**Q1. What is Java Virtual Machine (JVM)?**

JVM phases of program execution.

Phases are as follows: we write the program, then we compile the program and at last we run the program.

1) Writing of the program is of course done by java programmer like you and me.

2) Compilation of program is done by javac compiler, javac is the primary java compiler included in java development kit (JDK). It takes java program as input and generates java bytecode as output (.class).

3) In third phase, JVM executes the bytecode generated by compiler. This is called program run phase.

Bytecode: Machine-independent code generated by the Java (TM) compiler and executed by the Java interpreter.

JAR Files (.jar) Java ARchive. A file format used for aggregating many files into one.

https://www.javacodemonk.com/difference-between-jdk-jre-and-jvm-6380989d

**Q2. What is Java Development Kit (JDK)?**

While explaining JVM and bytecode, I have used the term JDK. Let’s discuss about it. As the name suggests this is complete java development kit that includes JRE (Java Runtime Environment), compilers and various tools like JavaDoc, Java debugger etc. **In order to create, compile and run Java program you would need JDK installed on your computer.**

**Q3. What is Java Runtime Environment (JRE)?**

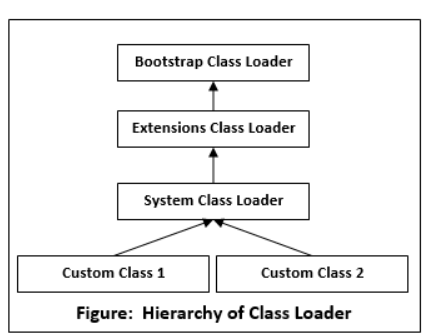
JRE is a part of JDK which means that JDK includes JRE. When you have JRE installed on your system, you can run a java program however you won’t be able to compile it. JRE includes JVM, browser plugins and applets support. When you only need to run a java program on your computer, you would only need JRE.

**Q4. What gives Java its 'write once and run anywhere' nature? (10)**

The bytecode. Java compiler converts the Java programs into the class file (Byte Code) which is the intermediate language between source code and machine code. This bytecode is not platform specific and can be executed on any computer.

**Q5. What is classloader? (11)**

Classloader is a subsystem of JVM which is used to load class files. Whenever we run the java program, it is loaded first by the classloader. There are three built-in classloaders in Java.



**Q6. Is Empty .java file name a valid source file name? (12)**

Yes, Java allows to save our java file by .java only, we need to compile it by javac .java and run by java classname. Let's take a simple example:

//save by .java only

class A{

public static void main(String args[]){

System.out.println("Hello java");

}

}

//compile by javac .java

//run by java A

**Q7. Why Java is platform independent?**

Java is a platform independent because java can be run on any platform or operative system without any change.

**Q8. Why Java is not 100% Object-oriented?**

Java is not 100% object oriented because it supports Primitive datatype such as int, byte, long... etc, to be used, which are not objects.

**Q9. If I don't provide any arguments on the command line, then what will be the value stored in the String array passed into the main () method, empty or NULL? (14)**

It is empty, but not null.

**Q10. What if I write static public void instead of public static void? (15)**

The program compiles and runs correctly because the order of specifiers doesn't matter in Java.

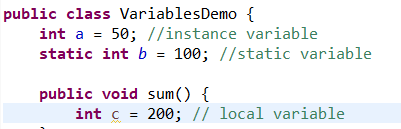
**Q11. What is the default value of the local variables? (16)**

The local variables are not initialized to any default value, neither primitives nor object references.

**Q12. What is variable in Java?**

A variable is a container which holds the data values. A variable is assigned with a data type.

Int data = 50; Here data is variable. There are three types of variables:



**1) Local Variable:**

* A variable declared inside the body of the method is called local variable.
* You can use this variable only within that method and the other methods in the class aren't even aware that the variable exists.
* **A local variable cannot be defined with "static" keyword because we can’t use it outside the method.**

**2) Instance or Global Variable:**

* A variable declared inside the class but outside the body of the method, is called instance variable. It is not declared as static.
* It is called instance variable because its value is instance specific and is not shared among instances.

**3) Static variable**

* A variable which is declared as static is called static variable. It cannot be local.
* **You can create a single copy of static variable and share among all the instances of the class.**
* **Memory allocation for static variable happens only once when the class is loaded in the memory.**

**Q13. Why a static block executes before the main method?**

* A class has to be loaded in main memory before we start using it. Static block is executed during class loading.
* This is the reason why a static block executes before the main method.

**Q14. What is the Difference Between static and final in Java? See another sheet**

**Q15. What are the advantages of Packages in Java? (19)**

There are various advantages of defining packages in Java.

* Packages avoid the name clashes.
* The Package provides easier access control.
* We can also have the hidden classes that are not visible outside and used by the package.
* It is easier to locate the related classes.

**Q16. What will be the initial value of an object reference which is defined as an instance variable? (26)**

All object references are initialized to null in Java.

**Q17. What is the static variable? (39)**

The static variable is used to refer to the common property of all objects (that is not unique for each object), e.g., The company name of employees, college name of students, etc. Static variable gets memory only once in the class area at the time of class loading. Using a static variable makes your program more memory efficient (it saves memory). Static variable belongs to the class rather than the object.

**Q18. What is the static method? (40)**

A static method belongs to the class rather than the object.

There is no need to create the object to call the static methods.

**A static method can access and change the value of the static variable.**

**Q19. What are the restrictions that are applied to the Java static methods? (41)**

Two main restrictions are applied to the static methods.

The static method cannot use non-static data member or call the non-static method directly.

this and super cannot be used in static context as they are non-static.

**Q20. Why is the main method static? (42)**

Because the object is not required to call the static method. If we make the main method non-static, JVM will have to create its object first and then call main() method which will lead to the extra memory allocation.

**Q21. What is the static block? (44)**

Static block is used to initialize the static data member. It is executed before the main method, at the time of classloading.

**Q22. Can we execute a program without main () method? (45)**

No, It was possible before JDK 1.7 using the static block. Since JDK 1.7, it is not possible.

**Q23. What if the static modifier is removed from the signature of the main method? (46)**

Program compiles. However, at runtime, it throws an error "NoSuchMethodError."

**Q24. How can constructor chaining be done using this keyword? (55)**

Constructor chaining enables us to **call one constructor from another constructor of the class with respect to the current class object.** We can use this keyword to perform constructor chaining within the same class. Consider the following example which illustrates how can we use this keyword to achieve constructor chaining.

https://www.javatpoint.com/corejava-interview-questions

#### Q25. What are the advantages of passing this into a method instead of the current class object itself? (56)

As we know, that this refers to the current class object, therefore, it must be similar to the current class object. However, there can be two main advantages of passing this into a method instead of the current class object.

this is a final variable. Therefore, this cannot be assigned to any new value whereas the current class object might not be final and can be changed. this can be used in the synchronized block.

**Q26. What is aggregation? (61)**

Aggregation can be defined as the relationship between two classes where the aggregate class contains a reference to the class it owns. Aggregation is best described as a has-a relationship. For example, the aggregate class Employee having various fields such as age, name, and salary also contains an object of Address class having various fields such as Address-Line 1, City, State, and pin-code. In other words, we can say that Employee (class) has an object of Address class. Consider the following example.

https://www.javatpoint.com/corejava-interview-questions

**Q27. What is composition? (62)**

Holding the reference of a class within some other class is known as composition. When an object contains the other object, if the contained object cannot exist without the existence of container object, then it is called composition. In other words, we can say that composition is the particular case of aggregation which represents a stronger relationship between two objects. Example: A class contains students. A student cannot exist without a class. There exists composition between class and students.

**Q28. What is the difference between aggregation and composition? (63)**

Aggregation represents the weak relationship whereas composition represents the strong relationship. For example, the bike has an indicator (aggregation), but the bike has an engine (composition).

**Q29. Why does Java not support pointers? (64)**

The pointer is a variable that refers to the memory address. They are not used in Java because they are unsafe(unsecured) and complex to understand.

**Q30. How can constructor chaining be done by using the super keyword? (66)**

**Q31. Can you use this () and super () both in a constructor? (70)**

No, because this () and super () must be the first statement in the class constructor.

**Q32. What is object cloning? (71)**

The object cloning is used to create the exact copy of an object. The clone () method of the Object class is used to clone an object. The java. lang. Cloneable interface must be implemented by the class whose object clone we want to create. If we don't implement Cloneable interface, clone () method generates CloneNotSupportedException.

**Q33. Can we change the scope of the overridden method in the subclass? (84)**

Yes, we can change the scope of the overridden method in the subclass. However, we must notice that we cannot decrease the accessibility of the method. The following point must be taken care of while changing the accessibility of the method.

The private can be changed to protected, public, or default.

The protected can be changed to public or default.

The default can be changed to public.

The public will always remain public.

**Q34. Can we modify the throws clause of the superclass method while overriding it in the subclass? (85)**

Yes, we can modify the throws clause of the superclass method while overriding it in the subclass. However, there are some rules which are to be followed while overriding in case of exception handling.

If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception, but it can declare the unchecked exception.

If the superclass method declares an exception, subclass overridden method can declare same, subclass exception or no exception but cannot declare parent exception.

**Q35. Can you have virtual functions in Java? (87)**

Yes, all functions in Java are virtual by default.

Now, since java5, it is possible to override any method by changing the return type if the return type of the subclass overriding method is subclass type. It is known as covariant return type.

**Q36. What is the difference between the final method and abstract method? (100)**

The main difference between the final method and abstract method is that the abstract method cannot be final as we need to override them in the subclass to give its definition.

**Q37. What is Java instance Of operator? (106)**

The instanceof in Java is also known as type comparison operator because it compares the instance with type. It returns either true or false. If we apply the instanceof operator with any variable that has a null value, it returns false.

**Q38. How to make a read-only class in Java? (121)**

A class can be made read-only by making all of the fields private. The read-only class will have only getter methods which return the private property of the class to the main method. We cannot modify this property because there is no setter method available in the class. Consider the following example.

**Q39. How to make a write-only class in Java? (122)**

A class can be made write-only by making all of the fields private. The write-only class will have only setter methods which set the value passed from the main method to the private fields. We cannot read the properties of the class because there is no getter method in this class.

**Q40. How can we access some class in another class in Java? (127)**

There are two ways to access a class in another class.

* By using the fully qualified name: To access a class in a different package, either we must use the fully qualified name of that class, or we must import the package containing that class.
* By using the relative path, We can use the path of the class that is related to the package that contains our class. It can be the same or subpack age.

**Q41. Do I need to import java.lang package any time? Why? (128)**

No. It is by default loaded internally by the JVM.

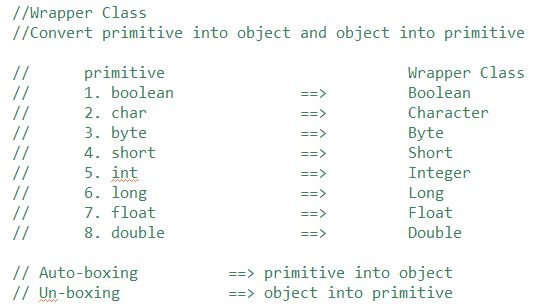
**Q42. Can I import same package/class twice? Will the JVM load the package twice at runtime? (129)**

One can import the same package or the same class multiple times. Neither compiler nor JVM complains about it. However, the JVM will internally load the class only once no matter how many times you import the same class.

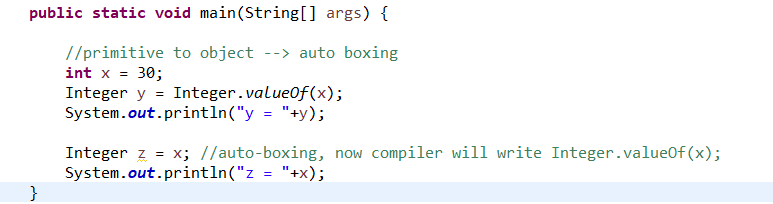
**Q43. What is the static import? (130)**

By static import, we can access the static members of a class directly, and there is no to qualify it with the class name.

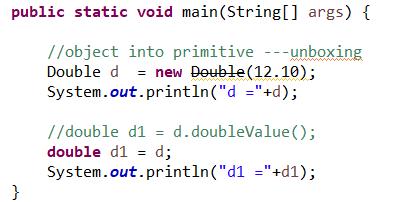
**Q.44. What is Wrapper Class?**

****

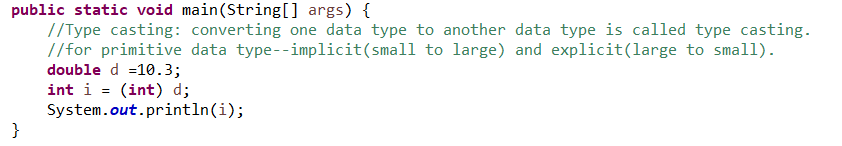
**Example-1:**

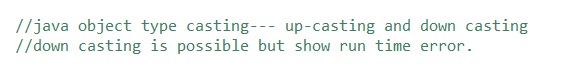
****

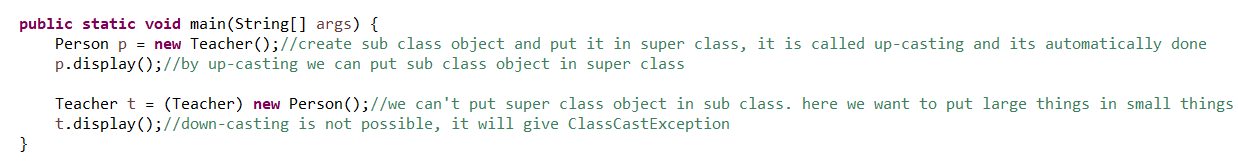
**Example-2:**

****

**Q.45. What is Casting in Java?**

****

****

****

**Q.46 What is escape sequence?**

A special character followed/preceded by a blackslash (\).

**Q.47. What is ENUM in Java?**

* The Enum in Java is a data type which contains a fixed set of constants. You can create your own data type

Or

* An enum is a special "class" that represents a group of constants (unchangeable variables, like final variables).
* It can be used for days of the week (SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, and SATURDAY).
* Directions (NORTH, SOUTH, EAST, and WEST), season (SPRING, SUMMER, WINTER, and AUTUMN or FALL).
* Colors (RED, YELLOW, BLUE, GREEN, WHITE, and BLACK) etc. According to the Java naming conventions.
* We should have all constants in capital letters. So, we have enum constants in capital letters.
* Java Enums can be thought of as classes which have a fixed set of constants (a variable that does not change).
* The Java enum constants are static and final implicitly. It is available since JDK 1.5.
* Enum can be traversed.
* Enum can have fields, constructors and methods.
* Enum may implement many interfaces but cannot extend any class because it internally extends Enum class.

**Q.48. Difference between Enums and Classes.**

An enum can, just like a class, have attributes and methods. **The only difference is that enum constants are public, static and final (unchangeable - cannot be overridden).**

An enum cannot be used to create objects, and it cannot extend other classes (but it can implement interfaces).

**Q49. Why and When to Use Enums?**

Use enums when you have values that you know aren't going to change, like month days, days, colors, deck of cards, etc.

**Q50. What is difference between path and classpath variables?**

The PATH variable gives the location of executables like javac, java etc. It is possible to run a program without specifying the PATH but you will need to give full path of executable like **C:\Program Files\Java\jdk1.8.0\_271\bin\javac A.java** instead of simple **javac A.java**

The CLASSPATH variable gives location of the Library Files.

**Q.51. How is Java platform independent if JVM is platform dependent?**

JVM, JRE and JDK are platform dependent because configuration of each OS differs. **The use of the same byte code for all JVMs on all platforms** make java platform independent. Or java is platform independent because java does not run directly on operating system. It runs on the JVM which you have to install separately.

**An important point to be noted is that while JAVA is platform-independent language, the JVM is platform-dependent. Different JVM is designed for different OS and byte code is able to run on different OS.**

**Q.52 Java memory management?**

Java memory is divided into two parts:

1. **Heap Memory:** Store objects, JRE classes, GC runs on heap memory only to free some memory by destroying some unused objects which doesn’t have any references.
2. **Stack Memory:**

* Used to execute threads
* Methods specific values
* Local variables
* **Object references which are referring some objects on heap memory.**
* **It maintains LIFO.**

**Q. 53. When to use transient variable in java?**

Transient is a variables modifier used in serialization. At the time of serialization, **if we don't want to save value of a particular variable in a file**, then we use transient keyword. When JVM comes across transient keyword, it ignores original value of the variable and save default value of that variable data type.

**Q. 54 Which one will take more memory int or Integer?**

An Integer object will take more memory an Integer is an object and it store meta data overhead about the object and int is primitive type so it takes less space.

**int** helps in storing integer value into memory. ... On other hand Integer is an object which takes 128 bits (16 bytes) to store its int value.

**Q. 55 The difference between Serial and Parallel Garbage Collector?**

The parallel collector is also known as throughput collector, it's a generational collector similar to the serial collector. The primary difference between the serial and parallel collectors is **that the parallel collector has multiple threads that are used to speed up garbage collection. Serial collector uses one thread to execute garbage collection.**

Serial GC is the garbage collector of choice for applications that do not have low pause time requirements and run-on client-style machines.

**Q. 56. What is Serialization and Deserialization?**

Serialization is a mechanism of converting the state of an object into a byte stream. Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory.

For serializing the object, we call the **writeObject ()** method of ObjectOutputStream class, and for deserialization we call the **read Object ()** method of ObjectInputStream class.

If you want a class object to be serializable, all you need to do it implement the java.io.Serializable interface. Serializable in java is a marker interface and has no fields or methods to implement.

Serialization in java is implemented by ObjectInputStream and ObjectOutputStream,

**Q. 57. What is Synchronization in Java?**

Synchronization in java is **the capability to control the access of multiple threads to any shared resource**. In the Multithreading concept, multiple threads try to access the shared resources at a time to produce inconsistent results. The synchronization is necessary for reliable communication between threads.

**Q.58. What is lambda expression in Java?**

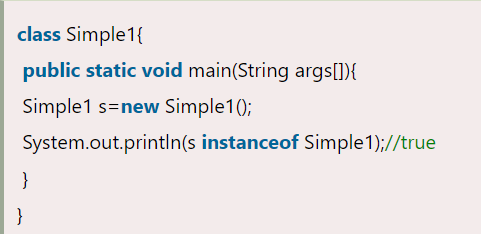
Lambda Expressions were added in Java 8.

A lambda expression is a short block of code which takes in parameters and returns a value. Lambda expressions are similar to methods, but they do not need a name and they can be implemented right in the body of a method.

*parameter* -> *expression*

**Q.59. What is Java instanceOf operator? (106)**

**The instanceof in Java is also known as type comparison operator because it compares the instance with type. It returns either true or false.** If we apply the instanceof operator with any variable that has a null value, it returns false. Consider the following example.

****

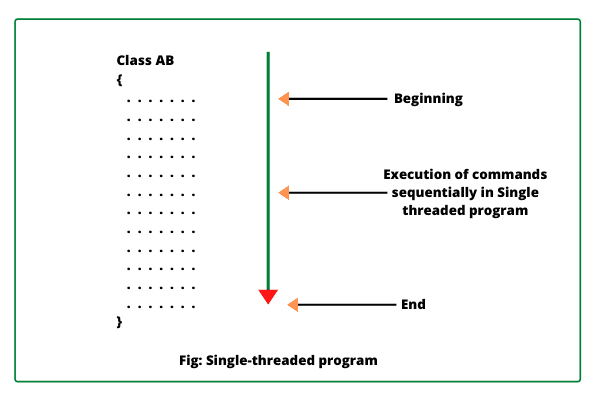
**Q.60. What is Thread?**

A **thread in Java** simply represents a single independent path of execution of a group of statements. It is the flow of execution, from beginning to end, of a task.

When we write a group of statements in a program, these statements are executed by JVM one by one. This execution process is called thread in Java.

There is always at least one thread running internally in every program and this thread is used by JVM to execute statements in the program.

When a program contains a single flow of control, it is called single-threaded program. In a single thread program, there is a beginning, a body, and an end, and execute commands sequentially. Look at the below figure.



We can also create more than one execution thread in a program that can be used to perform multiple tasks simultaneously.

**When we create more than one thread in a program, each thread has its own path of execution and all the threads share the same memory address space, data, and code in a program.**

**Q.61. What is process in Java?**

A process is a program that executes as a single thread. In other words, when an executable program is loaded into memory, it is called process.

**Q. 62. What is Main Thread in Java?**

Every Java program has always at least one thread, even if you do not create any thread. This thread is called main thread. The main thread is also called parent thread and the rest of threads that are generated from it are called child threads of the program.

Main thread is the last thread to be executed in a program. When main thread finishes the execution, the program terminates immediately. Whenever Java program starts, main thread is created automatically.

Thread obj = Thread.currentThread();

**Q. 63. What is Multithreading in Java?**

Multithreading means multiple threads of execution concurrently. The process of executing multiple threads simultaneously (concurrently) is called **multithreading in Java**. Each thread runs parallel to each other. Multiple threads don’t allocate separate memory area; hence they save memory. Also, context switching between threads takes less time.

**Advantages of multithread:**

The users are not blocked because threads are independent, and we can perform multiple operations at times. **As such the threads are independent, the other threads won’t get affected if one thread meets an exception.**

**Q.64. What is singleton class in Java and how can we make a class singleton?**

In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time.

After first time, if we try to instantiate the Singleton class, the new variable also points to the first instance created.

To design a singleton class:

1. Make constructor as private.

2. Write a static method that has return type object of this singleton class**.**

**Q.65. What is the difference between equals () and == in Java?**

**equals () is a method and == is compare 2 values.**

**Q.66. What is Object Oriented Programming?**

Object Oriented Programming is a programming methodology based on objects, instead of functions and procedures. These objects are organized into classes, which allow individual objects to be group together.

**Q.67. Differentiate between static and non-static methods in Java.**

Static --> We can call the method by its class name

Non-Static method-->We need to create object to call the method

**Q.68. What is collection class in Java? List down its methods and interfaces**

(Collections is a framework)

Collection is an interface which is implemented by all the classes in the collection framework. It declares the methods that every collection will have. In other words, we can say that the Collection interface builds the foundation on which the collection framework depends.

Some of the methods of Collection interface are Boolean add ( Object obj), Boolean addAll ( Collection c), void clear(), etc. which are implemented by all the subclasses of Collection interface.

**Q69. What are the different Java exceptions?**

ArithmeticException. It is thrown when an exceptional condition has occurred in an arithmetic operation.

ArrayIndexOutOfBoundsException. ...

ClassNotFoundException. ...

FileNotFoundException. ...

IOException. ...

InterruptedException. ...

NoSuchFieldException. ...

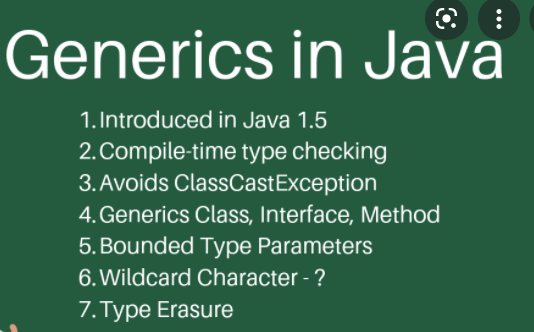
NoSuchMethodException.

**Q70. Why java is Threadsafe?**

1) Immutable objects are by default thread-safe because their state cannot be modified once created. Since String is immutable in Java, it's inherently thread-safe. 2) Read-only or final variables are also thread-safe in Java. 3) Locking is one way of achieving thread-safety in Java.

**Q71. What is Generics?**

Java Generics is **a set of related methods or a set of similar types**. Generics allow types Integer, String, or even user-defined types to be passed as a parameter to classes, methods, or interfaces. Generics are mostly used by classes like HashSet or HashMap.

****

**Q72. What is multiple inheritance? Is it supported by Java?**

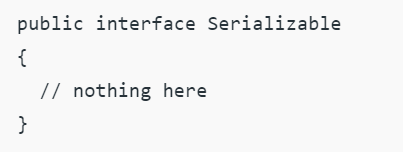
If a child class inherits the property from multiple classes is known as multiple inheritance. Java does not allow to extend multiple classes.

The problem with multiple inheritance is that if multiple parent classes have the same method name, then at runtime it becomes difficult for the compiler to decide which method to execute from the child class.

Therefore, Java doesn’t support multiple inheritance. The problem is commonly referred to as Diamond Problem.

**Q73. What is a marker interface?**

A Marker interface can be defined as the interface having no data member and member functions. In simpler terms, an empty interface is called the Marker interface. The most common examples of Marker interface in Java are Serializable, Cloneable etc. The marker interface can be declared as follows.



**Q74. What is a copy constructor in Java?**

Copy constructor is a member function that is used to initialize an object using another object of the same class. Though there is no need for copy constructor in Java since all objects are passed by reference. Moreover, Java does not even support automatic pass-by-value.

https://www.tutorialspoint.com/how-to-access-the-members-of-a-class-from-another-class-in-java