Android Assignment

Research

a. What is Android? Who created it? What are Android Apps?

Android is a mobile operating system based on a modified version of the Linux kernel and other open source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Andy Rubin, Nick Sears and Rich Miner created it.
An Android app is a software application running on the Android platform.

b. What is the software used in the development of Android Apps?

Android Studio is the software used in the development of Android Apps.

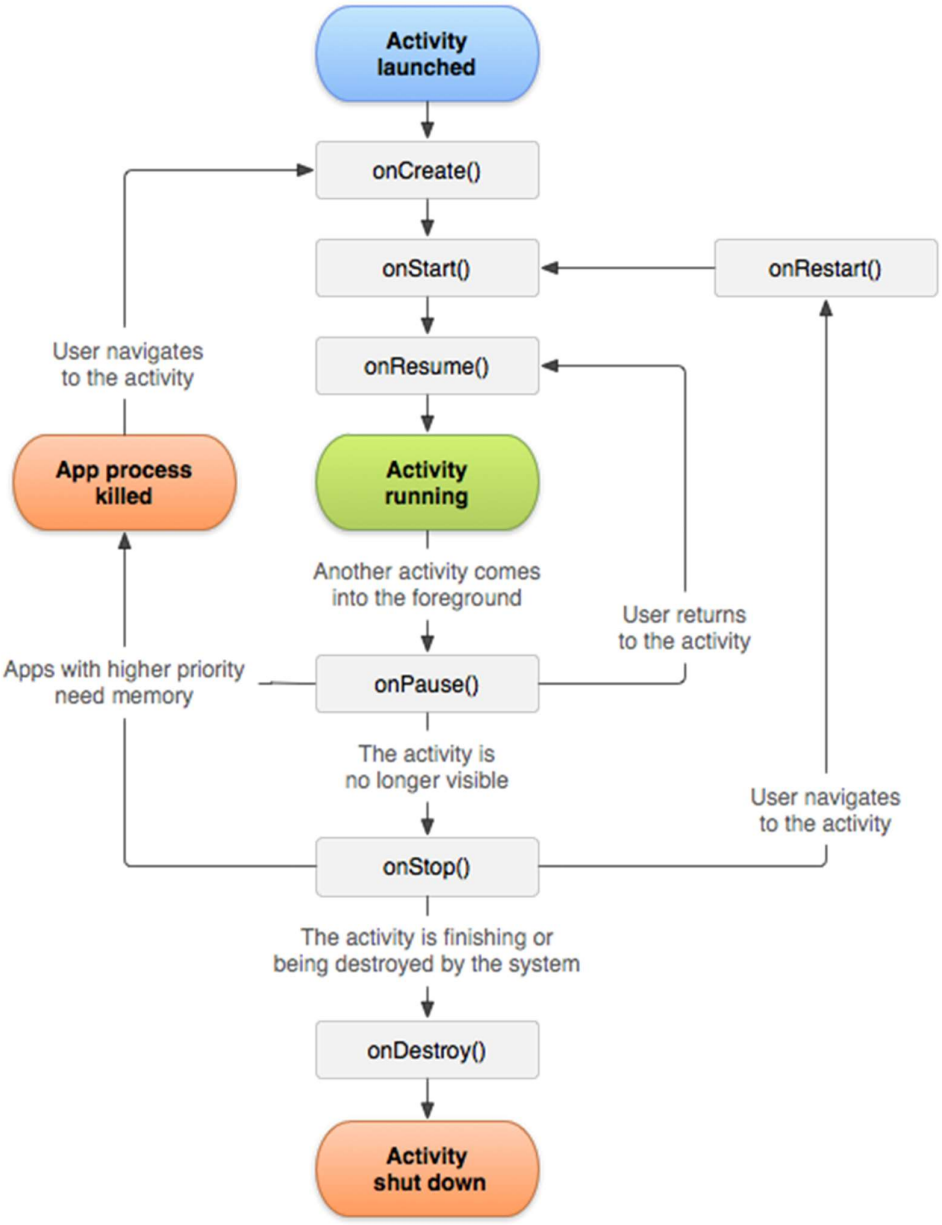
c. Which are the languages commonly used in the development of android apps? Which language does InstiApp use?

Android apps are commonly written using Kotlin, Java, and C++ languages. InstiApp uses Java.

d. What is the activity cycle of a basic Android application? Diagrams/flowcharts preferred.

As a user navigates through, out of, and back to your app, the Activity instances in your app transition through different states in their lifecycle. The Activity class provides a number of callbacks that allow the activity to know that a state has changed: that the system is creating, stopping, or resuming an activity, or destroying the process in which the activity resides. To navigate transitions between stages of the activity lifecycle, the Activity class provides a core set of seven callbacks: onCreate(), onStart(), onResume(), onPause(), onStop(), onRestart() and onDestroy(). The system invokes each of these callbacks as an activity enters a new state.

1. *onCreate():* You must implement this callback, which fires when the system creates your activity. When onCreate() finishes, the next callback is always onStart().
2. *onStart():* As onCreate() exits, the activity enters the Started state, and the activity becomes visible to the user. This callback contains what amounts to the activity's final preparations for coming to the foreground and becoming interactive.
3. *onResume():* The system invokes this callback just before the activity starts interacting with the user. The onPause() callback always follows onResume().
4. *onPause():* The system calls onPause() when the activity loses focus and enters a Paused state. Once onPause() finishes executing, the next callback is either onStop() or onResume(), depending on what happens after the activity enters the Paused state.
5. *onStop():* The system calls onStop() when the activity is no longer visible to the user. The next callback that the system calls is either onRestart(), if the activity is coming back to interact with the user, or by onDestroy() if this activity is completely terminating.
6. *onRestart():*The system invokes this callback when an activity in the Stopped state is about to restart. This callback is always followed by onStart().
7. *onDestroy():* The system invokes this callback before an activity is destroyed. onDestroy() is usually implemented to ensure that all of an activity's resources are released when the activity, or the process containing it, is destroyed.



e. What are 5 different UI elements in an android app? One example is a “TextView”.

1. *TextView:* This control is used to display text to the user.
2. *Button:* A push-button that can be pressed, or clicked, by the user to perform an action.
3. *CheckBox :* An on/off switch that can be toggled by the user. You should use check box when presenting users with a group of selectable options that are not mutually exclusive.
4. *ProgressBar:* The ProgressBar view provides visual feedback about some ongoing tasks, such as when you are performing a task in the background.
5. *Spinner:* A drop-down list that allows users to select one value from a set.

f. [BONUS]What are some of the salient features of those languages (part c)? How similar are they to C++?

Java:

1. Java is secure (no threat to security because nothing gets executed outside the JVM)
2. Java supports Object Oriented Paradigm.
3. Java's core features are complete and vast. Also, they're regularly updated and maintained by oracle.
4. Java is platform independent.
5. Outside the core library, java has many frameworks and classes for features like networking, threading, IO operations and thus, programmers can leverage these qualities in their apps.
6. Java is open source.

Kotlin: Kotlin is increasingly replacing Java because -

1. Kotlin allows writing less code
2. It solves developer challenges
3. Adopting Kotlin is easy
4. Kotlin is fully compatible with Java
5. It imposes no runtime overhead
6. Kotlin has a strong community
7. Kotlin suits for the multi-platform development
8. Kotlin development offers more safety

Comparison with C++:

Similarities:

1. Syntax: Classes, looping structure, defining variables, and conditional operators are all there in all three languages.
2. Object-Oriented: All three languages use 'OOPS' concept for easier recognition of components in your program.
3. Entry Points: All three languages look for the entry point.

Differences:

1. Interpreted vs. Compiled: Java is an interpreted language due to which it can be used on any operating system. C++ and Kotlin, on the other hand, are compiled languages which means they can be operated only on specific OS.
2. Memory Safe: Java and Kotlin are extremely safe languages. Even if you attempt to assign values outside the program or an array parameter, the programmer receives an error. C++, allows the user to assign values outside the program which later on can cause bugs and run-time errors.
3. Performance: When we talk about performance, developers like the way Java and Kotlin execute without any real-time errors. But they are a bit slower than C++.

Task

1. Read about relative and linear layouts and how they are used to design the UI of Apps.

   A layout defines the structure for a user interface in your app, such as in an activity. All elements in the layout are built using a hierarchy of View and ViewGroup objects. A View usually draws something the user can see and interact with. Whereas a ViewGroup is an invisible container that defines the layout structure for View and other ViewGroup objects. The View objects are usually called "widgets" and can be one of many subclasses, such as Button or TextView. The ViewGroup objects are usually called "layouts" can be one of many types that provide a different layout structure, such as LinearLayout or RelativeLayout.

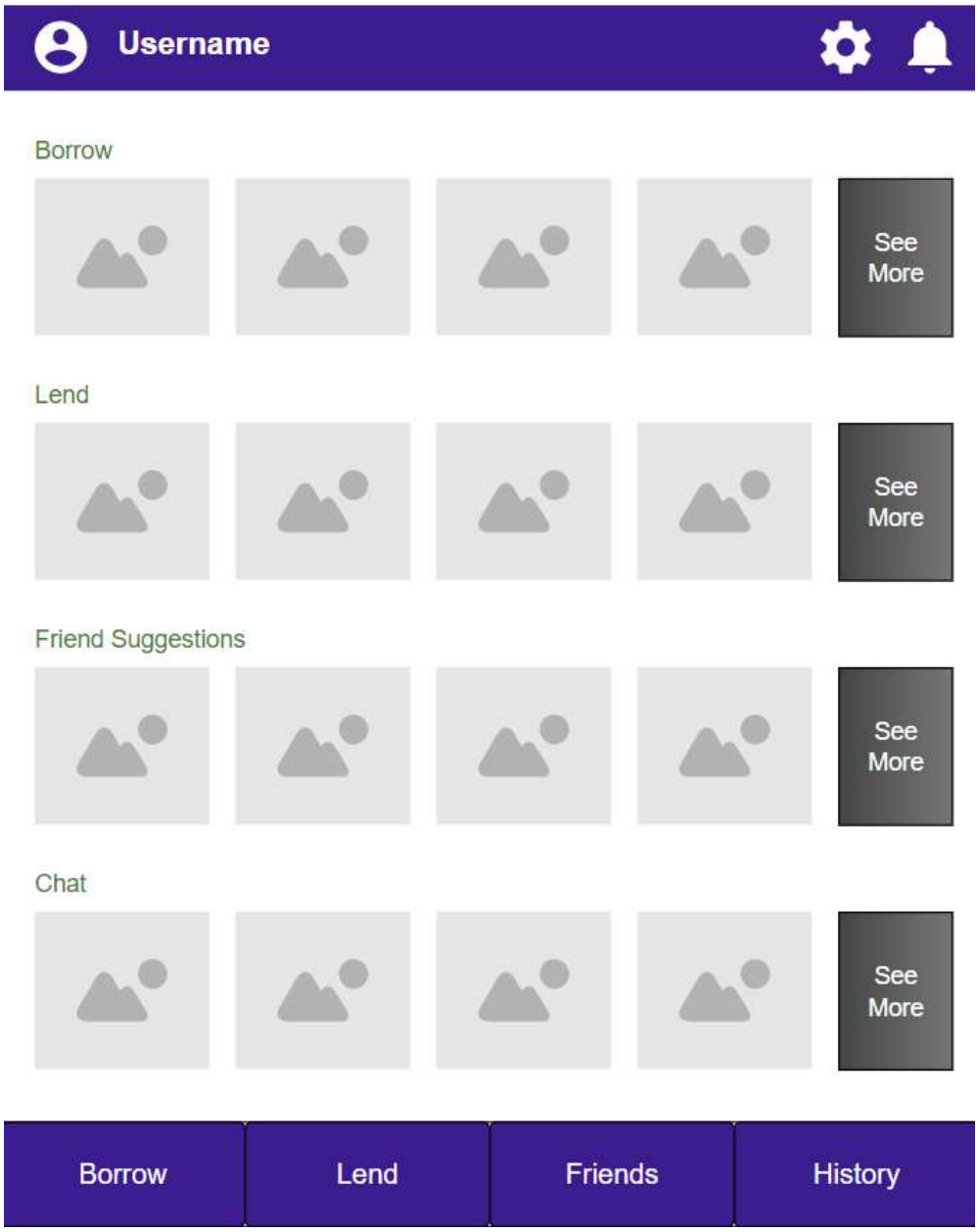
   Linear Layout: LinearLayout is a view group that aligns all children in a single direction, vertically or horizontally. You can specify the layout direction with the android:orientation attribute. All children of a LinearLayout are stacked one after the other, so a vertical list will only have one child per row, no matter how wide they are, and a horizontal list will only be one row high (the height of the tallest child, plus padding). A LinearLayout respects margins between children and the gravity (right, center, or left alignment) of each child.

   RelativeLayout: is a view group that displays child views in relative positions. The position of each view can be specified as relative to sibling elements (such as to the left-of or below another view) or in positions relative to the parent RelativeLayout area (such as aligned to the bottom, left or center). A RelativeLayout is a very powerful utility for designing a user interface because it can eliminate nested view groups and keep your layout hierarchy flat, which improves performance. If you find yourself using several nested LinearLayout groups, you may be able to replace them with a single RelativeLayout.

2. Now suppose you want to design the landing screen / dashboard of Book-ed!, what do you think should be the various features of that screen?

   1. Top App Bar: Contains username on left, settings and notifications on right.
   2. Borrow (Based on what friends are lending)
   3. Lend (Based on what friends want to borrow)
   4. Friend suggestions (Based on similar borrow/lend history)
   5. Chat (With friends)
   6. Bottom App Bar: Contains ‘Borrow, ‘Lend’, ‘Friends’ and ‘History’.

3. Draw a schematic diagram of the screen. You can do this using a pen and paper or use online android prototyping tools.



Made using Moqup(online mockup making tool).

4. Now break down your design into various layouts and elements. Clearly mark what is a linear layout, what are the various elements being used in your design. For example, if there is a piece of text somewhere on the screen, that part would be the “TextView”.

   Layout: Vertical Linear Layout consisting of ten Viewgroups(2 app bars, 4 text lines, 4 image bars), each having Horizonatal Linear Layout.

   Elements: In top app bar, 3 ImageButtons and 1TextView is used. In bottom app bar, 4 Buttons are used. In 4 text lines, TextView is used. In 4 image bars, 4 ImageButtons and 1 Button is used.

5. [BONUS] Install “Android Studio” on your laptop with all the necessary requirements and run the starter app on your android device.

6. [BONUS] Write the code for the dashboard which you just designed/prototyped.