5]Design a Graphical User Interface (GUI) to find factorial of a given numbers. Implement using RMI. Program:

FactorialServer.java

```
package defaulter_assignss;
import java.rmi.Remote;
import java.rmi.RemoteException;
public interface
FactorialServer extends Remote { long calculateFactorial(int
   number) throws RemoteException;
}
```

FactorialServerImpl.java

```
package defaulter_assignss;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
public class FactorialServerImpl extends UnicastRemoteObject implements
FactorialServer {     public FactorialServerImpl() throws RemoteException {
   }
   @Override     public long calculateFactorial(int number) throws
RemoteException {        if (number < 0) {           throw new
RemoteException("Input should be a non-negative integer");
      }
      long factorial = 1;        for (int i
= 1; i <= number; i++)
{ factorial *= i;
      }           return
      factorial;
   }
}
```

FactorialClient.java

```
package defaulter_assignss;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.awt.event.ActionEvent;
```

```
import java.awt.event.ActionListener; import javax.swing.JButton;
import javax.swing.JFrame; import javax.swing.JLabel; import
javax.swing.JOptionPane; import javax.swing.JPanel; import
javax.swing.JTextField; import javax.swing.SwingUtilities; public class
FactorialClient { private JFrame frame; private JTextField textField;
private JLabel resultLabel;
   public static void main(String[] args) {
     SwingUtilities.invokeLater(() -> new FactorialClient().createAndShowGUI());
   }
   private void createAndShowGUI() {        frame
=            new            JFrame("Factorial            Calculator");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JPanel panel = new JPanel();
frame.getContentPane().add(panel);        textField = new
JTextField(10);
      JButton calculateButton = new JButton("Calculate Factorial"); resultLabel
= new JLabel();
calculateButton.addActionListener(new ActionListener() {
         @Override          public void
actionPerformed(ActionEvent e) {
calculateFactorial();
         }
      });
      panel.add(new JLabel("Enter a non-negative integer: "));
panel.add(textField); panel.add(calculateButton);
panel.add(resultLabel);        frame.pack();
frame.setVisible(true);
   }
   private void calculateFactorial()
{ String host = "localhost"; int
number; try {
         number = Integer.parseInt(textField.getText());
      } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(frame, "Invalid input. Please enter a valid
integer.", "Error", JOptionPane.ERROR_MESSAGE); return;
      }
try {
```

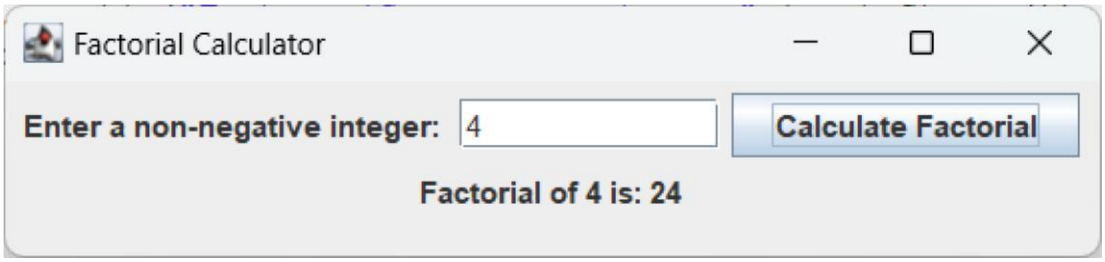
```
        Registry registry = LocateRegistry.getRegistry(host);
        FactorialServer stub = (FactorialServer) registry.lookup("FactorialServer");
long result = stub.calculateFactorial(number); resultLabel.setText("Factorial of "
+ number + " is: " + result);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(frame, "Error communicating with the server.",
"Error", JOptionPane.ERROR_MESSAGE);
        e.printStackTrace();
    }
  }
}
```

FactorialServerMain.java

```
package defaulter_assignss; import java.rmi.registry.LocateRegistry; import
java.rmi.registry.Registry; public class FactorialServerMain { public static void
main(String[] args) { // TODO Autogenerated method stub try {
        FactorialServer server = new FactorialServerImpl();
Registry registry = LocateRegistry.createRegistry(1099);
registry.rebind("FactorialServer", server);
        System.out.println("FactorialServer is running");
    } catch (Exception e) {
        System.err.println("FactorialServer exception: " + e.toString());
e.printStackTrace();
    }
  }
}
```

Output:

```
FactorialServer is running
```


Factorial Calculator
Enter a non-negative integer:
4
Calculate Factorial
Factorial of 4 is: 24