Lab 7

RMI Program

1. Write a client server application to find the product of two numbers.

Multiply.java

```
import java.rmi.*;
public interface Multiply extends Remote{
   public int CalcMultiply(int a, int b) throws RemoteException;
}
```

MultiplyImple.java

```
import java.rmi.RemoteException;
public class MultiplyImple implements Multiply{
   @Override
   public int CalcMultiply(int a, int b) throws RemoteException {
      return a * b;
   }
}
```

Server.java

```
import java.rmi.*;
import java.rmi.registry.*;
import java.rmi.server.*;
import java.util.logging.*;
public class Server extends MultiplyImple{
   public static void main(String[] args) throws AlreadyBoundException {
      MultiplyImple obj = new MultiplyImple();
      try {
         Multiply skeleton = (Multiply) UnicastRemoteObject.exportObject(obj, 0);
         Registry reg = LocateRegistry.getRegistry();
```

```
            reg.bind("mul", skeleton);
            System.out.println("server is ready");
        } catch (RemoteException ex) {
            Logger.getLogger(Server.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Client.java

```
import java.rmi.*;
import java.rmi.registry.*;
import java.util.logging.*;
public class Client {
    public static void main(String[] args) throws NotBoundException {
        try {
            Registry reg = LocateRegistry.getRegistry(0);
            Multiply stub =  (Multiply) reg.lookup("mul");
            int result = stub.CalcMultiply(8, 9);
            System.out.println("The multiplication result is " + result);
        } catch (RemoteException ex) {
            Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

2. Write a client server application to display the reverse of a String value.

Reverse.java

```
import java.rmi.*;
public interface Reverse extends Remote{
   public String ReverseStr(String str) throws RemoteException;
}
```

ReverseImpl.java

```
import java.rmi.RemoteException;
public class ReverseImpl implements Reverse{
  @Override
   public String ReverseStr(String str) throws RemoteException {
      StringBuilder sb = new StringBuilder(str);
      sb.reverse();
      return sb.toString();
   }
   }
```

Server.java

```
   import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;

public class Server{
   public static void main(String[] args){
        try {
          ReverseImpl obj = new ReverseImpl();
```

```
            Reverse skeleton = (Reverse) UnicastRemoteObject.exportObject(obj, 0);
            Registry reg = LocateRegistry.getRegistry();
            reg.bind("rev", skeleton);
            System.out.println("server ready ");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Client.java

```
import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
public class Client {
    public static void main(String[] args) {
        try {
            Registry reg = LocateRegistry.getRegistry();
            Reverse stub = (Reverse) reg.lookup("rev");
            String reverseString = stub.ReverseStr("cody");
            System.out.println("The reverse string is :" + reverseString);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

3. Write a Client Server Application in Java where the Server provides price of different laptop models in dollar. Client Program can ask for the price of a laptop model and displays it.

Price.java

```
import java.rmi.*;

public interface Price extends Remote{
   public String modelPrice (String modelName) throws RemoteException;
}
```

PriceImpl.java

```
import java.rmi.*;
import java.util.HashMap;
import java.util.Map;

public class PriceImpl implements Price{
  private Map<String, String> prices;

   public PriceImpl(){
      prices = new HashMap<>();
      prices.put("Lenovo" , "120k");
      prices.put("Dell", "100k");
      prices.put("Mac", "250k");

   }

   @Override
   public String modelPrice(String modelName) throws RemoteException{
      return prices.get(modelName);
   }

   }
```

Server.java

```
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;

public class Server {
```

```
    public static void main(String[] args) {
        try {
            PriceImpl obj = new PriceImpl();

            Price skeleton = (Price) UnicastRemoteObject.exportObject(obj, 0);
            Registry reg = LocateRegistry.getRegistry();
            reg.bind("price", skeleton);
            System.out.println("server ready ");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Client.java

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

public class Client {

    public static void main(String[] args) {
        try {
            Registry reg = LocateRegistry.getRegistry();
            Price stub =(Price) reg.lookup("price");
            String price = stub.modelPrice("Lenovo");
            System.out.println("Price of Levono Model =" + price);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Discussion and Conclusion:

In this lab, we learn and discussed about RMI. We learn about Client-Server model and solve simple multiplication, reverse of string and about the list of price using RMI . steps to run RMI program: start rmiregistry, javac *.java, java server and then java client.