**Assignment No: 5**

1. **TITLE**

A Mobile App for Calculator having Trigonometry functionality is to be designed and tested. The data storage uses 1.text files, 2. XML Use latest open source software modeling, Designing and testing tool/Scrum-it. Implement the design using HTML-5/Scala/Python/

Java/C++/Rubi on Rails. Perform Positive and Negative testing.

**PREREQUISITES**

Android Studio/adt-bundle-windows

Testing tool

JAVA, XML

**OBJECTIVE**

To study testing tool.

To perform Positive and Negative testing.

**THEORY**

**Android Studio Overview**

Android Studio is the official IDE for Android application development, based on  [IntelliJ IDEA](https://www.jetbrains.com/idea/). On top of the capabilities you expect from IntelliJ, Android Studio offers:

Flexible Gradle-based build system

Build variants and multiple apk file generation

Code templates to help you build common app features

Rich layout editor with support for drag and drop theme editing

lint tools to catch performance, usability, version compatibility, and other problems

Pro Guard and app-signing capabilities

Built-in support for  [Google Cloud Platform,](http://developers.google.com/cloud/devtools/android_studio_templates/) making it easy to integrate Google Cloud

Messaging and App Engine

And much more

**Android Project Structure**

By default, Android Studio displays your project files in the Android project view. This view shows a flattened version of your project's structure that provides quick access to the key source files of Android projects and helps you work with the  [Gradle-based build system.](http://developer.android.com/sdk/installing/studio-build.html) The Android project view:

Shows the most important source directories at the top level of the module hierarchy.

Groups the build files for all modules in a common folder.

Groups all the manifest files for each module in a common folder.

Shows resource files from all Gradle source sets.

Groups resource files for different locales, orientations, and screen types in a single group per resource type

java/ - Source files for the module. manifests/ - Manifest files for the module. res/ - Resource files for the module.

Gradle Scripts/ - Gradle build and property files.

Software testing is process of verifying and validating the software or application and checks whether it is working as expected. The intent is to find defects and improve the product quality. There are two ways to test the software viz, Positive Testing and Negative Testing.

Positive testing can be performed on the system by providing the valid data as input. It checks whether an application behaves as expected with the positive input. This is to test to check the application that does what it is supposed to do so. There is a text box in an application which can accept only numbers. Entering values up to 99999 will be acceptable by the system and any other values apart from this should not be acceptable. To do positive testing, set the valid input values from 0 to 99999 and check whether the system is accepting the values.

Negative Testing can be performed on the system by providing invalid data as input. It checks whether an application behaves as expected with the negative input. This is to test the application that does not do anything that it is not supposed to do so. For example - Negative testing can be performed by testing by entering alphabets characters from A to Z or from a to z. Either system text box should not accept the values or else it should throw an error message for these invalid data inputs.

**Positive Testing:**

Test

Expected Result

Actual Result

Case ID

1

Check if all the numbers are

working ( 0 to 9)

All the numbers are working ( 0 to 9)

2

Check if the arithmetic keys ( +,-,\*, %, /) are working

The arithmetic keys ( +, -, \*, %, /) are working

3

Check if the brackets keys are working

The bracket keys are working

4

Check if the square and square

root key is working

The square and square root key is working

5

Check if the sin, cos, tan, cot keys are working

The sin, cos, tan, cot keys are working

6

Check if it is showing the correct values for sin, cos, tan and cot

It is showing the correct values for sin, cos, tan and cot

7

Check the addition of two sin and cos values

The addition of two sin and cos values

8

Check the addition of two tan and cot values

The addition of two tan and cot values

9

Check that it is returning the float values or integer values

It is returning the float values or integer values

**Negative Testing:**

Test

Expected Result

Actual Result

Case ID

1

Check if it is allowing letters

It is taking only numbers as input

instead of numbers

2

Check if it is returning float

It is returning integer values only

values instead of integer

3

Check if it is returning integer

It is returning float values only

values instead of float

4

Check if the functionality using

Functioning Properly

BODMAS/BIDMAS works as

expected

5. **MATHEMATICAL MODEL**

Let, S be the System Such that,

A={ S, E, I,O, F, DD, NDD, success,failure}

Where,

S= Start state, E= End State, I= Set of Input

O= Set of Output

F =Set of Function

DD=Deterministic Data NDD=Non Deterministic Data

Success Case: It is the case when all the inputs are given by system are entered correctly. Failure Case: It is the case when the input does not match the validation Criteria.

**CONCLUSION**

A Mobile App for Calculator having Trigonometry functionality is designed and tested.

Code:

package com.example.banty.calculator;

import android.app.Activity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

import java.text.DecimalFormat;

public class MainActivity extends Activity implements View.OnClickListener{

private TextView display = null;

private boolean userStillTyping = false;

private Calculator calc;

private static final String DIGITS = "0123456789.";

DecimalFormat df = new DecimalFormat("@###########");

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

calc = new Calculator();

display = (TextView)findViewById(R.id.textView1);

df.setMinimumFractionDigits(0);

df.setMinimumIntegerDigits(1);

df.setMaximumIntegerDigits(8);

findViewById(R.id.button0).setOnClickListener(this);

findViewById(R.id.button1).setOnClickListener(this);

findViewById(R.id.button2).setOnClickListener(this);

findViewById(R.id.button3).setOnClickListener(this);

findViewById(R.id.button4).setOnClickListener(this);

findViewById(R.id.button5).setOnClickListener(this);

findViewById(R.id.button6).setOnClickListener(this);

findViewById(R.id.button7).setOnClickListener(this);

findViewById(R.id.button8).setOnClickListener(this);

findViewById(R.id.button9).setOnClickListener(this);

findViewById(R.id.buttonAdd).setOnClickListener(this);

findViewById(R.id.buttonSubtract).setOnClickListener(this);

findViewById(R.id.buttonMultiply).setOnClickListener(this);

findViewById(R.id.buttonDivide).setOnClickListener(this);

findViewById(R.id.buttonToggleSign).setOnClickListener(this);

findViewById(R.id.buttonDecimalPoint).setOnClickListener(this);

findViewById(R.id.buttonEquals).setOnClickListener(this);

findViewById(R.id.buttonClear).setOnClickListener(this);

findViewById(R.id.buttonClearMemory).setOnClickListener(this);

findViewById(R.id.buttonAddToMemory).setOnClickListener(this);

findViewById(R.id.buttonSubtractFromMemory).setOnClickListener(this);

findViewById(R.id.buttonRecallMemory).setOnClickListener(this);

// The following buttons only exist in layout-land (Landscape mode) and require extra attention.

// The messier option is to place the buttons in the regular layout too and set android:visibility="invisible".

if (findViewById(R.id.buttonSquareRoot) != null) {

findViewById(R.id.buttonSquareRoot).setOnClickListener(this);

}

if (findViewById(R.id.buttonSquared) != null) {

findViewById(R.id.buttonSquared).setOnClickListener(this);

}

if (findViewById(R.id.buttonInvert) != null) {

findViewById(R.id.buttonInvert).setOnClickListener(this);

}

if (findViewById(R.id.buttonSine) != null) {

findViewById(R.id.buttonSine).setOnClickListener(this);

}

if (findViewById(R.id.buttonCosine) != null) {

findViewById(R.id.buttonCosine).setOnClickListener(this);

}

if (findViewById(R.id.buttonTangent) != null) {

findViewById(R.id.buttonTangent).setOnClickListener(this);

}

}

@Override

public void onClick(View v) {

String buttonPressed = ((Button) v).getText().toString();

if(DIGITS.contains(buttonPressed)){

//a digit button is pressed

if(userStillTyping){

if(buttonPressed.equals(".") &&

display.getText().toString().contains(".")){

//user pressed the decimal button and the textview already

// contains a decimal so ignore

}else{

display.append(buttonPressed);

}

}else{

if(buttonPressed.equals(".")){

display.setText(0+buttonPressed);

}else{

display.setText(buttonPressed);

}

userStillTyping = true;

}

}else{

//an operation button is pressed

if(userStillTyping) {

calc.setOperand(Double.parseDouble(display.getText().toString()));

userStillTyping = false;

}

calc.performOperation(buttonPressed);

display.setText(df.format(calc.getResult()));

}

}

@Override

protected void onSaveInstanceState(Bundle outState) {

super.onSaveInstanceState(outState);

// Save variables on screen orientation change

outState.putDouble("OPERAND", calc.getResult());

outState.putDouble("MEMORY", calc.getMemory());

}

@Override

protected void onRestoreInstanceState(Bundle savedInstanceState) {

super.onRestoreInstanceState(savedInstanceState);

// Restore variables on screen orientation change

calc.setOperand(savedInstanceState.getDouble("OPERAND"));

calc.setMemory(savedInstanceState.getDouble("MEMORY"));

display.setText(df.format(calc.getResult()));

}

}

Activity\_main.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/functionPad"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_gravity="center"

android:orientation="vertical"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin" >

<LinearLayout

android:id="@+id/row1"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".12" >

<TextView

android:id="@+id/textView1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="right"

android:maxLines="1"

android:paddingLeft="10dp"

android:paddingRight="10dp"

android:text="0"

android:textAppearance="?android:attr/textAppearanceLarge"

android:textSize="40sp" />

</LinearLayout>

<LinearLayout

android:id="@+id/row2"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".12" >

<Button

android:id="@+id/buttonClearMemory"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonClearMemory"

android:textSize="25sp" />

<Button

android:id="@+id/buttonAddToMemory"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonAddToMemory"

android:textSize="25sp" />

<Button

android:id="@+id/buttonSubtractFromMemory"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonSubtractFromMemory"

android:textSize="25sp" />

<Button

android:id="@+id/buttonRecallMemory"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonRecallMemory"

android:textSize="25sp" />

</LinearLayout>

<LinearLayout

android:id="@+id/row3"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".12" >

<Button

android:id="@+id/buttonClear"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonClear"

android:textSize="25sp" />

<Button

android:id="@+id/buttonToggleSign"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonToggleSign"

android:textSize="25sp" />

<Button

android:id="@+id/buttonDivide"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonDivide"

android:textSize="25sp" />

<Button

android:id="@+id/buttonMultiply"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonMultiply"

android:textSize="25sp" />

</LinearLayout>

<LinearLayout

android:id="@+id/row4"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".12" >

<Button

android:id="@+id/button7"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/button7"

android:textSize="25sp" />

<Button

android:id="@+id/button8"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/button8"

android:textSize="25sp" />

<Button

android:id="@+id/button9"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/button9"

android:textSize="25sp" />

<Button

android:id="@+id/buttonSubtract"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonSubtract"

android:textSize="25sp" />

</LinearLayout>

<LinearLayout

android:id="@+id/row5"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".12" >

<Button

android:id="@+id/button4"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/button4"

android:textSize="25sp" />

<Button

android:id="@+id/button5"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/button5"

android:textSize="25sp" />

<Button

android:id="@+id/button6"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/button6"

android:textSize="25sp" />

<Button

android:id="@+id/buttonAdd"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonAdd"

android:textSize="25sp" />

</LinearLayout>

<LinearLayout

android:id="@+id/row6"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".24"

android:baselineAligned="false" >

<LinearLayout

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".75"

android:orientation="vertical" >

<LinearLayout

android:id="@+id/linearLayout1"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".50"

android:textSize="25sp" >

<Button

android:id="@+id/button1"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".33"

android:text="@string/button1"

android:textSize="25sp" />

<Button

android:id="@+id/button2"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".33"

android:text="@string/button2"

android:textSize="25sp" />

<Button

android:id="@+id/button3"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".34"

android:text="@string/button3"

android:textSize="25sp" />

</LinearLayout>

<LinearLayout

android:id="@+id/linearLayout2"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight=".50" >

<Button

android:id="@+id/button0"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".66"

android:text="@string/button0"

android:textSize="25sp" />

<Button

android:id="@+id/buttonDecimalPoint"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".34"

android:text="@string/buttonDecimalPoint"

android:textSize="25sp" />

</LinearLayout>

</LinearLayout>

<Button

android:id="@+id/buttonEquals"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight=".25"

android:text="@string/buttonEquals"

android:textSize="25sp" />

</LinearLayout>

</LinearLayout>

Output:





