M’sila University, Department of Computer Science, ISIL

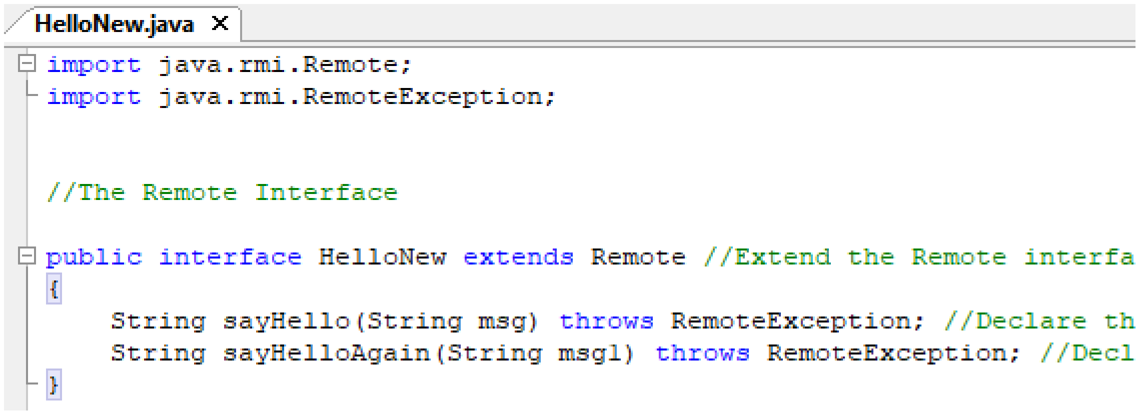
COURSE: DISTRIBUTED INFORMATION SYSTEMS **DR. R. BENTRCIA**

TP 3 Solution

Required software:

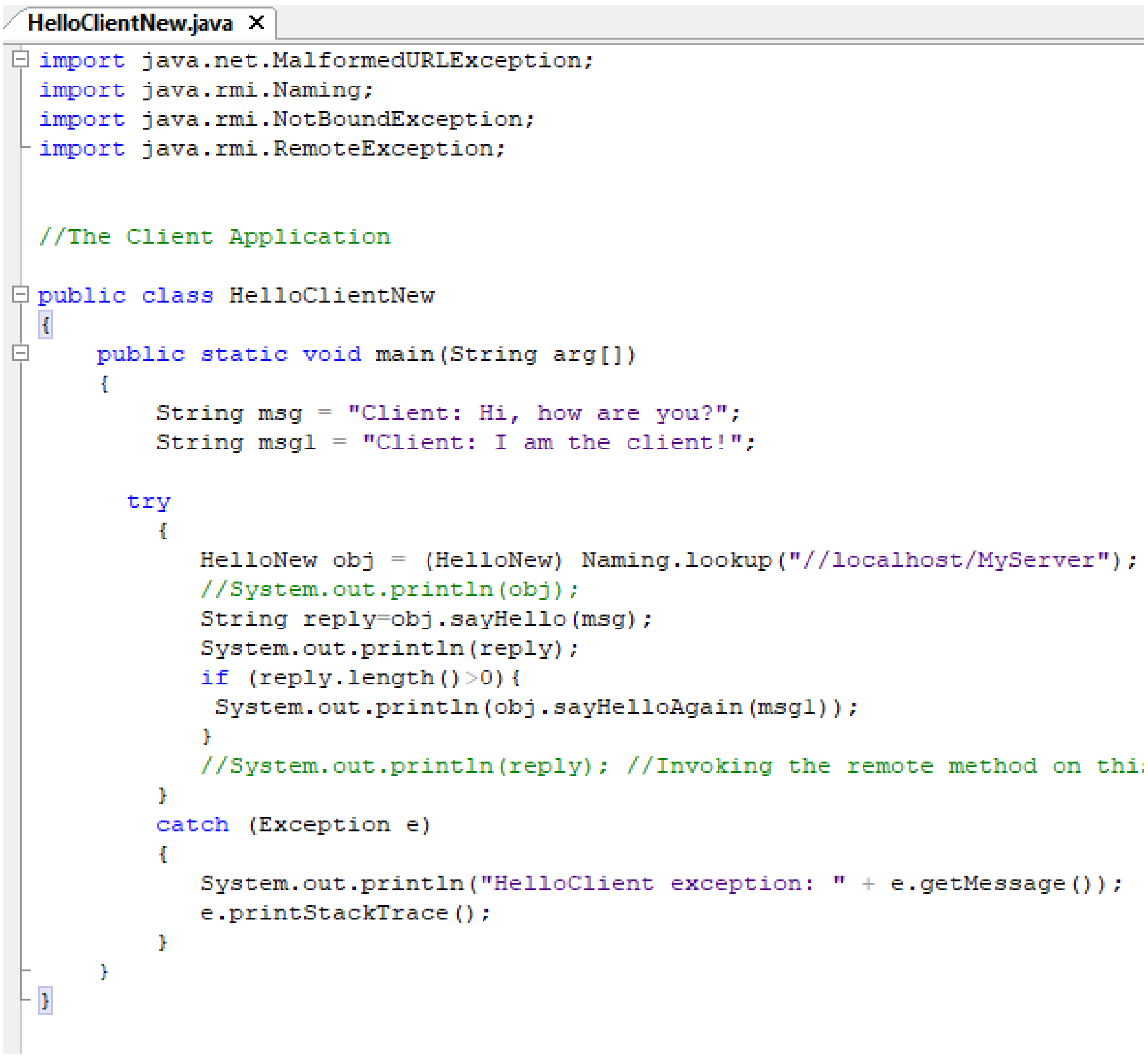
- Java Software Development Kit (jdk 1.8 or later)
- Java editor such as JCreator

Exercise 1:

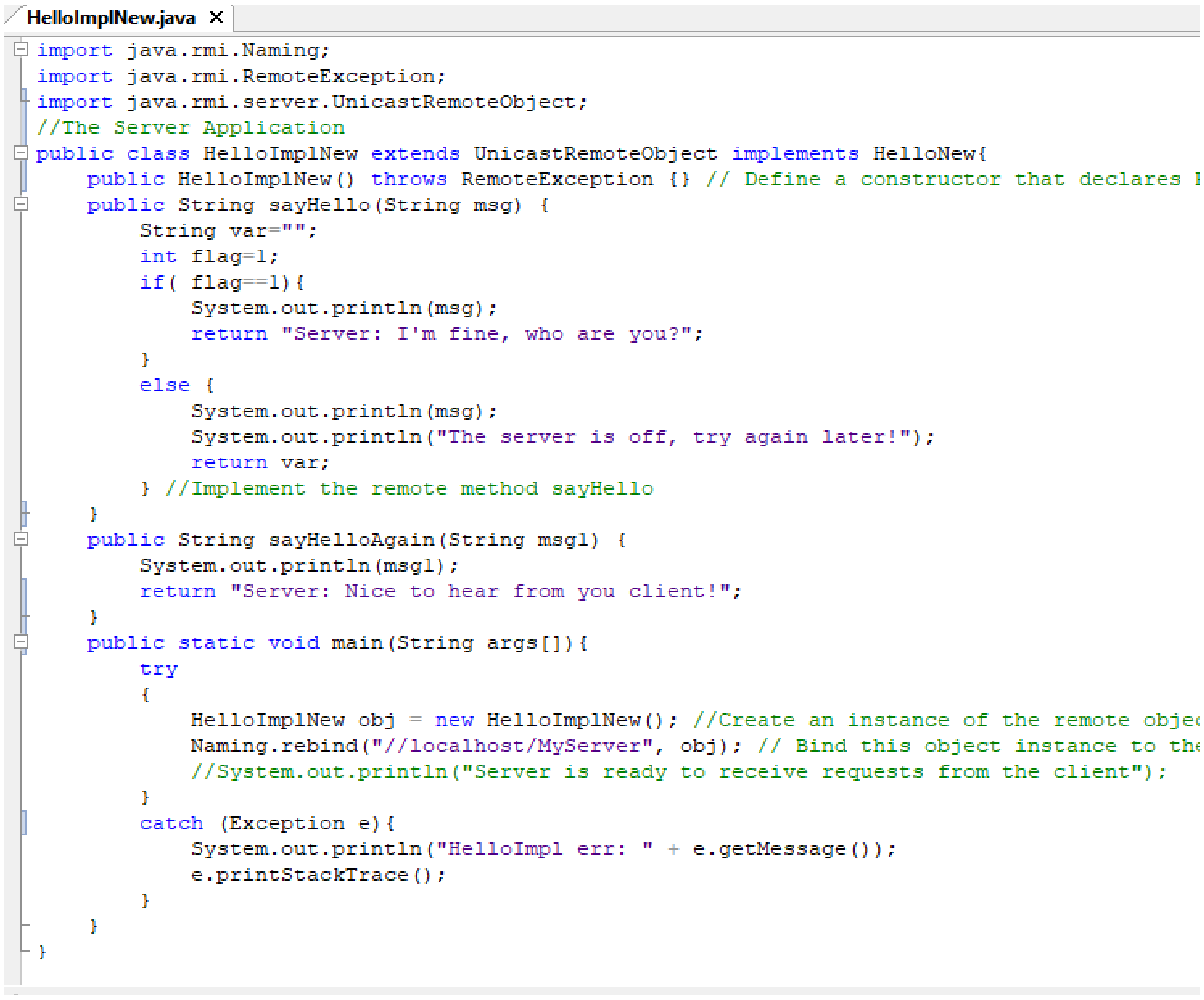


HelloNew.java

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


//The Remote Interface

public interface HelloNew extends Remote //Extend the Remote interfa
{
    String sayHello(String msg) throws RemoteException; //Declare th
    String sayHelloAgain(String msg1) throws RemoteException; //Decl
}
```

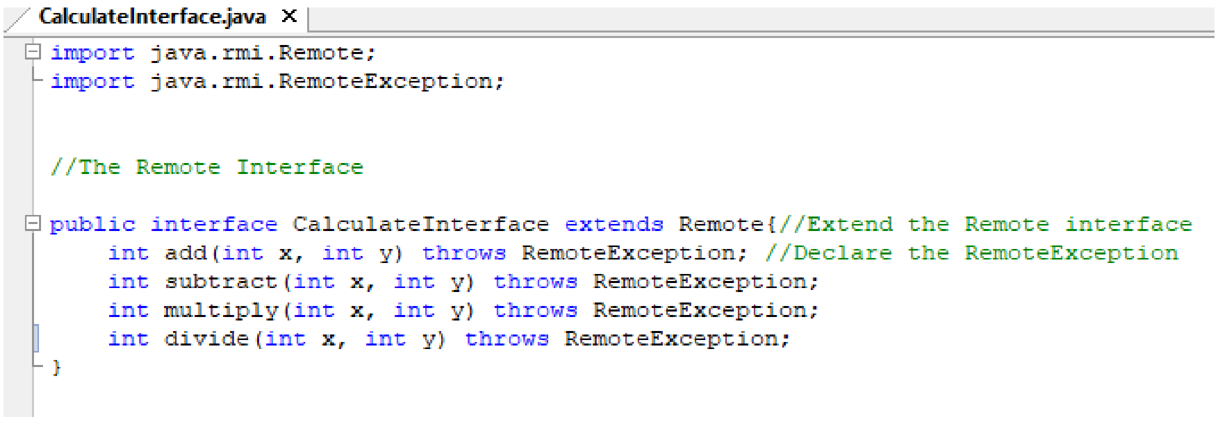

HelloClientNew.java

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


//The Client Application

public class HelloClientNew
{
    public static void main(String arg[])
    {
        String msg = "Client: Hi, how are you?";
        String msg1 = "Client: I am the client!";

      try
        {
           HelloNew obj = (HelloNew) Naming.lookup("//localhost/MyServer");
           //System.out.println(obj);
           String reply=obj.sayHello(msg);
           System.out.println(reply);
           if (reply.length()>0){
            System.out.println(obj.sayHelloAgain(msg1));
           }
           //System.out.println(reply); //Invoking the remote method on thi
        }
        catch (Exception e)
        {
           System.out.println("HelloClient exception: " + e.getMessage());
           e.printStackTrace();
        }
    }
}
```

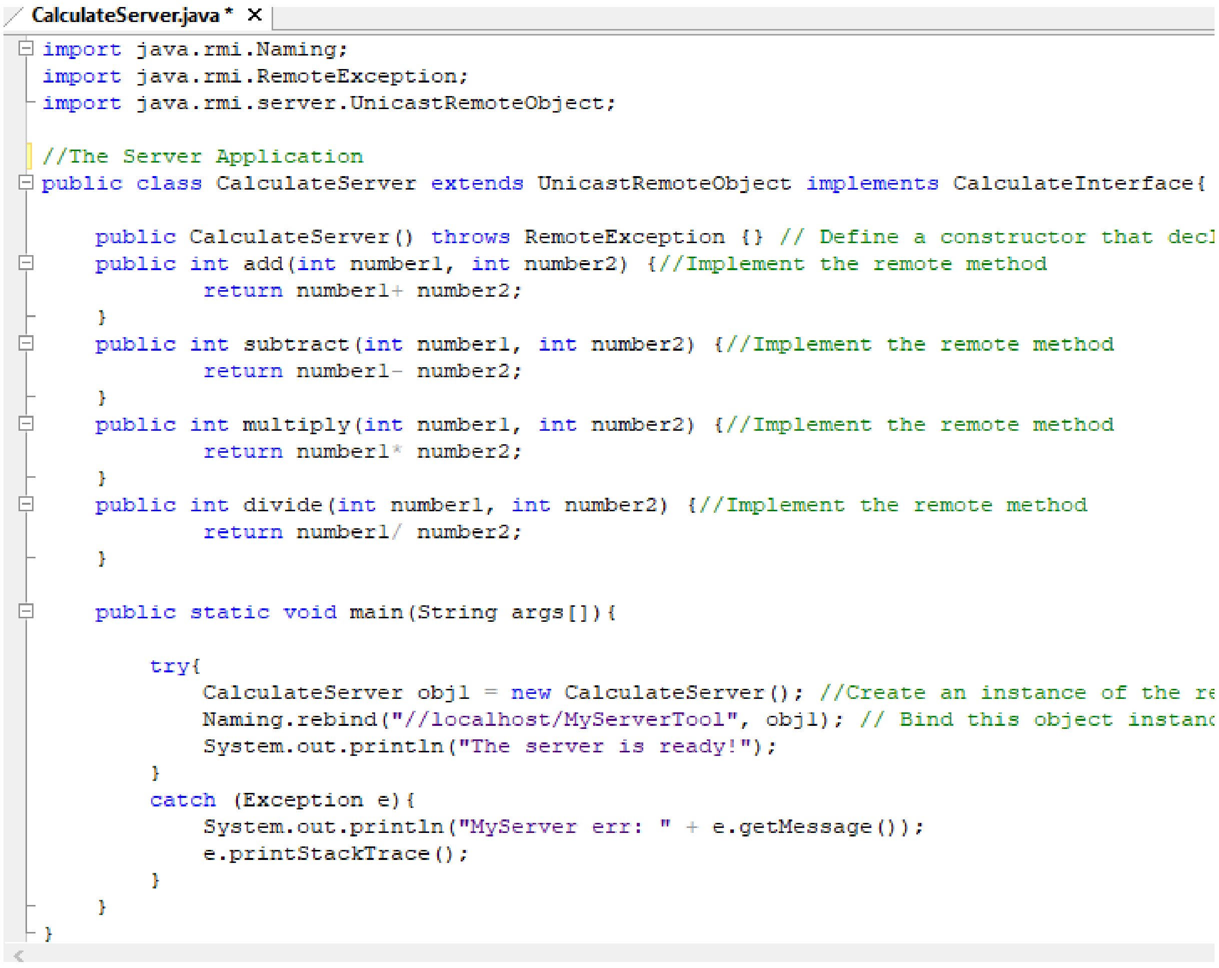

HelloImplNew.java

```
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
//The Server Application
public class HelloImplNew extends UnicastRemoteObject implements HelloNew{
    public HelloImplNew() throws RemoteException {} // Define a constructor that declares R
    public String sayHello(String msg) {
        String var="";
        int flag=1;
        if( flag==1){
            System.out.println(msg);
            return "Server: I'm fine, who are you?";
        }
        else {
            System.out.println(msg);
            System.out.println("The server is off, try again later!");
            return var;
        } //Implement the remote method sayHello
    }
    public String sayHelloAgain(String msg1) {
        System.out.println(msg1);
        return "Server: Nice to hear from you client!";
    }
    public static void main(String args[]){
        try
        {
            HelloImplNew obj = new HelloImplNew(); //Create an instance of the remote objec
            Naming.rebind("//localhost/MyServer", obj); // Bind this object instance to the
            //System.out.println("Server is ready to receive requests from the client");
        }
        catch (Exception e){
            System.out.println("HelloImpl err: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

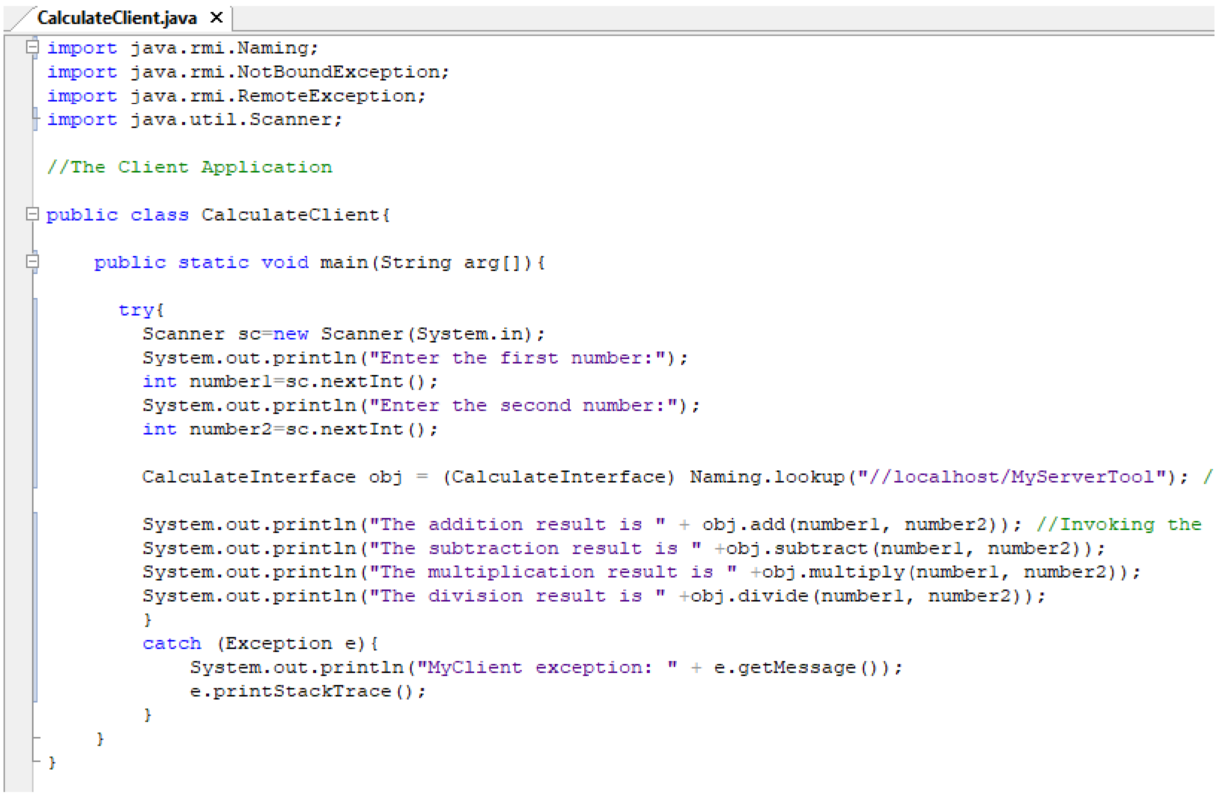
Exercise 2:


CalculateInterface.java

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


//The Remote Interface

public interface CalculateInterface extends Remote{//Extend the Remote interface
    int add(int x, int y) throws RemoteException; //Declare the RemoteException
    int subtract(int x, int y) throws RemoteException;
    int multiply(int x, int y) throws RemoteException;
    int divide(int x, int y) throws RemoteException;
}
```



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

//The Server Application
public class CalculateServer extends UnicastRemoteObject implements CalculateInterface{

    public CalculateServer() throws RemoteException {} // Define a constructor that decl
    public int add(int number1, int number2) {//Implement the remote method
            return number1+ number2;
    }
    public int subtract(int number1, int number2) {//Implement the remote method
            return number1- number2;
    }
    public int multiply(int number1, int number2) {//Implement the remote method
            return number1* number2;
    }
    public int divide(int number1, int number2) {//Implement the remote method
            return number1/ number2;
    }

    public static void main(String args[]){

        try{
            CalculateServer obj1 = new CalculateServer(); //Create an instance of the re
            Naming.rebind("//localhost/MyServerTool", obj1); // Bind this object instanc
            System.out.println("The server is ready!");
        }
        catch (Exception e){
            System.out.println("MyServer err: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```



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

//The Client Application

public class CalculateClient{

    public static void main(String arg[]){

      try{
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the first number:");
        int number1=sc.nextInt();
        System.out.println("Enter the second number:");
        int number2=sc.nextInt();

        CalculateInterface obj = (CalculateInterface) Naming.lookup("//localhost/MyServerTool"); /

        System.out.println("The addition result is " + obj.add(number1, number2)); //Invoking the
        System.out.println("The subtraction result is " +obj.subtract(number1, number2));
        System.out.println("The multiplication result is " +obj.multiply(number1, number2));
        System.out.println("The division result is " +obj.divide(number1, number2));
        }
        catch (Exception e){
            System.out.println("MyClient exception: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```