Advanced Android Programming

1. Study following object oriented concepts

You have studied object oriented programming concepts on earlier courses. I want that

you recap following issues in Java programming language. Understanding these are

really important to successfully complete the course

1. Object:

Ans: An object is an instance of class. It has states and behaviours. For example: A dog has states: colour, name, breed as well as behaviours: wagging tail, barking etc.

2. Class:

Ans: Class is a blueprint/template that describes the states/behaviour that supports object of its type.

3. Instantiation of object (creating an object):

Ans: The creation of an instance is called an instantiation. Object created from a class.

4. Visibility (public / private / protected)

Ans Public: It is accessible from other classes too.

Private: It is accessible only in the same class.

Protected: It is accessible only in the same class as well as to the inherited class methods.

5. Member datas / methods

Ans: Member datas are variables that belong to an object. A cat object for instance could have member data such as a string color and int age. Each cat object can then store, maintain and provide upon request its own information regarding its color and age.

6. Inheritance

Ans: It means that one class inherits the other class; meaning it has the property of the other class too if it inherits it.

7. Interface

Ans: An interface in java is a blueprint of a class. It has static constants and abstract methods.

It is used to achieve abstraction.

By interface, we can support the functionality of multiple inheritance.

It can be used to achieve loose coupling.

8. Polymorphism

Ans: Polymorphism in Java is a concept by which we can perform a single action in different ways

We can perform polymorphism in java by method overloading and method overriding.

9. Overriding

Ans: In any object-oriented programming language, Overriding is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super-classes or parent classes.

10.Abstract classes

Ans: An abstract class is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be sub-classed.

3. Study Android fundamental concepts

<https://developer.android.com/guide/components/fundamentals>

Answer to following questions:

a) What programming languages you can use for Android app development?

Android apps can be written using Kotlin, Java, and C++ languages

b) What is .apk file?

APK stands for Android Package Kit. .apk is the package file format used by the Android operating system for distribution and installation of mobile apps. It is similar as Windows (PC) systems use an .exe file for installing software, the APK does the same for Android.

c) How Android system runs apps?

One APK file contains all the contents of an Android app and is the file that Android

powered devices use to install the app.

Each Android app lives in its own security sandbox, protected by the following Android security features:

The Android operating system is a multi-user Linux system in which each app is a different user.

By default, the system assigns each app a unique Linux user ID (the ID is used only by the system and is unknown to the app). The system sets permissions for all the files in an app so that only the user ID assigned to that app can access them.

Each process has its own virtual machine (VM), so an app's code runs in isolation from other apps.

By default, every app runs in its own Linux process. The Android system starts the process when any of the app's components need to be executed, and then shuts down the process when it's no longer needed or when the system must recover memory for other apps.

d) Name four types of Android components. Describe each.

There are four different types of app components which are described below:

i). Activities: An activity is the entry point for interacting with the user. It represents a single screen with a user interface. For example, an email app might have one activity that shoes a list of new emails, another activity to compose and email, and another activity for reading emails. Although the activities work together to form a cohesive user experience in the email app, each one is independent of the others. As such, a different app can start any one of these activities if the email app allows it. For example, a camera app can start the activity in the email app that composes new mail to allow the user to share a picture. An activity facilitates the following key interactions between system and app.

You implement an activity as a subclass of the Activity.

ii) Services: A service is a general-purpose entry point for keeping an app running in the background for all kinds of reasons. It is a component that runs in the background to perform long-running operations or to perform work for remote processes. A service does not provide a user interface. For example, a service might play music in the background while the user is in different app, or it might fetch dat over the network without blocking user interaction with an activity. Bound services run because some other app has said that it wants to make use of the service.

A service is implemented as a subclass of Service.

iii) Broadcast receivers: A broadcast receiver is a component that enables the system to deliver events to the app outside of a regular user flow, allowing the app to respond to system-wide broadcast announcements. Because broadcast receivers are another well defined entry into the app, the system can deliver broadcasts even to apps that are not currently running. For example, an app can schedule an alarm to post a notification to tell the user about an upcoming event and by delivering that alarm to a BroadcastReceiver of the app, there is no need for the app to remain running until the alarm goes off.

A broadcast receiver is implemented as a subclass of BroadcastReceiver and each broadcast is delivered as an Intent object.

iv) Content providers: A content provides manages a shared set of app that you can store in the file system, in a SQLite databas, on the web, or on any other persistent storage location the you app can access.

A content provider is implemented as a subclass of ContentProvider and must implement a standard set of APIs that enable other apps to perform transactions.

e) What is manifest file and what is its purpose?

Every app project must have an AndroidManifest.xml file (with precisely that name) at the root of the project source set. The manifest file describes essential information about your app to the Android build tools, the Android operating system, and Google Play.

If you're using Android Studio to build your app, the manifest file is created for you, and most of the essential manifest elements are added as you build your app (especially when using code templates).

f. What are resources? Why they are needed?

Resources are the additional files and static content that your code uses, such as bitmaps, layout definitions, user interface strings, animation instructions, and more.