INSTITUTE OF INFORMATION TECHNOLOGY

JAHANGIRNAGAR UNIVERSITY

Number of Assignment : 02

Submission Date : 21/10/2023

Course Tittle : Mobile Application Development Lab

Course Code : ICT - 4106

Submitted To

Dr. M. Mesbahuddin Sarker

Professor

IIT – JU

Submitted By

Md. Shakil Hossain

Roll – 2023

4th year 1st Semester

IIT – JU

Simple Calculator app using Android Studio

Android Components

The App components are the building blocks of Android. Each component has its own role and life cycles from launching of an app till the end. Some of these components depend upon others also. Each component has a definite purpose. The four major app components are:

- Activities
- Services
- Broadcast Receivers
- Content Provider

Activities: It deals with the UI and the user interactions to the screen. In other words, it is a User Interface that contains activities. These can be one or more depending upon the App. It starts when the application is launched. At least one activity is always present which is known as MainActivity. The activity is implemented through the following.

Syntax:

```
public class MainActivity extends Activity{
  // processes
}
```

Services: Services are the background actions performed by the app, these might be long-running operations like a user playing music while surfing the Internet. A service might need other sub-services so as to perform specific tasks. The main purpose of the Services is to provide non-stop working of the app without breaking any interaction with the user.

Syntax:

```
public class MyServices extends Services{
  // code for the services
}
```

Broadcast Receivers: A Broadcast is used to respond to messages from other applications or from the System. For example, when the battery of the phone is low, then the Android OS fires a Broadcasting message to launch the Battery Saver function or app, after receiving the message the appropriate action is taken by the app. Broadcast Receiver is the subclass of BroadcastReceiver class and each object is represented by Intent objects.

Syntax:

```
public class MyReceiver extends BroadcastReceiver{
  public void onReceive(context,intent){
}
```

Content Provider: Content Provider is used to transferring the data from one application to the others at the request of the other application. These are handled by the class ContentResolver class. This class implements a set of APIs(Application Programming Interface) that enables the other applications to perform the transactions. Any Content Provider must implement the Parent Class of ContentProvider class.

Syntax:

```
public class MyContentProvider extends ContentProvider{
  public void onCreate()
  {}
}
```

Manifest Folder: Android Manifest is an XML file that is the root of the project source set. It describes the essential information about the app and the Android build tools, the Android Operating System, and Google Play. It contains the permission that an app might need in order to perform a specific task. It also contains the Hardware and the Software features of the app, which determines the compatibility of an app on the Play Store. It also includes special activities like services, broadcast receiver, content providers, package name, etc.

Java Folder: The JAVA folder consists of the java files that are required to perform the background task of the app. It consists of the functionality of the buttons, calculation, storing, variables, toast programming function, etc. The number of these files depends upon the type of activities created.

Resource Folder: The res or Resource folder consists of the various resources that are used in the app. This consists of sub-folders like drawable, layout, mipmap, raw, and values. The drawable consists of the images. The layout consists of the XML files that define the user interface layout. These are stored in res.layout and are accessed as R.layout class. The raw consists of the Resources files like audio files or music files, etc. These are accessed through R.raw.filename. values are used to store the hardcoded strings(considered safe to store string values) values, integers, and colors. It consists of various other directories like:

R.array : arrays.xml for resource arrays

R.integer : integers.xml for resource integers

R.bool : bools.xml for resource boolean

R.color : colors.xml for color values

R.string : strings.xml for string values

R.dimen : dimens.xml for dimension values

R.style : styles.xml for styles

Gradle Files: Gradle is an advanced toolkit, which is used to manage the build process, that allows defining the flexible custom build configurations. Each build configuration can define its own set of code and resources while reusing the parts common to all versions of your app. The Android plugin for Gradle works with the build toolkit to provide processes and configurable settings that are specific to building and testing Android applications. Gradle and the Android plugin run independently of Android Studio. This means that you can build your Android apps from within Android Studio. The flexibility of the Android build system enables you to perform custom build configurations without modifying your app's core source files.

Basic Layout Can be defined in a tree structure as:

```
Project/
  app/
    manifest/
      AndroidManifest.xml
  java/
    MyActivity.java
    res/
      drawable/
```

icon.png

background.png

drawable-hdpi/

icon.png

background.png

layout/

activity_main.xml

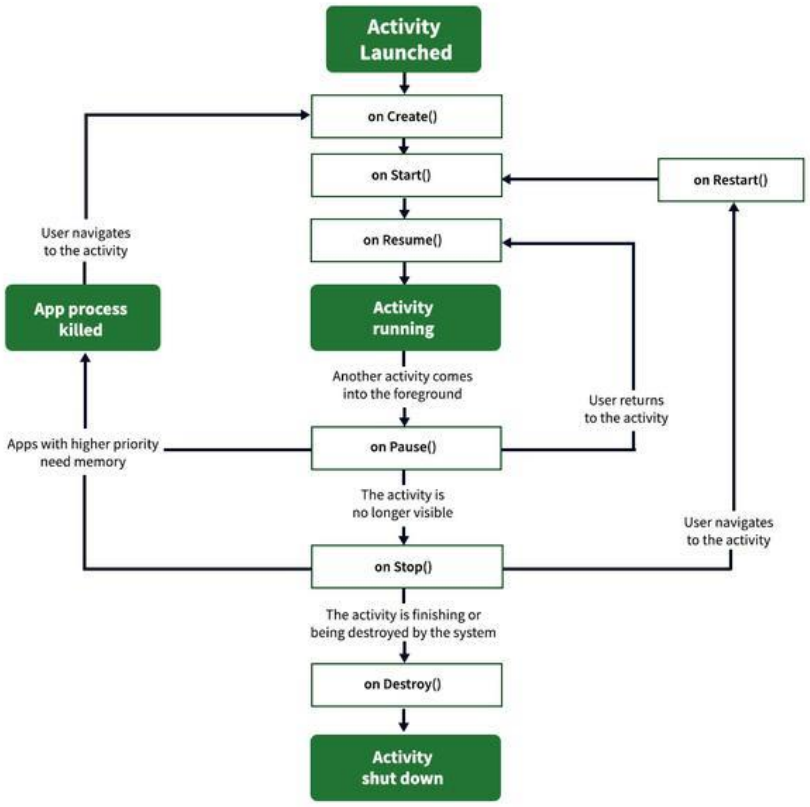
info.xml

values/

strings.xml

Lifecycle of Activity in Android App

The Lifecycle of Activity in Android App can be shown through this diagram:



Activity Lifecycle in Android

States of Android Lifecycle:

1. **OnCreate:** This is called when activity is first created.
2. **OnStart:** This is called when the activity becomes visible to the user.
3. **OnResume:** This is called when the activity starts to interact with the user.
4. **OnPause:** This is called when activity is not visible to the user.
5. **OnStop:** This is called when activity is no longer visible.
6. **OnRestart:** This is called when activity is stopped, and restarted again.
7. **OnDestroy:** This is called when activity is to be closed or destroyed.

Code: Working with the activity_main.xml file

Navigate to the **app > res > layout > activity_main.xml** and add the below code to that file. Below is the code for the **activity_main.xml** file.

XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#8BC34A"
    android:backgroundTint="@android:color/darker_gray"
    tools:context=".MainActivity">

    <!-- Text View to display our basic heading of "calculator"-->
    <TextView
        android:layout_width="194dp"
        android:layout_height="43dp"
```

```
    android:layout_marginStart="114dp"
    android:layout_marginLeft="114dp"
    android:layout_marginTop="58dp"
    android:layout_marginEnd="103dp"
    android:layout_marginRight="103dp"
    android:layout_marginBottom="502dp"
    android:scrollbarSize="30dp"
    android:text=" Calculator"
    android:textAppearance="@style/TextAppearance.AppCompat.Body1"
    android:textSize="30dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<!-- Edit Text View to input the values -->
<EditText
    android:id="@+id/num1"
    android:layout_width="364dp"
    android:layout_height="28dp"
    android:layout_marginStart="72dp"
    android:layout_marginTop="70dp"
    android:layout_marginEnd="71dp"
    android:layout_marginBottom="416dp"
    android:background="@android:color/white"
    android:ems="10"
    android:onClick="clearTextNum1"
    android:inputType="number"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<!-- Edit Text View to input 2nd value-->
<EditText
    android:id="@+id/num2"
    android:layout_width="363dp"
    android:layout_height="30dp"
    android:layout_marginStart="72dp"
    android:layout_marginTop="112dp"
    android:layout_marginEnd="71dp"
    android:layout_marginBottom="374dp"
    android:background="@android:color/white"
    android:ems="10"
    android:onClick="clearTextNum2"
    android:inputType="number"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<!-- Text View to display result -->
<TextView
    android:id="@+id/result"
    android:layout_width="356dp"
    android:layout_height="71dp"
```

```
    android:layout_marginStart="41dp"
    android:layout_marginTop="151dp"
    android:layout_marginEnd="48dp"
    android:layout_marginBottom="287dp"
    android:background="@android:color/white"
    android:text="result"
    android:textColorLink="#673AB7"
    android:textSize="25sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<!-- A button to perform 'sum' operation -->
<Button
    android:id="@+id/sum"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="292dp"
    android:layout_marginEnd="307dp"
    android:layout_marginBottom="263dp"
    android:backgroundTint="@android:color/holo_red_light"
    android:onClick="doSum"
    android:text="+"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
```

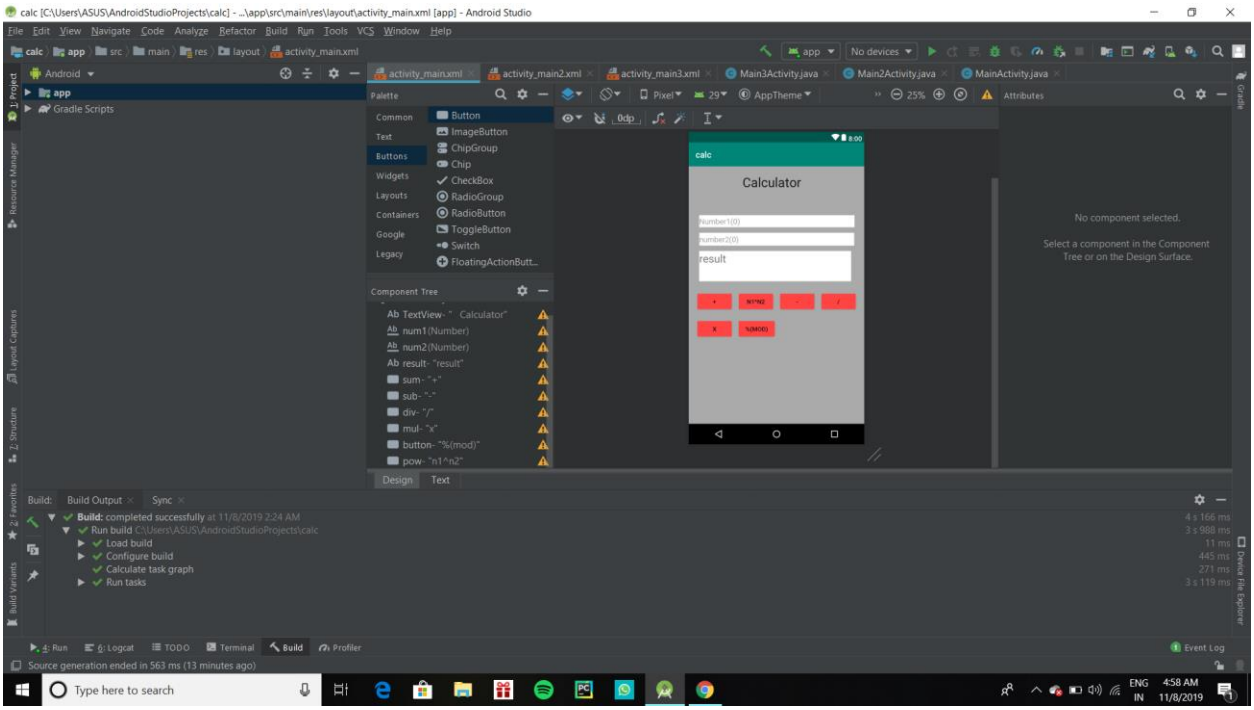
```
    app:layout_constraintTop_toTopOf="parent" />

<!-- A button to perform subtraction operation. -->

<!-- A button to perform division. -->

<Button
    android:id="@+id/sub"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="210dp"
    android:layout_marginTop="292dp"
    android:layout_marginEnd="113dp"
    android:layout_marginBottom="263dp"
    android:backgroundTint="@android:color/holo_red_light"
    android:onClick="doSub"
    android:text="-"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.507" />

<Button
    android:id="@+id/div"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
```

```
    android:layout_marginStart="307dp"
    android:layout_marginTop="292dp"
    android:layout_marginEnd="16dp"
    android:layout_marginBottom="263dp"
    android:backgroundTint="@android:color/holo_red_light"
    android:onClick="doDiv"
    android:text="/"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<!-- A button to perform multiplication. -->
<Button
    android:id="@+id/mul"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="356dp"
    android:layout_marginEnd="307dp"
    android:layout_marginBottom="199dp"
    android:backgroundTint="@android:color/holo_red_light"
    android:onClick="doMul"
    android:text="x"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
```

```
    app:layout_constraintTop_toTopOf="parent" />

<!-- A button to perform a modulus function. -->

<!-- A button to perform a power function. -->

<Button
    android:id="@+id/button"
    android:layout_width="103dp"
    android:layout_height="46dp"
    android:layout_marginStart="113dp"
    android:layout_marginTop="356dp"
    android:layout_marginEnd="206dp"
    android:layout_marginBottom="199dp"
    android:backgroundTint="@android:color/holo_red_light"
    android:onClick="doMod"
    android:text="%(mod)"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.515" />

<Button
    android:id="@+id/pow"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="113dp"
```

```
android:layout_marginTop="292dp"
android:layout_marginEnd="210dp"
android:layout_marginBottom="263dp"
android:backgroundTint="@android:color/holo_red_light"
android:onClick="doPow"
android:text="n1^n2"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.0"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.507" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

After using this code the UI will be like as follows:



Working with the MainActivity.java file

Open the **MainActivity.java** file there within the class, and make a method named doSum(View v). In this method, first of all, we have to link two EditText with variables so that we can use them for our input. So link those edit box with variables we have written

"EditText e1=(EditText)findViewById(R.id.num1);"

Here num1 is id for the textbox and we are just giving a variable name ‘e1’ to text box with id ‘num1’. Similarly, we have to use the same statement for the second textbox with the variable name ‘e2’. For the third text box, we have used

"TextView t1=(TextView) findViewById(R.id.result);"

Here we have used TextView because we only have to display text avoiding it being user-changeable. Now we have to input numbers in form of string using the **getText()** function. The input statement will be

"String s11=e1.getText().toString();"

Here s11 stores the number entered in textbox 1. We have to do the same with another Textbox(e2). Now store the number in int form and apply addition. store the added value in another variable. To display stored in sum we have to use setText() as follows:

result.setText(final_sum.toString())

final_sum stores the sum and it’s necessary to convert it to string(.toString()). Below is the code for the **MainActivity.java** file. Comments are added inside the code to understand the code in more detail.

File: MainActivity.java

Java

```
package com.example.calculator2;

import android.os.Bundle;

import com.google.android.material.snackbar.Snackbar;

import androidx.appcompat.app.AppCompatActivity;

import android.text.TextUtils;

import android.view.View;

import androidx.navigation.NavController;

import androidx.navigation.Navigation;

import androidx.navigation.ui.AppBarConfiguration;
```

```
import androidx.navigation.ui.NavigationUI;
import com.example.calculator2.databinding.ActivityMainBinding;
import android.view.Menu;
import android.view.MenuItem;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
    private AppBarConfiguration appBarConfiguration;
    private ActivityMainBinding binding;
    public EditText e1, e2;
    TextView t1;
    int num1, num2;
    public boolean getNumbers() {
        //checkAndClear();
        // defining the edit text 1 to e1
        e1 = (EditText) findViewById(R.id.num1);
        // defining the edit text 2 to e2
        e2 = (EditText) findViewById(R.id.num2);
        // defining the text view to t1
        t1 = (TextView) findViewById(R.id.result);
        // taking input from text box 1
        String s1 = e1.getText().toString();
        // taking input from text box 2
        String s2 = e2.getText().toString();
```

```
 if(s1.equals("Please enter value 1") && s2.equals(null))
 {
     String result = "Please enter value 2";
     e2.setText(result);
     return false;
 }
if(s1.equals(null) && s2.equals("Please enter value 2"))
{
    String result = "Please enter value 1";
    e1.setText(result);
    return false;
}
if(s1.equals("Please enter value 1") || s2.equals("Please enter value 2"))
{
    return false;
}
 if((!s1.equals(null) && s2.equals(null))|| (!s1.equals("") && s2.equals("")) ){

    String result = "Please enter value 2";

    e2.setText(result);
    return false;
}
if((s1.equals(null) && !s2.equals(null))|| (s1.equals("") && !s2.equals("")) ){
    //checkAndClear();
    String result = "Please enter value 1";
    e1.setText(result);
```

```
        return false;
    }
    if((s1.equals(null) && s2.equals(null))|| (s1.equals("") && s2.equals("")) ){
        //checkAndClear();
        String result1 = "Please enter value 1";
        e1.setText(result1);
        String result2 = "Please enter value 2";
        e2.setText(result2);
        return false;
    }
    else {
        // converting string to int.
        num1 = Integer.parseInt(s1);
        // converting string to int.
        num2 = Integer.parseInt(s2);
    }
    return true;
}
public void doSum(View v) {
    // get the input numbers
    if (getNumbers()) {
        int sum = num1 + num2;
        t1.setText(Integer.toString(sum));
    }
   else
    {
        t1.setText("Error Please enter Required Values");
    }
```

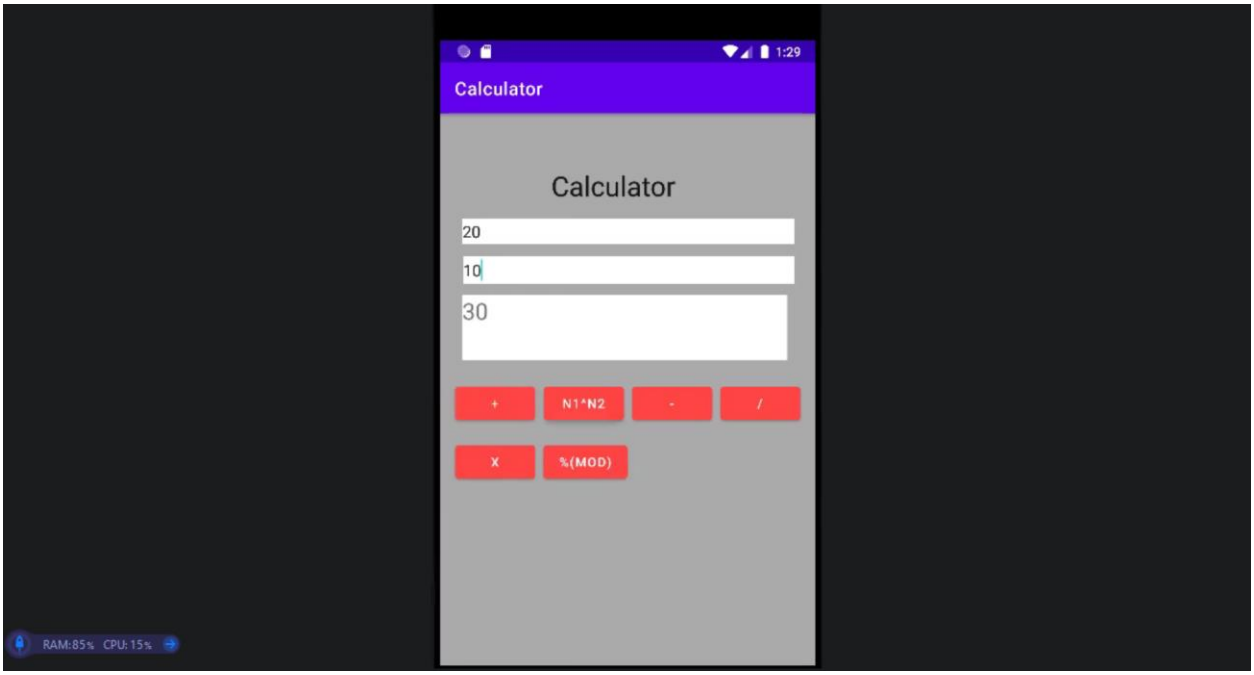
```
}
public void clearTextNum1(View v) {
    // get the input numbers
  e1.getText().clear();
}
public void clearTextNum2(View v) {
    // get the input numbers
    e2.getText().clear();
}
public void doPow(View v) {
    //checkAndClear();
    // get the input numbers
    if (getNumbers()) {
        double sum = Math.pow(num1, num2);
        t1.setText(Double.toString(sum));
    }
    else
    {
        t1.setText("Error Please enter Required Values");
    }
}
// a public method to perform subtraction
public void doSub(View v) {
    //checkAndClear();
    // get the input numbers
    if (getNumbers()) {
        int sum = num1 - num2;
        t1.setText(Integer.toString(sum));
```

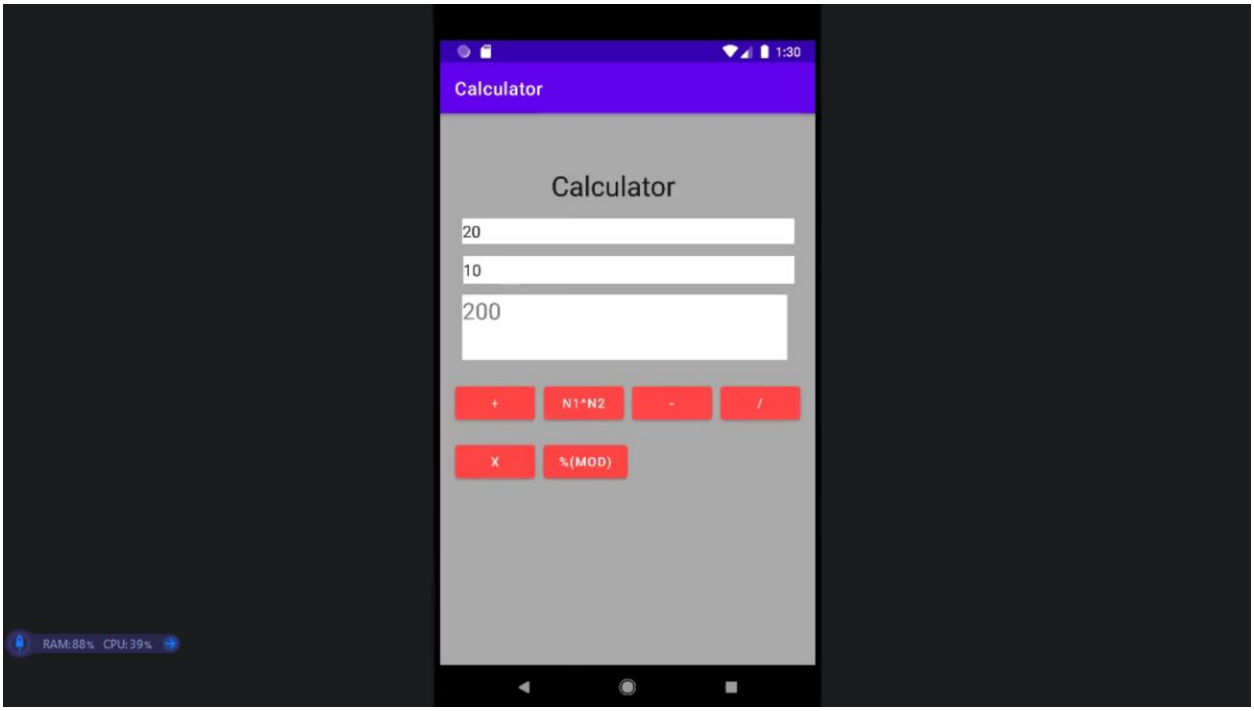
```
    }
    else
    {
        t1.setText("Error Please enter Required Values");
    }
}
// a public method to perform multiplication
public void doMul(View v) {
    //checkAndClear();
    // get the input numbers
    if (getNumbers()) {
        int sum = num1 * num2;
        t1.setText(Integer.toString(sum));
    }
    else
    {
        t1.setText("Error Please enter Required Values");
    }
}
// a public method to perform Division
public void doDiv(View v) {
    //checkAndClear();
    // get the input numbers
    if (getNumbers()) {
        // displaying the text in text view assigned as t1
        double sum = num1 / (num2 * 1.0);
        t1.setText(Double.toString(sum));
    }
```

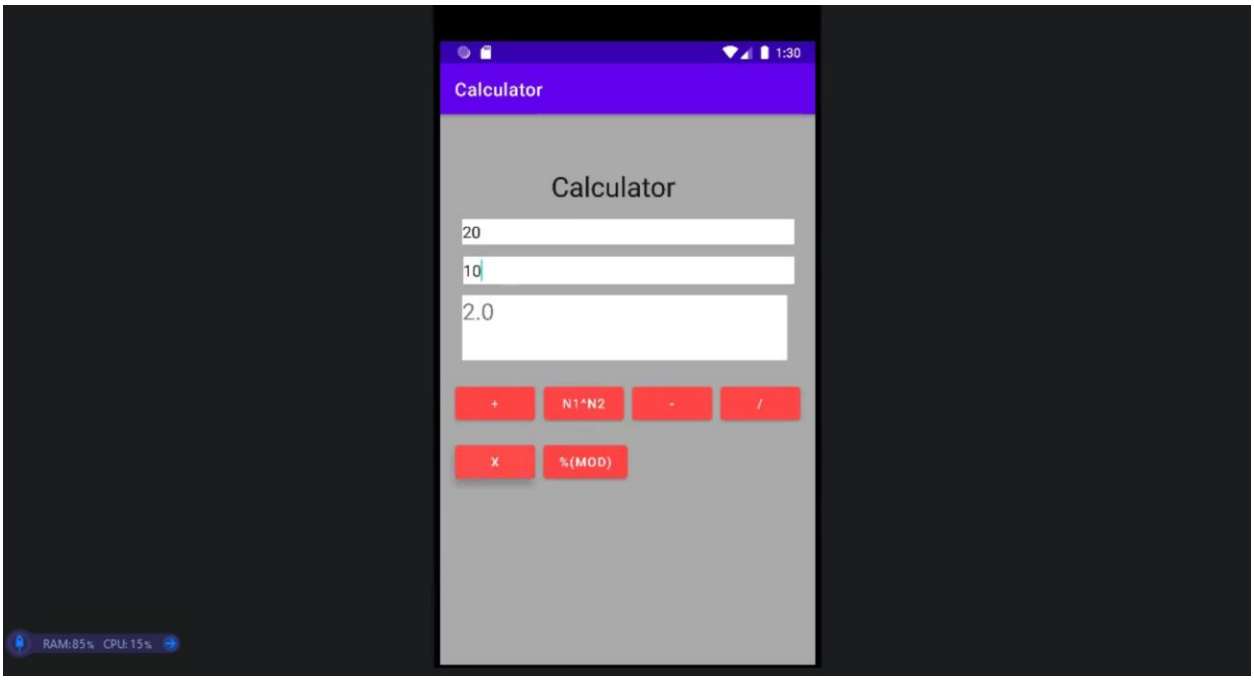
```
    else
    {
        t1.setText("Error Please enter Required Values");
    }
}
// a public method to perform modulus function
public void doMod(View v) {
    //checkAndClear();
    // get the input numbers
    if (getNumbers()) {
        double sum = num1 % num2;
        t1.setText(Double.toString(sum));
    }
    else
    {
        t1.setText("Error Please enter Required Values");
    }
}
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    e1 = (EditText) findViewById(R.id.num1);
    // defining the edit text 2 to e2
    e2 = (EditText) findViewById(R.id.num2);
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
```

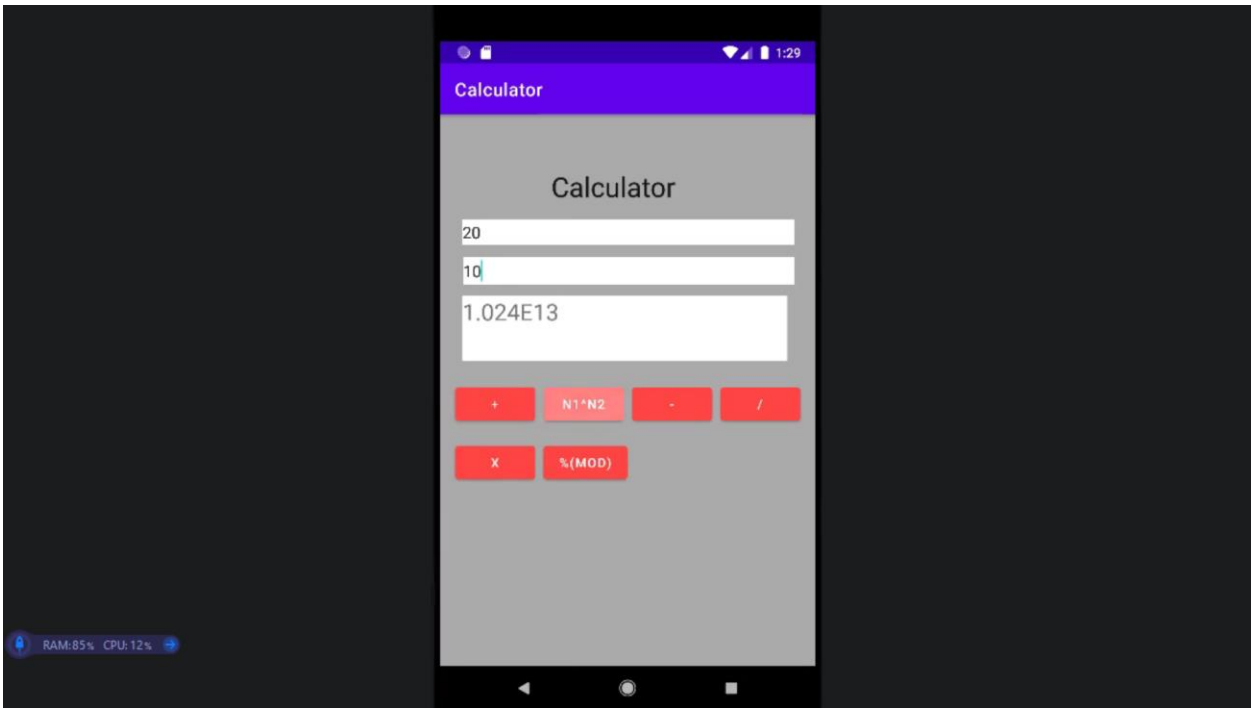
```
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
    @Override
    public boolean onSupportNavigateUp() {
        NavController navController = Navigation.findNavController(this,
R.id.nav_host_fragment_content_main);
        return NavigationUI.navigateUp(navController, appBarConfiguration)
                || super.onSupportNavigateUp();
    }
}
```

Output


1:29
Calculator
Calculator
20
10
30
+
N1^N2
-
/
X
%(MOD)
RAM:85% CPU:15%


1:30
Calculator
Calculator
20
10
200
+
N1^N2
-
/
X
%(MOD)
RAM:88% CPU:39%


1:30
Calculator
Calculator
20
10
2.0
+
N1*N2
-
/
X
%(MOD)
RAM:85% CPU:15%


1:29
Calculator
Calculator
20
10
1.024E13
+
N1*N2
-
/
X
%(MOD)
RAM:85% CPU:12%