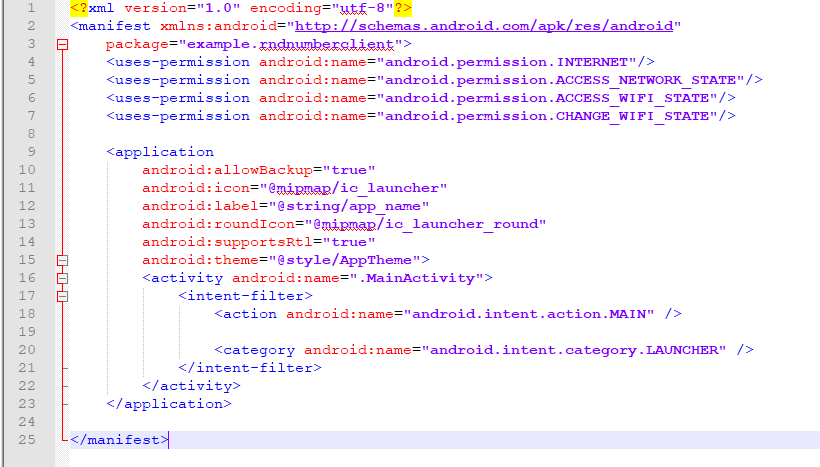
1. Remote Simple Calculator – The server was created using the Server Socket Class in Java.

***Server.java***

import java.io.IOException;  
import java.net.ServerSocket;  
import java.net.Socket;   
import java.util.Scanner;  
import java.lang.\*;  
import java.io.\*;   
  
public class Server{  
 public static void main(String[] args){  
 try{  
 ServerSocket serverSocket = new ServerSocket(4444);  
 System.out.println("Server Started...");  
 while(true){  
 new Thread(new ClientConnectionThread(serverSocket.accept())).start();  
 }  
   
 }catch(IOException e) {e.printStackTrace();}  
   
 }  
  
}  
class ClientConnectionThread implements Runnable{  
 private Socket socket;  
 public ClientConnectionThread(Socket socket){  
 this.socket = socket;  
 }  
 @Override  
 public void run(){  
 try{  
 DataInputStream dIn = new DataInputStream(socket.getInputStream());  
 DataOutputStream dOut = new DataOutputStream(socket.getOutputStream());  
 String message = dIn.readUTF();  
  
 System.out.println("Client Request : " + message);  
 String[] input = message.split(" ");   
 String result = input[0] + " " + input[2] + " " + input[1] + " = " + calculate(Integer.parseInt(input[0]), Integer.parseInt(input[1]), input[2]);  
 System.out.println("Server Response : " + result);  
  
 dOut.writeUTF(result);  
 dOut.flush();  
 dOut.close();  
 socket.close();  
 }catch(IOException e) {e.printStackTrace();}  
 }  
  
 public static String calculate (int num1, int num2, String operator) {  
 Integer result = 0;   
 switch (operator.charAt(0)){  
 case '+':  
 result = num1 + num2;  
 break;  
 case '-':  
 result = num1 - num2;  
 break;   
 case '\*':  
 result = num1 \* num2;  
 break;   
 case '/':  
 result = num1 / num2;  
 break;  
 }  
 return Integer.toString(result);  
   
 }  
}

***Android Client App: We had to make changes to the Manifest File  AndroidManifest.xml***

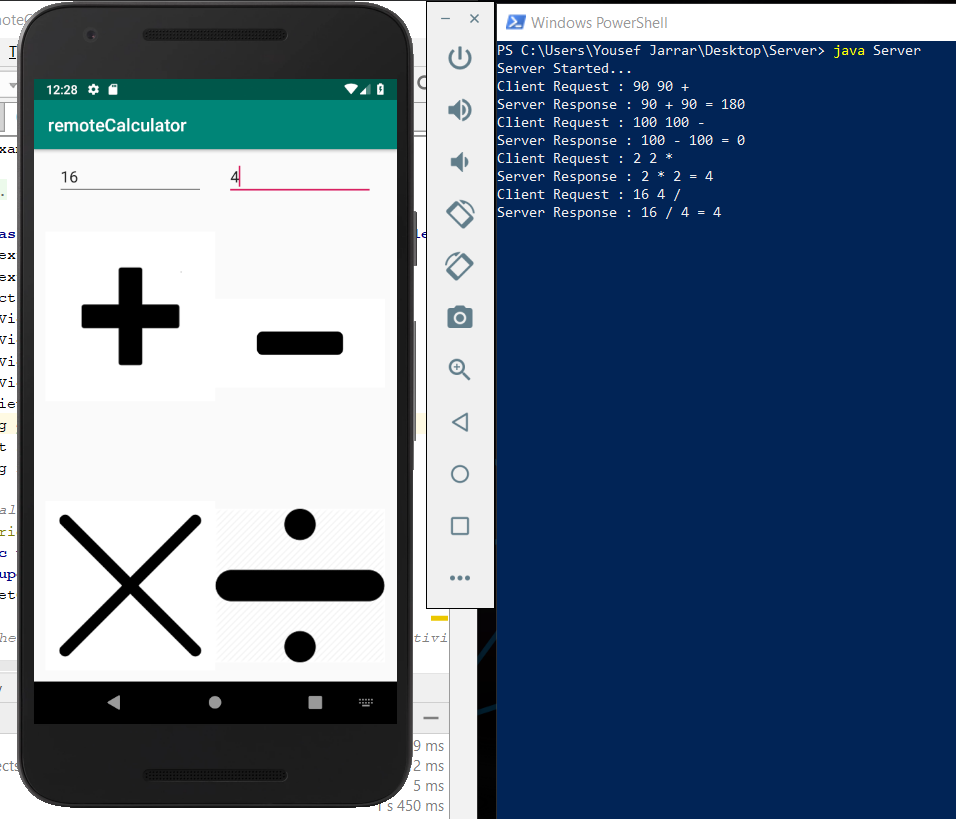


***MainActivity.java***

package example.rndnumberclient;  
  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.text.TextUtils;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.Button;  
import android.widget.TextView;  
import java.io.IOException;  
import java.net.Socket;  
import java.io.\*;  
import java.net.UnknownHostException;  
import java.lang.\*;  
  
public class MainActivity extends AppCompatActivity implements View.OnClickListener {  
  
 EditText n;  
 EditText max;  
 EditText min;  
 TextView displayResult;  
 Button submit;  
 MainActivity activity;  
 Socket socket;  
 String response = "";  
  
 */\*\* Called when first created. \*/* @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 // find the EditText elements (defined in res/layout/activity\_main.xml  
  
 n = (EditText) findViewById(R.id.editText);  
 min = (EditText) findViewById(R.id.editText2);  
 max = (EditText) findViewById(R.id.editText3);  
 submit = (Button) findViewById(R.id.button);  
 displayResult = (TextView) findViewById(R.id.displayResult);  
 // set listeners  
 submit.setOnClickListener(this);  
 }  
 // @Override  
 public void onClick( View view ) {  
 // check if the fields are empty  
 if (TextUtils.isEmpty(n.getText().toString())  
 || TextUtils.isEmpty(min.getText().toString()) ||  
 TextUtils.isEmpty(max.getText().toString())) {  
 return;  
 }  
 new Thread(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 response = "";  
 socket = new Socket("10.0.2.2", 4455);  
 DataOutputStream dOut = new  
 DataOutputStream(socket.getOutputStream());  
 DataInputStream dIn = new  
 DataInputStream(socket.getInputStream());  
 dOut.writeUTF(n.getText() + " " + min.getText() +  
 " " + max.getText());  
 dOut.flush();  
 response = dIn.readUTF();  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 displayResult.setText(response);  
 }  
 });  
 dIn.close();  
 dOut.close();  
 socket.close();  
 }  
 catch (UnknownHostException e) {  
 e.printStackTrace();  
 displayResult.setText("UnknownHostException: " + e.toString());  
 } catch (IOException e) {  
 e.printStackTrace();  
 displayResult.setText("IOException: " + e.toString());  
 }  
 }  
 }).start();  
 }  
  
}

***Activity\_Main.xml***

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent">  
<LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
<EditText  
 android:id="@+id/editText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="0.99"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:text="Enter Number" />  
<EditText  
 android:id="@+id/editText2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:text="LowerBound" />  
<EditText  
 android:id="@+id/editText3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:text="UpperBound" />  
<Button  
 android:id="@+id/button"  
 android:layout\_width="match\_parent"  
 android:layout\_height="46dp"  
 android:layout\_weight="1"  
 android:backgroundTint="@android:color/holo\_blue\_dark"  
 android:text="Submit" />  
<TextView  
 android:id="@+id/displayResult"  
  
 android:layout\_width="match\_parent"  
 android:layout\_height="116dp"  
 android:layout\_weight="1"  
 android:text="TextView" />  
</LinearLayout>  
</LinearLayout>

***Output:*** 

*Learning:*

*In section one of the lab we learned how to create a simple calculator and a server to communicate with the application. We were able to implement a basic calculator with its functions. Using remote java server program we successfully implemented RMI. Above is a sample of how the application contacted the server.*

1. **Remote Random Number Generator**

**RandomNumberServer.java**

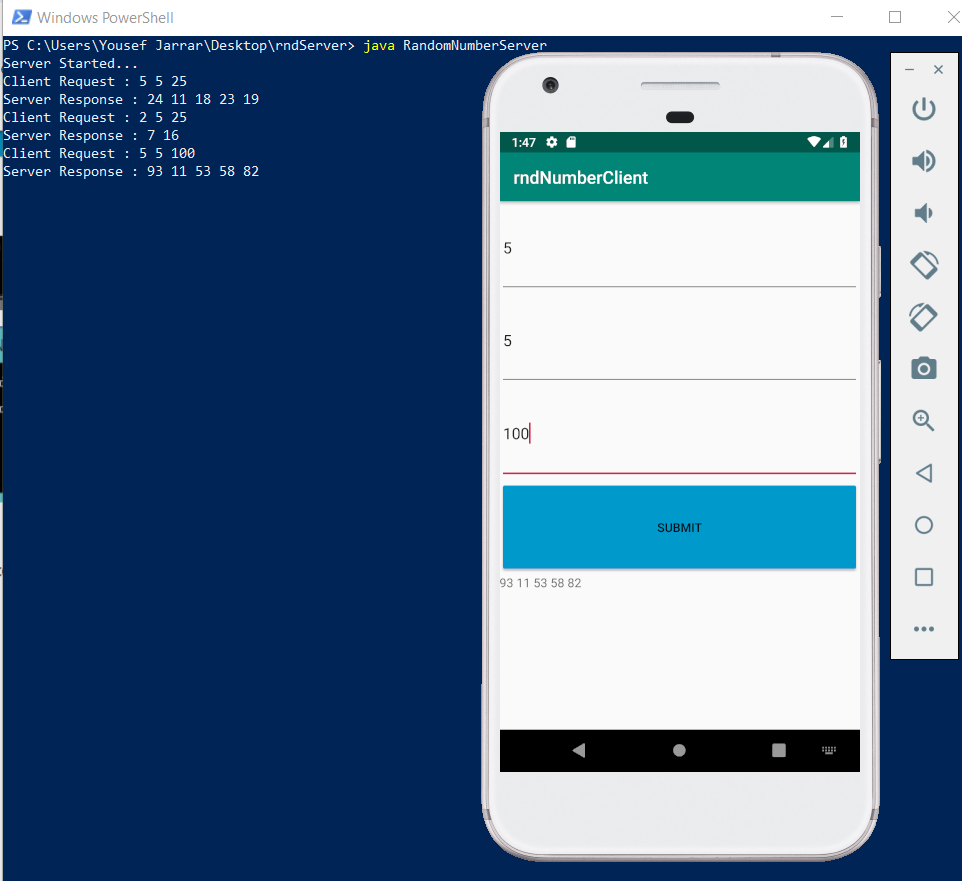
import java.io.IOException;  
import java.net.ServerSocket;  
import java.net.Socket;  
import java.util.Scanner;  
import java.lang.\*;  
import java.io.\*;  
import java.util.Random;  
import java.util.Arrays;  
  
public class RandomNumberServer {  
 public static void main(String[] args) {  
 try{  
 ServerSocket serverSocket = new ServerSocket(4444);  
 System.out.println("Server Started...");  
 while(true){  
 new Thread(new ClientConnection(serverSocket.accept())).start();  
 }  
 }catch(IOException e){e.printStackTrace();}  
 }  
}  
class ClientConnection implements Runnable{  
 private Socket socket;  
 public ClientConnection(Socket socket){  
 this.socket = socket;  
 }  
 @Override  
 public void run(){  
 try{  
 DataInputStream dIn = new DataInputStream(socket.getInputStream());  
 DataOutputStream dOut = new DataOutputStream(socket.getOutputStream());  
 String message = dIn.readUTF();  
 System.out.println("Client Request : " + message);  
 String[] input = message.split(" ");  
 String result = generateRandom(Integer.parseInt(input[0]),Integer.parseInt(input[1]), Integer.parseInt(input[2]));  
 System.out.println("Server Response : " + result);  
 dOut.writeUTF(result);  
 dOut.flush();  
 dOut.close();  
 socket.close();  
 }catch(IOException e){e.printStackTrace();}  
 }  
 public static String generateRandom(int num, int min, int max ) {  
 int range = (max - min) + 1;  
 String response ="";  
 for(int i=0;i<num;i++) {  
 response += Integer.toString((int)(Math.random() \* range) + min) + " ";  
 }  
 return response;  
 }  
}

**Client:**

**package** example.rndnumberclient;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.text.TextUtils;  
**import** android.view.View;  
**import** android.widget.EditText;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
**import** java.io.IOException;  
**import** java.net.Socket;  
**import** java.io.\*;  
**import** java.net.UnknownHostException;  
**import** java.lang.\*;  
  
**public class** MainActivity **extends** AppCompatActivity **implements** View.OnClickListener {  
  
 EditText **n**;  
 EditText **max**;  
 EditText **min**;  
 TextView **displayResult**;  
 Button **submit**;  
 MainActivity **activity**;  
 Socket **socket**;  
 String **response** = **""**;  
  
 */\*\* Called when first created. \*/* @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// find the EditText elements (defined in res/layout/activity\_main.xml* **n** = (EditText) findViewById(R.id.***editText***);  
 **min** = (EditText) findViewById(R.id.***editText2***);  
 **max** = (EditText) findViewById(R.id.***editText3***);  
 **submit** = (Button) findViewById(R.id.***button***);  
 **displayResult** = (TextView) findViewById(R.id.***displayResult***);  
 *// set listeners* **submit**.setOnClickListener(**this**);  
 }  
 *// @Override* **public void** onClick( View view ) {  
 *// check if the fields are empty* **if** (TextUtils.*isEmpty*(**n**.getText().toString())  
 || TextUtils.*isEmpty*(**min**.getText().toString()) ||  
 TextUtils.*isEmpty*(**max**.getText().toString())) {  
 **return**;  
 }  
 **new** Thread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **try** {  
 **response** = **""**;  
 **socket** = **new** Socket(**"10.0.2.2"**, 4455);  
 DataOutputStream dOut = **new** DataOutputStream(**socket**.getOutputStream());  
 DataInputStream dIn = **new** DataInputStream(**socket**.getInputStream());  
 dOut.writeUTF(**n**.getText() + **" "** + **min**.getText() +  
 **" "** + **max**.getText());  
 dOut.flush();  
 **response** = dIn.readUTF();  
 runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **displayResult**.setText(**response**);  
 }  
 });  
 dIn.close();  
 dOut.close();  
 **socket**.close();  
 }  
 **catch** (UnknownHostException e) {  
 e.printStackTrace();  
 **displayResult**.setText(**"UnknownHostException: "** + e.toString());  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **displayResult**.setText(**"IOException: "** + e.toString());  
 }  
 }  
 }).start();  
 }  
  
}

***UI.xml***

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"**>  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"**>  
 <**EditText  
 android:id="@+id/editText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="0.99"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:text="Enter Number"** />  
 <**EditText  
 android:id="@+id/editText2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:text="LowerBound"** />  
 <**EditText  
 android:id="@+id/editText3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:text="UpperBound"** />  
 <**Button  
 android:id="@+id/button"  
 android:layout\_width="match\_parent"  
 android:layout\_height="46dp"  
 android:layout\_weight="1"  
 android:backgroundTint="@android:color/holo\_blue\_dark"  
 android:text="Submit"** />  
 <**TextView  
 android:id="@+id/displayResult"  
  
 android:layout\_width="match\_parent"  
 android:layout\_height="116dp"  
 android:layout\_weight="1"  
 android:text="TextView"** />  
 </**LinearLayout**>  
</**LinearLayout**>



Outcome:

Jose and I were able to accomplish both sections of the lab. Our biggest difficulty was being able to understand how the client communicates with the server. We had to understand that the manifest of the app had to be changed, in order for it to communicate with the server. We also had to understand that based on RPC/RMI calls we had to include IP:PORT to it. Developing the application was not as hard as the last. I think we deserve full points on this lab. 20/20