Lab 4

Q1) Arithmetic Math Operations:

Java Code:

Client.java

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

/**
 *
 * @author Abhishek Karan
 */
public class ClientRMI {

    public static void main(String[] args) {

        try {

            String addServerURL = "rmi://localhost/ServerRMI";
            RemoteInterface ri = (RemoteInterface) Naming.lookup(addServerURL);

            Scanner sc = new Scanner(System.in);

            System.out.println("Enter Num1:");
            double num1 = sc.nextDouble();
            System.out.println("Enter Num2:");
            double num2 = sc.nextDouble();

            System.out.println("Addition:" + ri.add(num1, num2));
```

```
            System.out.println("Subtraction:" + ri.sub(num1, num2));
            System.out.println("Multiplication:" + ri.mul(num1, num2));
            System.out.println("Division:" + ri.div(num1, num2));
            System.out.println("Power:" + ri.power(num1, num2));

        } catch (Exception e) {
            System.out.println(e.getMessage());
        }

    }//main()

}//Client class
```

Interface Implementation:

```
import java.rmi.*;
import java.rmi.server.*;

/**
 *
 * @author Abhishek Karan
 */
public class InterfaceImpl extends UnicastRemoteObject implements RemoteInterface{

    public InterfaceImpl() throws RemoteException {
        //super();
    }

    @Override
    public double add(double num1, double num2) throws RemoteException {
```

```
        return (num1 + num2);
    }

    @Override
    public double sub(double num1, double num2) throws RemoteException {
        return (num1 - num2);
    }

    @Override
    public double mul(double num1, double num2) throws RemoteException {
        return (num1 * num2);
    }

    @Override
    public double div(double num1, double num2) throws RemoteException {
        return (num1 / num2);
    }

    @Override
    public double power(double num1, double num2) throws RemoteException {
        return (Math.pow(num1, num2));
    }

}//interfaceImpl
```

Interface:

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

```
/**
 *
 * @author student
 */
public interface RemoteInterface extends Remote{

    double add(double num1, double num2) throws RemoteException;

    double sub(double num1, double num2) throws RemoteException;

    double mul(double num1, double num2) throws RemoteException;

    double div(double num1, double num2) throws RemoteException;

    double power(double num1, double num2) throws RemoteException;

}//RemoteInterface
```

Server:

```
import java.rmi.Naming;

/**
 *
 * @author Abhishek Karan
 */
public class ServerRMI {

    public static void main(String[] args) {
```

```
        try {
            InterfaceImpl ii = new InterfaceImpl();
            Naming.rebind("ServerRMI", ii);
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }//try-catch
    }//main()
}//Server class
```

Q2) String Operations

Java Code:

Client:

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

/**
 *
 * @author Abhishek Karan
 */
public class ClientRMI2 {

    public static void main(String[] args) {

        try {

            String addServerURL = "rmi://localhost/ServerRMI2";
            RemoteInterface2 ri = (RemoteInterface2) Naming.lookup(addServerURL);

            Scanner sc = new Scanner(System.in);
```

```
            System.out.println("Enter First String:");
            String str= sc.next();
            System.out.println("Enter Second String:");
                String str2= sc.next();

            System.out.println("Concatenation:" + ri.con(str, str2));
            System.out.println("Length:" + ri.leng(str));
            System.out.println("Upper Case:" + ri.uCase(str));
            System.out.println("Lower Case:" + ri.lCase(str));

        } catch (Exception e) {
            System.out.println(e.getMessage());
        }

    }//main()

}//Client class
```

Interface Implementation:

```
import java.rmi.*;
import java.rmi.server.*;

/**
 *
 * @author Abhishek Karan
 */
public class InterfaceImpl2 extends UnicastRemoteObject implements RemoteInterface2 {

    public InterfaceImpl2() throws RemoteException {
```

```
    }

    @Override
    public String con(String str, String str2) throws RemoteException {
      return ((str + str2));
    }

    @Override
    public int leng(String str) throws RemoteException {
        return (str.length());
    }

    @Override
    public String uCase(String str) throws RemoteException {
        return (str.toUpperCase());
    }

    @Override
    public String lCase(String str) throws RemoteException {
      return (str.toLowerCase());
    }
}//interfaceImpl
```

Interface:

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

/**
 *
```

```
 * @author Abhishek Karan
 */
public interface RemoteInterface2 extends Remote{

    String con(String str,String str2) throws RemoteException;

    int leng(String str) throws RemoteException;

    String uCase(String str) throws RemoteException;

    String lCase(String str) throws RemoteException;

}//RemoteInterface
```

Server:

```
import java.rmi.Naming;

/**
 *
 * @author Abhishek Karan
 */
public class ServerRMI2 {

    public static void main(String[] args) {

        try {
            InterfaceImpl2 ii = new InterfaceImpl2();
            Naming.rebind("ServerRMI2", ii);
        } catch (Exception e) {
            System.out.println(e.getMessage());
```

```
        }//try-catch
    }//main()
}//Server class
```

Q3) Bubble Sort

Java Code:

Client:

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

/**
 *
 * @author Abhishek Karan
 */
public class ClientRMI3 {

    public static void main(String[] args) {

        try {

            String addServerURL = "rmi://localhost/ServerRMI3";
            RemoteInterface3 ri = (RemoteInterface3) Naming.lookup(addServerURL);

            Scanner sc = new Scanner(System.in);
            System.out.print("Enter Array Elements:");
                int n=sc.nextInt();
                int arr[]=new int[n];
            System.out.print("Elements:");
                    for(int i=0;i<n;i++)
```

```
                {
                  arr[i]=sc.nextInt();
                }

            int[] arr2=ri.bbSort(arr);
                System.out.println("Output:");
                for(int i=0;i<n;i++)
                {
                System.out.print(arr2[i]+"\t");
                }

        } catch (Exception e) {
            System.out.println(e.getMessage());
        }

    }//main()

}//Client class
```

Interface Implementation:

```
import java.rmi.*;
import java.rmi.server.*;

/**
 *
 * @author Abhishek Karan
 */
public class InterfaceImpl3 extends UnicastRemoteObject implements RemoteInterface3{

    public InterfaceImpl3() throws RemoteException {
```

```
    }

    @Override
    public int[] bbSort(int[] arr) throws RemoteException {

        int temp;
        for(int i=0;i<arr.length-1;i++){
                for(int j=0;j<arr.length-i-1;j++){
                        if(arr[j]>arr[j+1])
                        {
                          temp=arr[j+1];
                          arr[j+1]=arr[j];
                          arr[j]=temp;
                        }
                }
        }
        return (arr);
    }

}//interfaceImpl
```

Interface:

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

/**
 *
 * @author student
 */
```

```
public interface RemoteInterface3 extends Remote{

    int[] bbSort(int[] arr) throws RemoteException;

}//RemoteInterface
```

Server:

```
import java.rmi.Naming;

/**
 *
 * @author Abhishek Karan
 */
public class ServerRMI3 {

    public static void main(String[] args) {

        try {
            InterfaceImpl3 ii = new InterfaceImpl3();
            Naming.rebind("ServerRMI3", ii);
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }//try-catch
    }//main()
}//Server class
```