ANDROID QUESTIONS

Ques 1. Which one of the followings is not the nickname of a version of Android?

A. Cupcake
B. Gingerbread
C. Honeycomb
D. Muffin

Ques 2. Android is based on Linux for the following reasons.

A. Security
B. Portability
C. Networking
D. All of the above

Ques 3. What is the name of the program that convert the Java byte codes of the Dalvik byte codes?

A. Android interpretive complier
B. Dalvik convertor
C. Dex complier
D. Mobile Interpretive Complier

Ques 4. The Emulator is identical to running a real phone except when emulating/simulating what?

A. Telephony
B. Applications
C. Sensor
D. The emulator can emulate/simulate all aspects of a smart phone

Ques 5. To create an emulator, you need an AVD. What does it stand for?

A. Android Virtual Display
B. Android Virtual Device
C. Application Virtual Display
D. Active Virtual Device

Ques 6. What is the driving force behind an Android application and that ultimately gets converted into Dalvik executable?

A. Java source code

B. R- files
C. The emulator
D. The SDK

Ques 7. What runs in the background and doesn't have any UI components?

A. Applications
B. Services
C. Intents
D. Content provides

Ques 8. Who developed kotlin?

A. IBM
B. GOOGLE
C. JETBRAINS
D. MICROSOFT

Ques 9. Which extension is responsible to save Kotlin files?

A. .kot
B. .android
C. .src
D. .kt or .kts

Ques 10. What is stored in "obj" in my class obj; line of code?

Ans. Memory is allocated to an object using the "new" operator. While simply declares a reference to the object, no memory is allocated to it, hence it points to null.

Ques 11. Which of the operators is used to allocate memory for an object?

Ans. The operator "new" dynamically allocates memory for an object and returns a reference to it. This reference is the memory address of the object allocated by "new".

Ques 12. In java, when we implement an interface method, what it must be declared as?

Ans. It must be declared as "public"

Ques 13. What is an anonymous class in java?

Ans. A type of inner class without a name.

Ques 14. What is the final class in java?

Ans. A class that cannot be extended

Ques 15. How can you prevent inheritance from a class in c#.net?

Ans. Declare the class as sealed

Ques 16. Android is based on which of language?

Ans. Java language is mainly used to write the android code even though other languages can be used.

Ques 17. Android is licensed under what?

Ans. The android platform was released under the apache 2.0 license, and it is responsible for the copyright of the android open-source project. The apache foundation permits and grants licenses for software uses and distribution by the copyright under the android open-source project.

Ques 18. Which of the following virtual machine is used by the android operating system?

Ans. The dalvik virtual machine (dvm) is an android virtual machine optimized for mobile devices. It optimizes the virtual machine for memory, battery life, and performance. Dalvik is a name of a town in iceland. The dalvik vm was written by dan bornstein.

Ques 19. Which of the following converts java byte code into dalvik byte code?

Ans. The dex compiler converts the class files into a .dex file that runs on the dalvik vm. Multiple class files are converted into one dex file.

Ques 20. How can we stop the services in android?

Ans. A service is started when a component (like activity) calls the startservice() method; now, it runs in the background indefinitely. It is stopped by the stopservice() method. The service can stop itself by calling the stopself() method.

Ques 21. How can we kill an activity in android?

Ans. The finish() method is used to close the activity. Whereas the finishactivity(int requestcode) also closes the activity with requestcode.

Ques 22. Developers can test the application, during developing the android applications?

Ans. We can use the android emulator, physical android phone, or third-party emulator as a target device to execute and test our android application.

Ques 23. Does android support other languages than java?

Ans. Yes, an android app can be developed in c/c++ also using android ndk (native development kit). It makes the performance faster. It should be used with android sdk.

Ques 24. What is the use of content provider in android?

Ans. A content provider is used to share information between android applications.

Ques 25. Can you explain the android activity life cycle?

Ans. After a user navigates within the app, then the activity instances transit through different stages in their lifecycle. These activity classes provide several actions called as “callbacks” that gives information of the changed states the system creates, resumes, or stops while resuming the activity. The activity life cycle has 4 states –

Active or running – if the activity is in the foreground of the screen, it is called as active.

Paused – if the activity has lost focus but is still visible (like in the case of dialog comes top), then it is reoffered as paused.

Stopped – if an activity is completely obscured by another activity, it’s called as stopped. It still retains all states and the information of member components.

Finish – if an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish or simply killing the process.

Ques 26. What are loaders?

ANS. Loader is a set of APIs, known as Loader APIs, given by Android, to load data asynchronously in activity/fragment. The Use of Loader can fetch data asynchronously without blocking the main thread and it manages it's own lifecycle during onDestroy() and configuration changes.

Ques 27. Why is API security important?

Businesses use APIs to connect services and to transfer data. Broken, exposed, or hacked APIs are behind major data breaches. They expose sensitive medical, financial, and personal data for public consumption. That said, not all data is the same nor should be protected in the same way. How you approach API security will depend on what kind of data is being transferred. If your API connects to a third-party application, understand how that app is

funneling information back to the internet. To use the example above, maybe you don't care if someone finds out what's in your fridge, but if they use that same API to track your location you might be more concerned.

Ques 28. What are some of the most common API security best practices?

You probably don't keep your savings under your mattress. Most people their money in a trusted environment (the bank) and use separate methods to authorize and authenticate payments. API security is similar. You need a trusted environment with policies for authentication and authorization.

Here are some of the most common ways you can strengthen your API security:

- Use tokens. Establish trusted identities and then control access to services and resources by using tokens assigned to those identities.
- Use encryption and signatures. Encrypt your data using a method like TLS(see above). Require signatures to ensure that the right users are decrypting and modifying your data, and no one else.
- Identify vulnerabilities. Keep up with your operating system, network, drivers, and API components. Know how everything works together and identify weak spots that could be used to break into your APIs. Use sniffers to detect security issues and track data leaks.
- Use quotas and throttling. Place quotas on how often your API can be called and track its use over history. More calls on an API may indicate that it is being abused. It could also be a programming mistake such as calling the API in an endless loop. Make rules for throttling to protect your APIs from spikes and Denial-of-Service attacks.
- Use an API gateway. API gateways act as the major point of enforcement for API traffic. A good gateway will allow you to authenticate traffic as well as control and analyze how your APIs are used.

Ques 29. What is Inheritance?

Ans. Inheritance in OOP = When a class derives from another class. The child class will inherit all the public and protected properties and methods from the parent class. In addition, it can have its own properties and methods. An inherited class is defined by using the extends keyword.

Ques 30. Explain Hierarchy of Java Exception Class.

Ans. In Java "an event that occurs during the execution of a program that disrupts the normal flow of instructions" is called an exception. This is generally an unexpected or unwanted event

which can occur either at compile-time or run-time in application code. Java exceptions can be of several types and all exception types are organized in a fundamental hierarchy.

The class at the top of the exception class hierarchy is the Throwable class, which is a direct subclass of the Object class. Throwable has two direct subclasses –

A. Exception (IO exception, Reflective Operation Exception, Runtime Exception, Clone Not Supported Exception, Interrupted exception)
B. Error (LEAKAGE ERROR, ASSERTION ERROR, THREAD DEATH, VIRTUAL MACHINE ERROR).

Ques 31. What do you understand by Multi-threading?

Ans. Every application has at least one thread, the main thread. This is the topmost thread in the hierarchy of threads, which can delegate its tasks to other threads. This delegation of tasks by the main thread to several other background threads is what we call multithreading. Multithreading is nothing but performing tasks concurrently, by scheduling them on multiple threads to improve the application's performance.

Ques 32. What is the Runnable Interface in Java?

Ans. A runnable interface is an interface that contains a single method. The Java program defines this single method in java.lang package and calls it upon the execution of the thread class. It provides a template for objects that a class can implement using Threads. You can implement the runnable interface in Java in one of the following ways:

1. Using subclass thread
2. Overriding the run() method

Ques 33. Android versions and its names

Version	SDK/API level	Version Code	Code Name
Android 13	Level 33	Tiramisu	Tiramisu
Android 12	Level 32 (12L)	S_V2	Snow Cone
Android 12	Level 31 (12)	S	Snow Cone
Android 11	Level 30	R	Red Velvet Cake
Android 10	Level 29	Q	Quince Tart
Android 9	Level 28	P	Pie
Android 8	Level 27 (8.1)	O_MR1	Oreo
Android 8	Level 26 (8.0)	O	Oreo
Android 7	Level 25 (7.1)	N_MR1	Nougat
Android 7	Level 24 (7.0)	N	Nougat
Android 6	Level 23	M	Marshmallow
Android 5	Level 22 (5.1)	Lollipop_MR1	Lollipop
Android 5	Level 21 (5.0)	Lollipop, L	Lollipop

Android 4	Level 20 (4.4W)	Kitkat_watch	Kitkat
Android 4	Level 19 (4.4)	Kitkat	Kitkat
Android 4	Level 18 (4.3)	Jelly_Bean_MR2	Jelly_Bean
Android 4	Level 17 (4.2)	Jelly_Bean_MR1	Jelly_Bean
Android 4	Level 16 (4.1)	Jelly_Bean	Jelly_Bean
Android 4	Level 15 (4.0.3 – 4.0.4)	Ice_Cream_Sandwich_MR1	Ice_Cream_Sandwich
Android 4	Level 14 (4.0.1 - 4.0.2)	Ice_Cream_Sandwich	Ice_Cream_Sandwich
Android 3	Level 13 (3.2)	Honeycomb_MR2	Honeycomb
Android 3	Level 12 (3.1)	Honeycomb_MR1	Honeycomb
Android 3	Level 11 (3.0)	Honeycomb	Honeycomb
Android 2	Level 10 (2.3.3 - 2.3.7)	Gingerbread_MR1	Gingerbread
Android 2	Level 9 (2.3.0 - 2.3.2)	Gingerbread	Gingerbread
Android 2	Level 8 (2.2)	Froyo	Froyo
Android 2	Level 7 (2.1)	Eclair_MR1	Eclair
Android 2	Level 6 (2.0.1)	Eclair_0_1	Eclair
Android 2	Level 5 (2.0)	Eclair	Eclair
Android 1	Level 4 (1.6)	Donut	Donut
Android 1	Level 3 (1.5)	cupcake	Cupcake
Android 1	Level 2 (1.1)	Base_1_1	Petit Four
Android 1	Level 1 (1.0)	Base	None

Ques 34. Gradle

Kotlin variable	Groovy variable	Definition
minSdk	minSdkVersion	The minimum SDK version your app will support, defined in build.gradle. For example, if your minSdk is 26, this SDK version corresponds to API Level 26 and Android 8, so your app will only run on devices with Android 8 or higher.
targetSdk	targetSdkVersion	The SDK version that your app targets, defined in build.gradle. This should always be the same as compileSdk.
compileSdk	compileSdkVersion	The SDK version that your app compiles against, defined in build.gradle. Android Studio uses this SDK version to build your

		AABs and APKs. This should always be the same as targetSdk.

Ques 35. What do you understand by Build System? Mention a few pros and cons of Ant, Maven and Gradle.

Ans. The Android build system compiles app resources and source code, and packages them into APKs or Android App Bundles that you can test, deploy, sign, and distribute.

Pros & Cons of Ant, Maven and Gradle

	Ant	Maven	Gradle
Pros	○ Very Flexible and Adaptable ○ Widespread and highly developed ○ Procedural process	○ Build lifecycle ○ More intuitive ○ Declaration Process	○ Native Ivy Dependency management ○ Multi project Ability ○ Full Ant Support ○ Control Structure
Cons	○ Big xml overhead ○ Very specific	○ Many conventions ○ Poorly flexible	○ Hard to learn ○ Few plugins ○ Not fully developed

Ques 36. Core Steps to follow for reverse app engineering an app and hook some behavior with JADX & Frida?

Ans. To reverse engineer an app and hook some behaviour, there's a few core steps you need to work through:

1. Download a copy of the app on your computer
2. Extract the source code

3. Find the code we're interested in
4. Understand how that code works
5. Write a Frida hook to change how that code works

Ques 37. Difference between MVP and MVVM ?

MVP (MODEL VIEW PRESENTER)	MVVM (MODEL VIEW VIEWMODEL)
Developed as the second iteration of software architecture which is advance from MVC.	Industry-recognized architecture pattern for applications.
It resolves the problem of having a dependent View by using Presenter as a communication channel between Model and View.	This architecture pattern is more event-driven as it uses data binding and thus makes easy separation of core business logic from the View.
The one-to-one relationship exists between Presenter and View as one Presenter class manages one View at a time.	Multiple View can be mapped with a single ViewModel and thus, the one-to-many relationship exists between View and ViewModel.
Follows modular and single responsibility principle.	Follows modular and single responsibility principle.
Ideal for simple and complex applications.	Not ideal for small scale projects.
The View has references to the Presenter.	The View has references to the ViewModel
It has a low dependency on the Android APIs.	Has low or no dependency on the Android APIs.
Easy to carry out Unit testing but a tight bond of View and Presenter can make it slightly difficult.	Unit testability is highest in this architecture.
The View is the entry point to the Application	The View takes the input from the user and acts as the entry point of the application.
Code layers are loosely coupled and thus it is easy to carry out modifications/changes in the application code.	Easy to make changes in the application. However, if data binding logic is too complex, it will be a little harder to debug the application.