## 1. What is Java?

Java is a computer programming language that is concurrent,class-based and object-oriented. The advantages of object- oriented software development are shown below:

- Modular development of code, which leads to easy maintenance and modification.
- Reusability of code.
- Improved reliability and flexibility of code.
- Increased understanding of code.

## 2. What are the concepts of OOP?

Object Oriented Programming (OOP) includes:

- Abstraction
- Encapsulation
- Polymorphism
- Inheritance
- Predefined types must be objects
- User defined types must be objects
- Operations must be performed by sending messages to objects

## 3. Mention some features of Java

Some of the features which play important role in the popularity of java are as follows:

- Object-Oriented
- Platform independent
- High Performance
- Multithreaded
- Portable & Secur

## 4. Is Java 100% Object-oriented?

Not 100%. Java does not satisfy all the OOP conditions (predefined types must be objects) because it uses eight primitive data types(Boolean, byte, char, int, float, double,long, short) which are not objects.

## 5. What is Abstraction?

Abstraction is the process of separating ideas from specificinstances and thus, develop classes in terms of their own functionality, instead of their implementation details. Java supports the creation and existence of abstract classes that expose interfaces, without including the actual implementation of all methods. The abstraction technique aims to separate the implementation details of a class from its behavior.

## 6. What is Encapsulation?

Encapsulation provides objects with the ability to hide their internal characteristics and behavior. Each object provides a number of methods, which can be accessed by other objects and change its internal data. In Java, there are three access modifiers: public, private and protected. Each modifier imposesdifferent access rights to other classes, either in the same or inexternal packages. Some of the advantages of using encapsulation are listed below:

- The internal state of every object is protected by hiding its attributes.
- It increases usability and maintenance of code, because the behavior of an object can be independently changed or extended.
- It improves modularity by preventing objects to interactwith each other, in an undesired way

## 7. What are the differences between Abstraction and Encapsulation?

Abstraction and encapsulation are complementary concepts. Onthe one hand, abstraction focuses on the behavior of an object.On the other hand, encapsulation focuses on the implementation of an object’s behavior. Encapsulation is usuallyachieved by hiding information about the internal state of an object and thus, can be seen as a strategy used in order to provide abstraction.

## 8. What is Polymorphism?

Polymorphism is the ability of programming languages to present the same interface for differing underlying data types.A polymorphic type is a type whose operations can also be applied to values of some other type.
You can see the example below where Vehicle interface has the method increaseVelocity. Truck, Train and Aeroplane implement the Vehicle Interface and the method increases the velocity to the appropriate velocity related to the vehicle type.

## 9. What are the types of Polymorphism?

There are two types of Polymorphism in Java:

- Compile-time polymorphism (Static binding) – Method overloading
- Runtime polymorphism (Dynamic binding) – Method overriding

We can perform polymorphism by Method Overloading andMethod Overriding.

| **Compile Time** | **Runtime** |
|---|---|
| Methods of a class have the same name. Each method has adifferent number of parameters .It can have parameters with different types and order. | the subclass has method with the nameas of a superclass method. It has the number of paramers, type of parameters and the return type as of a superclass method. |

| | |
|---|---|
| Method Overloading is to add to the method behavior. It can be extending to the method’s behavior. | Method Overriding is to modify the method’s behavior . |
| Overloaded methods will not have same signature. | Overridden methods will have exactlythe same signature. |
| Inheritance is not need in this case. | Inheritance is reuqired. |

## 10. What is Inheritance?

Inheritance provides an object with the ability to acquire thefields and methods of another class, called base class.
Inheritance provides reusability of code and can be used to add additional features to an existing class, without modifying it.

## 11. What is composition?

Composition is exactly like Aggregation except that the lifetimeof the ‘part’ is controlled by the ‘whole’. This control may be direct or transitive. That is, the ‘whole’ may take direct responsibility for creating or destroying the ‘part’, or it may accept an already created part, and later pass it on to some other whole that assumes responsibility for it.

## 12. What is an association?

Association represents the ability of one instance to send a message to another instance. This is typically implemented with a pointer or reference instance variable, although it mightalso be implemented as a method argument or the creation ofa local variable.

## 13. What is aggregation?

Aggregation is the typical whole/part relationship. This is exactly the same as an association with the exception thatinstances cannot have cyclic aggregation relationships.

## 14. What is JVM?

## 15. Why is Java called the Platform Independent Programming Language?

## 16. What is the Difference between JDK andJRE?

The Java Runtime Environment (JRE) is basically the Java Virtual Machine (JVM) where your Java programs are being executed. It also includes browser plugins for applet execution. The Java Development Kit(JDK) is the full-featured Software Development Kit for Java, including the JRE, the compilers and tools (like JavaDoc, and Java Debugger), in order for a user to develop, compile and execute Java applications.

| JDK | JRE |
|---|---|
| JDK stands for the term : Java Development Kit. | JRE stands for the term: Java Runtime Environment. |
| JDK is the tool for compilng, documenting and packaging Java software. | JRE is a runtime environment. JavaBytecode gets executed in the environment. |
| JDK has JRE and development tools. | JRE is a JVM implementation |

## 17. What does the static keyword mean?

The static keyword denotes that a member variable or method can beaccessed, without requiring an instantiation of the class to which it belongs.

## 18. Can you override private or static methodin Java?

A user cannot override static methods in Java, because method overriding is based upon dynamic binding at runtime and static methodsare statically bound at compile time. A static method is not associated with any instance of a class so the concept is not applicable.

## 19. Can you access the non-static variable instatic context?

A static variable in Java belongs to its class and its value remains the same for all its instances. A static variable is initialized when the class isloaded by the JVM. If your code tries to access a non-static variable, without any instance, the compiler will complain, because those variables are not created yet and they are not associated with any instance.

## 20. What are the Data Types supported by Java?

The eight primitive data types supported by the Java programming language are:

- byte
- short
- int
- long
- float
- double
- boolean
- char

## 21. What is Autoboxing and Unboxing?

Autoboxing is the automatic conversion made by the Java compiler between the primitive types and their corresponding object wrapper classes. For example, the compiler converts an int to an Integer, a double to a Double, and so on. If the conversion goes theother way, this operation is called unboxing.

## 22. What is Function Overriding and Overloading in Java?

Method overloading in Java occurs when two or more methods in the same class have the exact same name, but different parameters. On theother hand, method overriding is defined as the case when a child class redefines the same method as a parent class. Overridden methods musthave the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides.

## 23. What is a Constructor?

A constructor gets invoked when a new object is created. Every class has a constructor. In case the programmer does not provide aconstructor for a class, the Java compiler (Javac) creates a default constructor for that class.

## 24. What is Constructor Overloading?

The constructor overloading is similar to method overloading in Java. Different constructors can be created for a single class. Each constructormust have its own unique parameter list.

## 25. What is Copy-Constructor?

Finally, Java does support copy constructors like C++, but the differencelies in the fact that Java does not create a default copy constructor if you do not write your own.

## 26. Does Java support multiple inheritance?

No, Java does not support multiple inheritance. Each class is able to extend only on one class but is able to implement more than one interfaces.

## 27. What is the difference between an Interfaceand an Abstract class?

Java provides and supports the creation of both the abstract classes and interfaces. Both implementations share some common characteristics, but they differ in the following features:

- All methods in an interface are implicitly abstract. On the other hand, an abstract class may contain both abstract and non-abstract methods.
- A class may implement a number of Interfaces but can extend only one abstract class.
- In order for a class to implement an interface, it must implement all its declared methods. However, a class may not implement all declared methods of an abstract class. Though, in this case, the sub-class must alsobe declared as abstract.
- Abstract classes can implement interfaces without even providing the implementation of interface methods.
- Variables declared in a Java interface is by default final. An abstract classmay contain non-final variables.
- Members of a Java interface are public by default. A member of an abstract class can either be private, protected or public.
- An interface is absolutely abstract and cannot be instantiated. An abstract class also cannot be instantiated but can be invoked if it contains the mainmethod.

| **Interface** | **Abstract Class** |
|---|---|
| An interface has the method signatures. It does not have any implementation. | Abstract class has the abstract methods and details to be overridden. |
| A Class can implement multiple interfaces | In this case, a class can extend justone abstract class |
| Interface has all abstract methods. | Non abstract methods can be therein an abstract class. |
| Instance properties cannot be there in an interface. | Instance properties can be there inan abstract class. |
| An Interface is publicly visibleor not visible. | An abstract class can be public, private and protected visibile. |
| Any change in the interface will impact the classes implementing the interface. | Adding a method to an abstract classand implementing it does not require change in the code for derived classes. |
| An Interface cannot have constructors | An abstract class can have constructors |
| Interfaces are slow in terms of performance | Abstract classes are fast in executionof the methods in the derived classes. |

## 28. What are pass by reference and pass by value?

When an object is passed by value, this means that a copy of the objectis passed. Thus, even if changes are made to that object, it does not affect the original value. When an object is passed by reference, this means that the actual object is not passed, rather a reference of the object is passed. Thus, any changes made by the external method, are also reflected in all places.

## 29. What is the purpose of a Volatile Variable?

Volatile variable values can be modified by different threads. They willnever have the chance to block and hold a lock. Synchronization will happen whenever the variables are accessed. Using volatile may be faster than a lock, but it will not work in some situations. The range ofsituations in which volatile is effective was expanded in Java 5; in particular, double-checked locking now works correctly.

## 30. What is the purpose of a Transient variable?

A transient variable would not be serialized even if the class to which it belongs is serialized.

## 31. What is Local Variable and Instance Variable?

| **Local variable** | **Instance variable** |
|---|---|
| Local variable is declared insidea method or constructor. It can be declared within a block | Instance variable is declared inside a class. |
| Local variable need to be initialized before use. The code will not compile. | Instance variable initialization is not necessary. If not initialized, default value is used. |

## 32. What are the different access modifiers available in Java?

## 33. Difference between static binding and dynamic binding

| **Static Binding** | **Dynamic Binding** |
|---|---|
| Definition of a procedure is related to static binding | An example for dynamic binding is activation of a procedure |
| Declaration of a name for a variable is done to bind statically the variable. | Binding of a name can be dynamic bound. |

| | |
|---|---|
| The Scope of the declaration is statically bound. | Lifetime of a binding is dynamically bound. |

## 34. What are wrapper classes?

A wrapper class converts java primitives into objects. So a primitive wrapper class is a wrapper class that encapsulates, hides or wraps datatypes from the eight primitive data types so that these can be used to create instantiated objects with methods in another class or in other classes. The primitive wrapper classes are found in the Java API.

## 35. What is singleton class and how can wemake a class singleton?

In a singleton class we:

- ensure that only one instance of the singleton class ever exists
- provide global access to that instanceTo

create a singleton class we:

- declare all constructors of the class as private
- provide a static method that returns a reference to the instance