// Design a Distributed application using RMI in which a server implements maximum procedures for mathematical operations like calculating square roots, square, factorial of a number etc.

1. Define the Remote Interface (MathOperations.java):

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

public interface MathOperations extends Remote {
    double calculateSquareRoot(double number) throws RemoteException;
    double calculateSquare(double number) throws RemoteException;
    int calculateFactorial(int number) throws RemoteException;
}
```

2. Implement the Remote Object (MathOperationsImpl.java):

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

public class MathOperationsImpl extends UnicastRemoteObject implements
MathOperations {
    public MathOperationsImpl() throws RemoteException {
        // Constructor
    }

    @Override
    public double calculateSquareRoot(double number) throws RemoteException {
        return Math.sqrt(number);
    }

    @Override
    public double calculateSquare(double number) throws RemoteException {
        return number * number;
    }

    @Override
    public int calculateFactorial(int number) throws RemoteException {
        if (number == 0 || number == 1) {
            return 1;
        }
        int result = 1;
```

```
        for (int i = 2; i <= number; i++) {
            result *= i;
        }
        return result;
    }
}
```

3. Create the RMI Server (MathServer.java):

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

public class MathServer {
    public static void main(String[] args) {
        try {
            // Create the remote object
            MathOperations mathOperations = new MathOperationsImpl();

            // Create and start the RMI registry on port 1099
            LocateRegistry.createRegistry(1099);

            // Bind the remote object to a name in the RMI registry
            Naming.rebind("MathOperations", mathOperations);

            System.out.println("Server is ready.");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

4. Create the RMI Client (MathClient.java):

```
import java.rmi.Naming;
import java.util.Scanner;

public class MathClient {
    public static void main(String[] args) {
        try {
            // Look up the remote object by its name
            MathOperations mathOperations = (MathOperations)
Naming.lookup("rmi://localhost/MathOperations");
```

```
            // Get user input for the number
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter a number: ");
            double number = scanner.nextDouble();

            // Calculate square root
            double squareRoot = mathOperations.calculateSquareRoot(number);
            System.out.println("Square root of " + number + " is: " + squareRoot);

            // Calculate square
            double square = mathOperations.calculateSquare(number);
            System.out.println("Square of " + number + " is: " + square);

            // Calculate factorial
            int factorialNumber = (int) number; // Assuming factorial of a double doesn't
make sense
            int factorial = mathOperations.calculateFactorial(factorialNumber);
            System.out.println("Factorial of " + factorialNumber + " is: " + factorial);

            // Close the scanner
            scanner.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Compile all the Java source files:
javac MathOperations.java MathOperationsImpl.java MathServer.java MathClient.java

Start the RMI server in one terminal:
java MathServer

Run the RMI client in another terminal:
java MathClient