Assignment No: A5

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.

2. PREREQUISITES

- Android Studio/adt-bundle-windows
- Testing tool
- JAVA, XML

3. OBJECTIVE

- To study testing tool.
- To perform Positive and Negative testing.

4. THEORY

Android Studio Overview

Android Studio is the official IDE for Android application development, based on IntelliJ 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, 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. 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 Case ID	Expected Result	Actual Result	Status
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	
10	Check if the functionality using BODMAS/BIDMAS works as expected	Working Properly	

Negative Testing:

Test Case ID	Expected Result	Actual Result	Status
1	Check if it is allowing letters instead of numbers	It is taking only numbers as input	
2	Check if it is returning float values instead of integer	It is returning integer values only	
3	Check if it is returning integer values instead of float	It is returning float values only	
4	Check if the functionality using BODMAS/BIDMAS works as expected	Functioning Properly	

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 Out put
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.

6. CONCLUSION

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

MainActivity.java

```
package example.com.calculator;
import java.io.IOException;
import java.text.DecimalFormat;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.TextView;
/**
* Created by -- on 19-03-2016.
*/
public class MainActivity extends Activity implements OnClickListener {
private TextView mCalculatorDisplay;
private Boolean userIsInTheMiddleOfTypingANumber = false;
private CalculatorBrain mCalculatorBrain;
private static final String DIGITS = "0123456789.";
   DecimalFormat df = new DecimalFormat("@##########");
   @SuppressLint("NewApi")
@Override
protected void onCreate(Bundle savedInstanceState) {
        // hide the window title.
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        // hide the status bar and other OS-level chrome
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mCalculatorBrain = new CalculatorBrain();
        mCalculatorDisplay = (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);
        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)) {
        // digit was pressed
        if (userIsInTheMiddleOfTypingANumber) {
                if (buttonPressed.equals(".") &&
mCalculatorDisplay.getText().toString().contains(".")) {
                    // ERROR PREVENTION
                    // Eliminate entering multiple decimals
                } else {
                    mCalculatorDisplay.append(buttonPressed);
                   }
        } else {
                if (buttonPressed.equals(".")) {
                    // ERROR PREVENTION
                    // This will avoid error if only the decimal is hit before an operator, by
placing a leading zero
                    // before the decimal
                    mCalculatorDisplay.setText(0 + buttonPressed);
                } else {
                    mCalculatorDisplay.setText(buttonPressed);
                }
                userIsInTheMiddleOfTypingANumber = true;
        }
        } else {
        // operation was pressed
        if (userIsInTheMiddleOfTypingANumber) {
                mCalculatorBrain.setOperand(Double.parseDouble(mCalculatorDisplay.getText().toS
tring()));
                userIsInTheMiddleOfTypingANumber = false;
        }
        try {
                mCalculatorBrain.performOperation(buttonPressed);
        } catch (IOException e) {
                e.printStackTrace();
        }
            mCalculatorDisplay.setText(df.format(mCalculatorBrain.getResult()));
        }
}
```

```
@Override
protected void onSaveInstanceState(Bundle outState) {
       super.onSaveInstanceState(outState);
       // Save variables on screen orientation change
       outState.putDouble("OPERAND", mCalculatorBrain.getResult());
       outState.putDouble("MEMORY", mCalculatorBrain.getMemory());
}
@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
       super.onRestoreInstanceState(savedInstanceState);
       // Restore variables on screen orientation change
       mCalculatorBrain.setOperand(savedInstanceState.getDouble("OPERAND"));
       mCalculatorBrain.setMemory(savedInstanceState.getDouble("MEMORY"));
       mCalculatorDisplay.setText(df.format(mCalculatorBrain.getResult()));
  }
}
```

CalculatorBrain.java

```
package example.com.calculator;
import android.content.Context;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
/**
* Created by -- on 19-03-2016.
*/
public class CalculatorBrain {
// 3 + 6 = 9
// 3 & 6 are called the operand.
// The + is called the operator.
// 9 is the result of the operation.
private double mOperand;
private double mWaitingOperand;
private String mWaitingOperator;
private double mCalculatorMemory;
// operator types
public static final String ADD = "+";
public static final String SUBTRACT = "-";
public static final String MULTIPLY = "*";
public static final String DIVIDE = "/";
public static final String CLEAR = "C" ;
public static final String CLEARMEMORY = "MC";
public static final String ADDTOMEMORY = "M+";
public static final String SUBTRACTFROMMEMORY = "M-";
public static final String RECALLMEMORY = "MR";
public static final String SQUAREROOT = "√";
public static final String SQUARED = "x²";
public static final String INVERT = "1/x";
public static final String TOGGLESIGN = "+/-";
public static final String SINE = "sin";
public static final String COSINE = "cos";
public static final String TANGENT = "tan";
// public static final String EQUALS = "=";
// constructor
public CalculatorBrain() {
       // initialize variables upon start
       mOperand = 0;
       mWaitingOperand = 0;
       mWaitingOperator = "";
```

```
        mCalculatorMemory = 0;
    }
    public void setOperand(double operand) {
        mOperand = operand;
    }
    public double getResult() {
        return mOperand;
    }
    // used on screen orientation change
    public void setMemory(double calculatorMemory) {
        mCalculatorMemory = calculatorMemory;
    }
    // used on screen orientation change
    public double getMemory() {
        return mCalculatorMemory;
    }
    public String toString() {
        return Double.toString(mOperand);
    }
    protected double performOperation(String operator) throws IOException {
        if (operator.equals(CLEAR)) {
            mOperand = 0;
            mWaitingOperator = "";
            mWaitingOperand = 0;
        // mCalculatorMemory = 0;
        } else if (operator.equals(CLEARMEMORY)) {
            mCalculatorMemory = 0;
        } else if (operator.equals(ADDTOMEMORY)) {
            mCalculatorMemory = mCalculatorMemory + mOperand;
        } else if (operator.equals(SUBTRACTFROMMEMORY)) {
            mCalculatorMemory = mCalculatorMemory - mOperand;
        } else if (operator.equals(RECALLMEMORY)) {
            mOperand = mCalculatorMemory;
        } else if (operator.equals(SQUAREROOT)) {
            mOperand = Math.sqrt(mOperand);
        } else if (operator.equals(SQUARED)) {
            mOperand = mOperand * mOperand;
        } else if (operator.equals(INVERT)) {
        if (mOperand != 0) {
                mOperand = 1 / mOperand;
        }
        } else if (operator.equals(TOGGLESIGN)) {
            mOperand = -mOperand;
        } else if (operator.equals(SINE)) {
            mOperand = Math.sin(Math.toRadians(mOperand)); // Math.toRadians(mOperand) converts
result to degrees
        } else if (operator.equals(COSINE)) {
            mOperand = Math.cos(Math.toRadians(mOperand)); // Math.toRadians(mOperand) converts
result to degrees
        } else if (operator.equals(TANGENT)) {
            mOperand = Math.tan(Math.toRadians(mOperand)); // Math.toRadians(mOperand) converts
result to degrees
        } else {
            performWaitingOperation();
        mWaitingOperator = operator;
            mWaitingOperand = mOperand;
        }
        return mOperand;
    }
    protected void performWaitingOperation() {
```

```
        if (mWaitingOperator.equals(ADD)) {
            mOperand = mWaitingOperand + mOperand;
          } else if (mWaitingOperator.equals(SUBTRACT)) {
            mOperand = mWaitingOperand - mOperand;
        } else if (mWaitingOperator.equals(MULTIPLY)) {
            mOperand = mWaitingOperand * mOperand;
        } else if (mWaitingOperator.equals(DIVIDE)) {
        if (mOperand != 0) {
                mOperand = mWaitingOperand / mOperand;
        }
        }
}
}
```

activity_main.xml (in res/layout)

```
<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/row7"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight=".12" >
        <Button
            android:id="@+id/buttonSquareRoot"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight=".25"
            android:text="@string/buttonSquareRoot"
            android:textSize="25sp" />
        <Button
            android:id="@+id/buttonSquared"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight=".25"
            android:text="@string/buttonSquared"
            android:textSize="25sp" />
        <Button
            android:id="@+id/buttonInvert"
            android:layout_width="0dp"
       android:layout_height="match_parent"
            android:layout_weight=".25"
            android:text="@string/buttonInvert"
            android:textSize="17sp" />
        <Button
            android:id="@+id/buttonSine"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight=".25"
            android:text="@string/buttonSine"
            android:textSize="17sp" />
        <Button
            android:id="@+id/buttonCosine"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight=".25"
            android:text="@string/buttonCosine"
            android:textSize="17sp" />
        <Button
            android:id="@+id/buttonTangent"
            android:layout_width="0dp"
```

```
            android:layout_height="match_parent"
            android:layout_weight=".25"
            android:text="@string/buttonTangent"
            android:textSize="17sp" />
    </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>
```

strings.xml (in res/values)

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<resources>
<string name="app_name">Calculator</string>
<string name="menu_settings">Settings</string>
<string name="action_settings">Settings</string>
<string name="button0">0</string>
<string name="button1">1</string>
<string name="button2">2</string>
<string name="button3">3</string>
<string name="button4">4</string>
<string name="button5">5</string>
<string name="button6">6</string>
<string name="button7">7</string>
<string name="button8">8</string>
<string name="button9">9</string>
<string name="buttonAdd">+</string>
<string name="buttonSubtract">-</string>
<string name="buttonMultiply">*</string>
<string name="buttonDivide">/</string>
<string name="buttonToggleSign">+/-</string>
<string name="buttonDecimalPoint">.</string>
<string name="buttonEquals">=</string>
<string name="buttonClear">C</string>
<string name="buttonClearMemory">MC</string>
<string name="buttonAddToMemory">M+</string>
<string name="buttonSubtractFromMemory">M-</string>
<string name="buttonRecallMemory">MR</string>
<string name="buttonSquareRoot">√</string>
<string name="buttonSquared">x²</string>
<string name="buttonInvert">1/x</string>
<string name="buttonSine">sin</string>
<string name="buttonCosine">cos</string>
<string name="buttonTangent">tan</string>
</resources>
```


63
MC
M+
M-
MR
√
X²
1/X
SIN
CO S
TA N
C
+/-
/
*
7
8
9
-
4
5
6
+
1
2
3
=
0
.