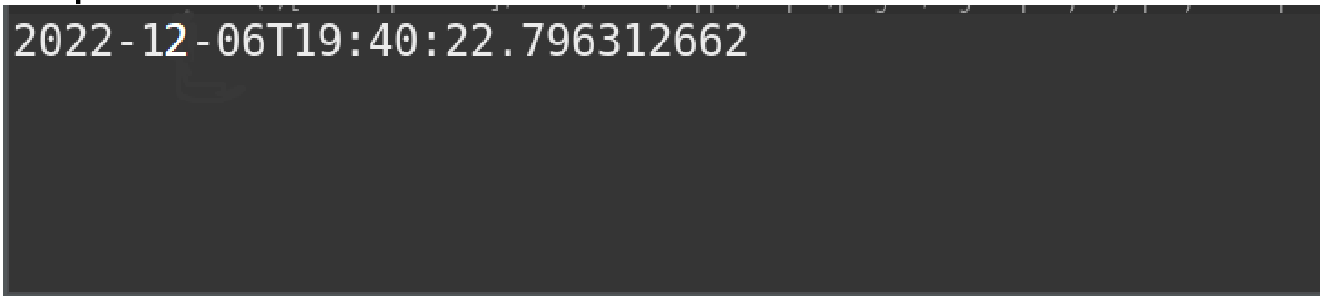
Server.java

```
package datetime;

import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;

public class Server {

        public static void main(String[] args) {
                try
                {
                        Dater dt=new Main();
                        LocateRegistry.createRegistry(1900);
                        Naming.rebind("rmi://localhost:1900/datedisplay", dt);
                }
                catch(Exception ex)
                {
                        System.out.println(ex);
                }

        }

}
```

Client.java

```
package datetime;

import java.rmi.Naming;
import java.sql.Date;
import java.time.LocalDateTime;

public class Client {

	public static void main(String[] args) {
		LocalDateTime answer;
		try
		{
			Dater
access=(Dater)Naming.lookup("rmi://localhost:1900/datedisplay");
			answer=access.getDate();
			System.out.println(answer);
		}
		catch(Exception ex)
		{
			System.out.println(ex);
		}

	}

}
```

Output:



```
2022-12-06T19:40:22.796312662
```

3. Equation solver. The client should provide an equation to the server through an interface. The server will solve the expression given by the client. (a-b)2 = a2 –2ab + b2;
If a = 5 and b = 2 then return value = 52 – 2*5*2 + 22 = 9.

Equator.java

```
package mypackage;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface Equator extends Remote{
	public int getEquation(int a,int b) throws RemoteException;
}
```

Main.java

```
package mypackage;

import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class Main extends UnicastRemoteObject implements Equator{

        protected Main() throws RemoteException {
                super();
        }

        private static final long serialVersionUID = 1L;

        @Override
        public int getEquation(int a, int b) throws RemoteException {
                int result=((a*a)-(2*a*b)+(b*b));
                return result;
        }

}
```

Server.java

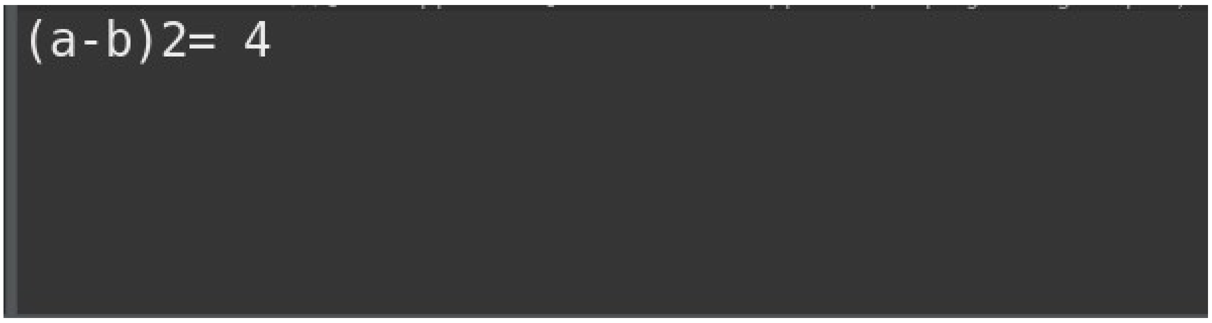
```
package mypackage;

import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;

public class Server {

        public static void main(String[] args) {
                try
                {
                        Equator eq=new Main();
                        LocateRegistry.createRegistry(1900);
                        Naming.rebind("rmi://localhost:1900/equationsolver", eq);
                }
                catch(Exception ex)
                {
                        System.out.println(ex);
                }

        }

}
```

PRACTICAL NO. 3

Client.java

```
package mypackage;

import java.rmi.Naming;

public class Client {

	public static void main(String[] args) {
		try
		{
			Equator
access=(Equator)Naming.lookup("rmi://localhost:1900/equationsolver");
			int answer=access.getEquation(5, 3);
			System.out.println("(a-b)2= "+answer);
		}
		catch(Exception ex)
		{
			System.out.println(ex);
		}

	}

}
```

Output:



```
(a-b)2= 4
```