**What is an APP?**

Application specifically devised for smart phones or tablets. Generally apps are small, individual software units with limited features.

**What is Android?**  
It is an open source and linux based operating system for mobile devices like smart phones and tablet computers.

**Features of Android**

1. User friendly User Interface (UI)
2. Connectivity – GSM/EDGE, CDMA, LTE, WiFi, Bluetooth
3. Data Storage – SQLite – Relational Database
4. Media Support – MPEG, MP3, MP4, OGG, WAV, JPEG, PNG, GIF and BMP
5. Messaging – SMS, MMS
6. Web Browser (Chrome, Firefox, Opera, tor)
7. Touch Screen
8. Multitasking – Performing various operations simultaneously.

**Android Platform Block Diagram**

Hardware

Linux Kernel

Libraries

Android Run Time

Application Frame Work

App-1

App-2

App-3

**Hardware**

All the available physical components.

**Linux Kernel**

Android is built using Linux 2.6. It acts as an abstraction layer between hardware and the software lying above it. It provides services such as security, memory management, process management, file management etc., The hardware device specific drivers such as audio drivers, camera drivers etc., are implemented here. This enables variety of devices to be handled by Android.

**Libraries**

Code libraries written in C/C++ meant for various components of Android system are kept here. These libraries can be accessed by applications developed by developers through application layer.

Example:

SGL – Scalable Games Language that supports 2D Graphics.

Media Libraries – These files helps to handle many audio, video files

Image Libraries – These files helps to handle images.

SQLite – A lightweight database engine.

**Android Run Time**

Android Run Time consists of Dalvik Virtual Machine (DVM) and Core Java. First the Java class byte codes are converted to form that can be executed with low memory than the Java byte codes. The transformed code is called Dalvik Executable (dex) format. The DVM can handle multiple VM, each different from another.

**Application Frame Work**

Application frame work provides several services to the applications than the user develops. The applications you develop directly interact with these blocks. These framework blocks provide all the tools to access the function of a phone. The application framework is designed to reuse the components. Any application can publish its capabilities and any other application can make use of these capabilities.

Example:

*Telephone manager* – It manages all voice calls

*View System* – It provides extensive set of views that can be use to build apps. An application with lists, text boxes, buttons etc.,

*Location Manager* - It provides location management using GPRS or Cell tower.

**Applications**

The top layer is the application Layer. In this layer only all the applications will run. Android comes with come built-in core applications line SMS, E-mail, browser and maps. All the applications are written using Java/Kotlin programming languages. A developer can develop their own applications and run it.

**Explain application components in Android**

Application components are the basic building blocks of an application. They are

* Activity
* Service
* Broadcast Receiver
* Content Providers

**Activity**

An activity represents a user screen. It is the most common component used in an application. One application may have one or more activity. But each activity is independent of each other. For example, an email application may have one activity for new e-mails, another activity for to compose e-mail and another activity for reading e-mails. All these three work together but all or independent. When a new screen appears, previous screen is paused and put on to stack. An activity is implemented as a subclass of Activity class and is defined in android.app java package. The activity employs or more views to display the actual screen to user.

**Services**

A service is a program that runs in the background. It has no user interface. Usually a service is a long live code. For example media player paying sound files. A service can be created by creating a subclass of the service class defined in android app java package. One application can start a service and it can continue to run in the background even if the user switches to another program.

**Content Provider**

A content provider is used to manage data used in one application and share it with other application running in the android environment. An application can store data in a file, SQLite databse, web, XML file and access it. Using content provider other applications can query and even modify (if permitted) the data. A content provider is created by subclassing the content provider class defined in android.content package.

**Broadcast Receivers**

The notification about the happening of an event by the Android system to other components of Android system is termed as Broadcast. Broadcast receivers, receive these notifications and respond to it. Not only the android system but also the application can also generate events. Broadcast receiver may not create an UI screen but they may create a status bar notification to alert the users.

**Android Manifest File**

<?xml version =”” encoding=”” utf-8=””>

<manifest>

<user-permissions>

<permission/>

<permission tree/>

<permission group/>

<instrumentation/>

<usersdk/>

<userconfiguration/>

<uses-feature/>

<supports-screen/>

<compatible-screens/>

<application>

<activity>

<intent filter>

<action/>

<category/>

<data/>

<intent filter/>

<activity/>

<uses-library/>

<application/>

<manifest/>

* This file is available in the root directory of every application.
* It contains the essential information required for running the application.
* It names the Java/Kotlin package for the package. The application package name serves as the unique identifier for the application.
* It describes the various components of application like – activities, services, broadcast receivers and content providers.
* It determines which processes will host application components.
* It declares which permissions the application must have in order to access protected part of the API and interact with other applications.
* It also declares the permissions that other are required to interact with the components of the applications.
* It lists the libraries that the application must be linked.

**Different types of Views/User Interface Controls**

*TextView* – Responsible for displaying text field in the screen

*EditText* – Provides editable text box.

*AutoCompleteTextView* – Displays the list of all the completed suggestions on its own.

*Button* – Used to perform on actions. This can be pressed or clicked.

*ImageButton* – Text on the button is replaced by an image.

*CheckBox* – On/Off toggle switch. User can select multiple options from the given set of options.

*ToggleButton* – Used to Switch on/off on activity with the help of light indicator.

*RadioButton* – Can be checked or un checked.

*RadioGroup* – Combination of more than one Radio Button.

*ProgressBar* – Useful to display the progress of a background activity.

*Spinner* – Represent a drop-down list.

*TimePicker* – Provides an option to the user to select a time of the day. 24-hour mode and AM/PM options are available.

*DatePicker* – Provides an option to select date.

Part A – Questions

1. Android is an **Operating System** for mobile phones
2. Android is built on **Linux** kernel
3. Code libraries in Android system are written in **C/C++** language
4. **SQLite** is a light weight database system for Android.
5. **Webkit** in Android system is used for web browsing.
6. Android Runtime consists of **Dalvic Virtual Machine** and **Core Java Libraries**
7. All applications in Android system are written using **Java/Kotlin** languages.
8. There is no distinction between built-in application and **User-defined applications**
9. In an Android application, a View represents an **Activity**
10. A service is a program that runs in the **background.**
11. A service program has a **long life**.
12. A content provider is used to share **data** with other applications
13. A broadcast receiver is used to **receive** messages from other applications.
14. Broadcast is a **message** sent by the Android system about the happening of an event
15. An intent is an **object** containing the details of the message created by the components of the Android system.
16. Intent filters select **appropriate** item that meets the requirement of an intent.
17. All components used in an Android application are declared in the **AndroidManifest.XML**
18. In Android system, the Java byte codes are converted to **dex file format** for increasing the efficiency of the executing code.

**Intent**

An intent is an object containing the details of the message created by the components of the Android system. Usually an intent is looking for a service or activity or broadcast receiver which meets it s requirement.

**Intent Filter**

An intent filter is an agent that contains the services, activities and broadcast receivers that are used in that application. The function of an intent filter is to select the most appropriate item that meets the requirement of an intent that has been dispatched by an Android component.

**Android View**

A view is a simple building block of a user interface. It refers to the android.view.View class. It is small rectangular box that can be a TextView, EditText or even a button. It occupies the area on the screen in a rectangular area and is responsible for drawing and event handling. View is a superclass of all the graphical user interface components.

**ViewGroup**

ViewGroup class is a subclass of the View class. It refers to the android.view.ViewGroup class. It will act as a base class for layouts and layouts parameters. The ViewGroup will provide an invisible container to hold other Views or ViewGroups and to define the layout properties.

View

TextView

ImageView

ViewGroup

Linear Layout

Relative Layout

Linear Layout

Relative Layout

Frame Layout

Table Layout

List View

Grid View

Web View

**Orientation**

There are two types of orientations available. They are horizontal and vertical.

Horizntal

Vertical

activity\_main.java – Code – Activity details

MainActivity.java –Run time version (Dalvic Executable Version)

<LinearLayout xmlns:android=<http://lfsjdaflsdajfsladfjslfjsa>

android.layout\_width = “fill\_parent”

android.layout\_height = “wrap\_content”

android.orientation = “horizontal”

<Button

android.layout\_width=”wrap\_content”

android.layout\_height = “wrap\_content”

android:text=”Button1”

android.id=”@+id/button”

android.background = “#5643” />

**Relative Layout**

The relative layout is very flexible layout used in android for custom layout designing, because we can position the component at any required position. It gives us the flexibility to position our view based on the relative component’s position.

**Android.layout\_above** – It accepts another sibling view id and places the view above the specified view id.

**Android.layout\_below** – It accepts another sibling view id and places the view below the specified view id.

**Android.layout\_toLeftOf** – It accepts another sibling view id and places the view left of the specified view id.

**Android.layout\_toRightOf** – It accepts another sibling view id and places the view right of the specified view id.

**Android.layout\_toTopOf** – It accepts another sibling view id and places the view top of the specified view id.

**Android.layout\_toBottomOf** – It accepts another sibling view id and places the view bottom of the specified view id.

**Parent – Child**

**Layout\_alignParentTop** – If it specified “true”, the top edge of view will match the top edge of the parent.

**Layout\_alignParentBottom** – If it specified “true”, the top edge of view will match the bottom edge of the parent.

**Layout\_alignParentLeft** – If it specified “true”, the top edge of view will match the left edge of the parent.

**Layout\_alignParentRight** – If it specified “true”, the top edge of view will match the right edge of the parent.

**Layout\_CenterInParent** – If it specified “true”, the top edge of view will match the centre of the parent.

**Layout\_CenterHorizontal** – If it specified “true”, the top edge of view will match the horizontal centre of the parent.

**Layout\_CenterVertical** – If it specified “true”, the top edge of view will match the horizontal centre of the parent.