Java Interview Questions

Curated by : Vaidhyanathan S M

1. Why Java doesn't support multiple Inheritance?

The reason behind this is to prevent ambiguity. Consider a case where class B extends class A and class C and both class A and class C have the same method "display()". Now Java compiler can't decide, which display method it should inherit. To prevent such situations, multiple inheritance is not allowed in Java.

2. Why are string objects immutable in Java?

Because Java uses the concept of string literal. Suppose there are 2 reference variables, all refer to one object "Hello". If one reference variable changes the value of the object, it will affect the other reference variable. Also, String class in Java is final.

3. Define Class and Object in Java.

In OOPs, a class is a template for creating and describing objects (real-life entities) and providing initial values for the state and implementation of behavior. States are called as attributes or data members. Behaviors are called as methods or member functions.

An object is an instance of the class. It is a real-life entity that has states and behavior(s).

4. What is Polymorphism?

It is the ability of an object to take multiple forms. There are two types of polymorphisms.

i) Compile time polymorphism - The function call is resolved by the compiler at the compile time itself. For example, function or method overloading. It is also known as Early binding.

ii) Runtime polymorphism - The function call is resolved by the compiler at the runtime. For example, method overriding. It is also known as Late binding or Dynamic binding.

5. What do you mean by encapsulation?

It is defined as wrapping up of data members and member functions into a single unit called class. Since the data is hidden from other classes, it is also known as data hiding.

6. What is Abstraction?

It is the property by virtue of which only essential details are displayed to the user. For example, consider the case of a driver driving a car. The driver only knows how to use the controls to drive the car but is unaware of the internal working of those components of the car. In Java, abstraction is achieved by using interfaces and abstract classes. 100% abstraction can be achieved using interfaces.

7. Define an Interface.

An interface is an abstract data type that is used to specify a behavior that classes must implement. It is used to achieve 100% abstraction. Members of an interface are static and final by default.

8. What do you understand by Object-Oriented Programming?

OOP is a programming paradigm based on the concept of “objects” that contain data and methods. It is used to increase flexibility and maintainability of programs.

9. Mention some features of Java.

1. Platform independent.
2. Object-oriented
3. Automatic memory allocation & garbage collection
4. Portable
5. Used in creation of Desktop, Mobile and Web applications.

10. Why doesn't Java have pointers?

Java doesn't have pointers because it doesn't need them for general purpose object-oriented programming. Furthermore, adding pointers to Java would undermine the security and robustness and make the language more complex.

11. Describe the usage of the final keyword in Java.

final keyword is used with variables when the value of the variable is not going to change and is going to remain constant. When the final keyword is used with methods, then they can't be overridden in the derived class. Final keyword is used with class to prevent the class from being subclassed when writing APIs or libraries so that the base behavior is not altered.

12. Why can't static methods call non-static methods?

A static method is not tied to any object/instance of the class, while a non-static method always refers to an actual object/instance of the class.

It is to be noted that non-static methods can access any static methods/variables without creating an instance of the class because the static method/variable belongs to the class.

13.What do you mean by static?

static is a keyword used for a constant variable or a method that is the same for every instance of the class.

14. What are wrapper classes in Java? Explain autoboxing and unboxing with examples.

A wrapper class is a class whose object wraps or contains primitive data types.

Autoboxing - Automatic conversion of primitive types to the object of their corresponding wrapper class is known as autoboxing.

For example : char ch = 'a';

Character a = ch;

Unboxing : Automatically converting an object of a wrapper class to its corresponding primitive type is known as unboxing.

For example : Character ch = ‘a’;

char chr = ch;

15. Is Java 100% Object-oriented?

Java is not 100% object-oriented because it makes use of primitive data types such as boolean, byte, char, int, float, double which are not objects.

16. Explain JDK, JRE and JVM.

JDK stands for Java Development Kit. It is the tool necessary to compile, document and package Java programs. It contains JRE + development tools.

JRE stands for Java Runtime Environment. A runtime environment is in which Java bytecode can be executed. It is an implementation of the JVM which physically exists.

JVM stands for Java Virtual Machine. It is an abstract machine. It is a specification that provides a runtime environment in which Java bytecode can be executed.

17. What are access and non-access modifiers?

Access modifiers - public, private, protected, default

Non-access specifiers:

- static
- final
- abstract
- synchronized - used for thread synchronization
- transient - to avoid serialization
- volatile - thread safety
- native - indicate method implemented using Java Native Interface (JNI).

18. What is an abstract class?

It is a class which can't be instantiated. It can function as a base class for subclasses. They are used to provide some common functionality across a set of related classes.

19. What is a singleton class?

It is a class whose only one instance can be created at any given time, in one JVM. A class can be made singleton by making its constructor private.

20. What is thread synchronization?

If a thread modifies an entity structurally and multiple threads access it concurrently, after which if its state remains unchanged, then the entity is said to be thread safe. Otherwise it must be synchronized externally.