**Assignment No.**

**Title:** A Calculator mobile application with Memory Save/Recall.

**Aim:**To design a mobile application for Calculator (+, - ,\*, /, Sin, Cos, sq.-

Root) withMemory Save/Recall.

**Objective:** To study and implement Trigonometry functionality in android studio.

To implement memory save/recall feature in calculator.

To study and perform Positive and Negative testing.

To implement software design and testing in android platform.

**Theory :**

Android operating system is a stack of software components which is roughly divided into five sections and four main layers.

## Linux kernel

At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches. This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

## Libraries

On top of Linux kernel there is a set of libraries including open-source Web browser engine Web Kit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

## Android Libraries

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows −

* **android.app** − Provides access to the application model and is the cornerstone of all Android applications.
* **android.content** − Facilitates content access, publishing and messaging between applications and application components.
* **android.database** − Used to access data published by content providers and includes SQLite database management classes.
* **android.opengl** − A Java interface to the OpenGL ES 3D graphics rendering API.
* **android.os** − Provides applications with access to standard operating system services including messages, system services and inter-process communication.
* **android.text** − Used to render and manipulate text on a device display.
* **android.view** − The fundamental building blocks of application user interfaces.
* **android.widget** − A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
* **android.webkit** − A set of classes intended to allow web-browsing capabilities to be built into applications.

Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

## Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

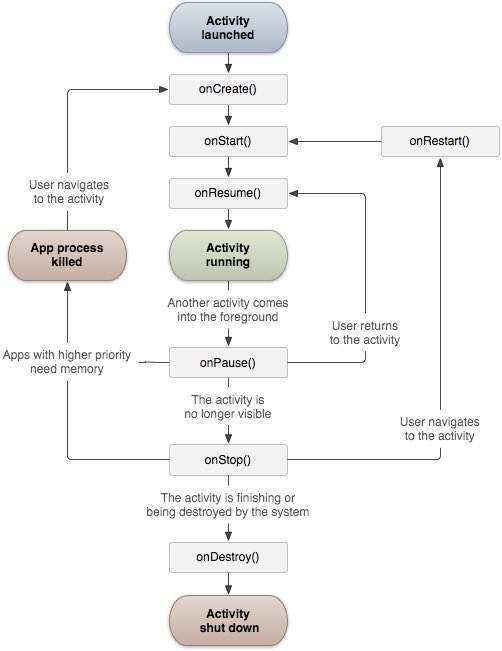
## Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

The Android framework includes the following key services −

* **Activity Manager** − Controls all aspects of the application lifecycle and activity stack.
* **Content Providers** − Allows applications to publish and share data with other applications.
* **Resource Manager** − Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
* **Notifications Manager** − Allows applications to display alerts and notifications to the user.
* **View System** − An extensible set of views used to create application user interfaces.

If you have worked with C, C++ or Java programming language then you must have seen that your program starts from **main()** function. Very similar way, Android system initiates its program with in an **Activity** starting with a call on*onCreate()* callback method. There is a sequence of callback methods that start up an activity and a sequence of callback methods that tear down an activity as shown in the below Activity life cycle diagram: (*image courtesy : android.com* )



**Fig: Activity Diagram**

The Activity class defines the following call backs i.e. events. You don't need to implement all the callbacks methods. However, it's important that you understand each one and implement those that ensure your app behaves the way users expect.

**Memory Functions of your Calculator**

Descrip­tion of each button:

MC > Clears the memory

MR > Recall value in memory

MS > Save value into memory

M+ > Adds the cur­rently dis­played num­ber on your cal­cu­la­tor to the  num­ber in memory

M– > Sub­tracts the cur­rently dis­played num­ber from the num­ber in memory.

**Mathematical Modelling**:

Let ‘S’ be the system such that,

S = {I, O, Fn, Sc, Fc}

Where,

I -> {I1, I2, . . . , In} : set of inputs

O -> {O1, O2, . . . , On} : set of outputs

Fn -> {Fn1, Fn2, . . . ,Fnn} : set of functions

Sc -> {Sc1, Sc2, . . . ,Scn} : set of success cases

Fc -> {Fc1, Fc2, . . . ,Fcn} : set of failure cases

**I: Set of Inputs**

I1: Number as input (float/double)

I2: Function to performed.

**O: Set of Outputs**

O1: Result of Operation

O2: Message for respective trigonometric function.

O3: Message according to memory save and recall operation.

Fn6: cosine of value

Fn7: tan

Fn8: cot

Fn9: sec

Fn10: cosec

Fn13: memory recall

**Fn: Set of Functions**

Fn1: Addition

Fn2:Subtraction

Fn3: Multiplication

Fn4: Division

Fn5: sine of value

Fn11: memory clear

Fn12: memory save

**Sc: Success Cases**

Sc1: valid credentials entered.

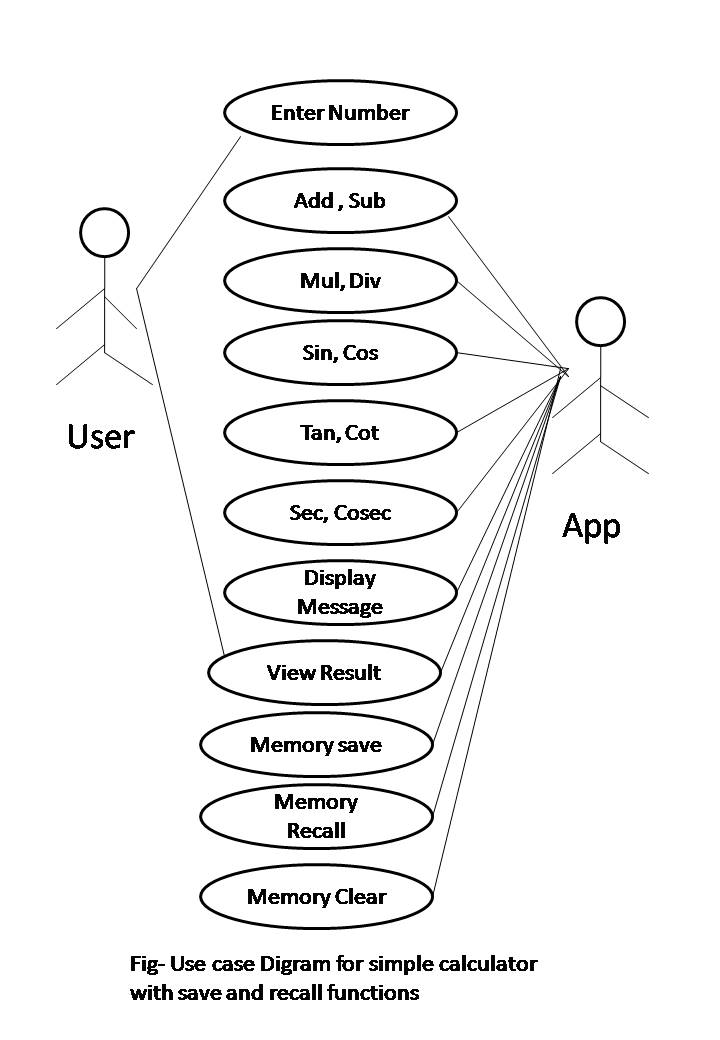
Sc2: Proper implementation of functions

**Fc: Failure Cases**

Sc1: Invalid credentials entered.

Sc2: Improper implementation of functions

**Use Case Diagram:**

****

**System testing:**

For testing purpose we have used manual testing.

**Algorithm**:

Step 1: Start Application

Step 2: Enter valid number in float or double or integer format.

Step 3: Press a button for a specific operation.

1.Add() 6.Cos()

2.Sub() 7.Tan()

3.Mul() 8.Cot()

4.Div() 9.Sec()

5.Sin() 10.Cosec()

11. MemorySave() 12.MemoryCleared()

13.MemoryRecall()

Step 4: Get the result.

**Input** : Number in valid format

**Output**: Result of operation

**Platform :**Ubuntu 14.04 , ADT Bundle

**Conclusion** : Thus a we implemented Simple calculator that performs simple trigonometric function with **memory save and recall function**

**Conclusion:**Thus we have successfully completed Android Mobile App for Calculator having save/recall functionality and Trigonometry functionality and tested on android mobile.