TOP 50

# JAVA

## INTERVIEW QUESTION

Java

Created by- TOPPERWORLD

## Q 1. Is Java Platform Independent if then how?

**Ans:** Yes, Java is a Platform Independent language.

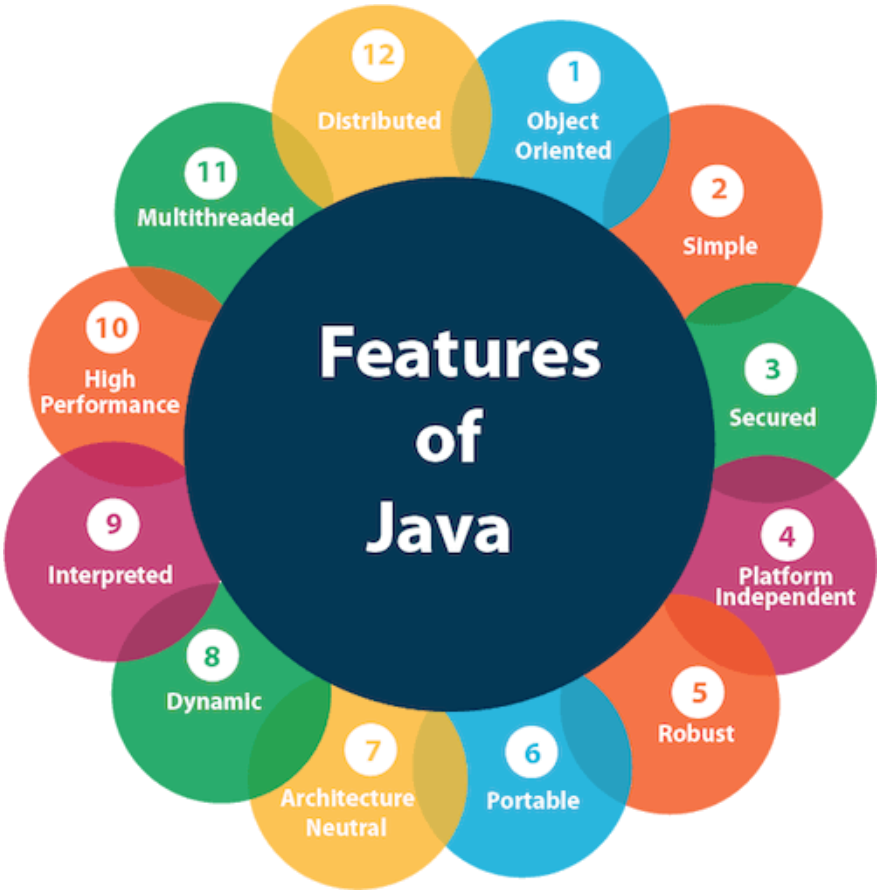
Unlike many programming languages javac compiler compiles the program to form a bytecode or .class file.

This file is independent of the software or hardware running but needs a JVM(Java Virtual Machine) file preinstalled in the operating system for further execution of the bytecode.

Although **JVM is platform dependent**, the bytecode can be created on any System and can be executed in any other system despite hardware or software being used which makes Java platform independent.

## Q 2. What are the top Java Features?

**Ans:** Java is one the most famous and most used language in the real world, there are many features in Java that makes it better than any other language some of them are mentioned below..

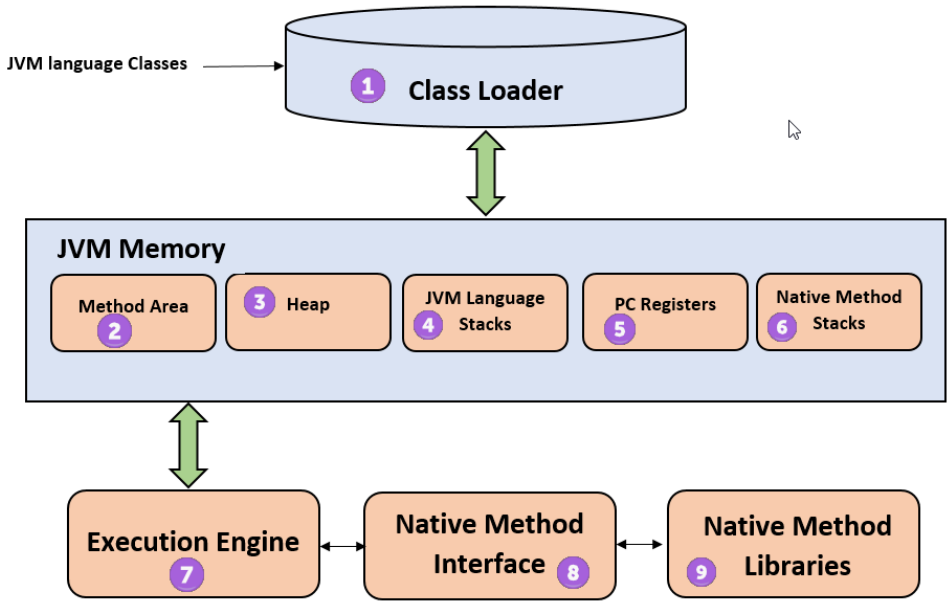



## Q 3. What is JVM?

**Ans:** JVM stands for Java Virtual Machine it is a Java interpreter.

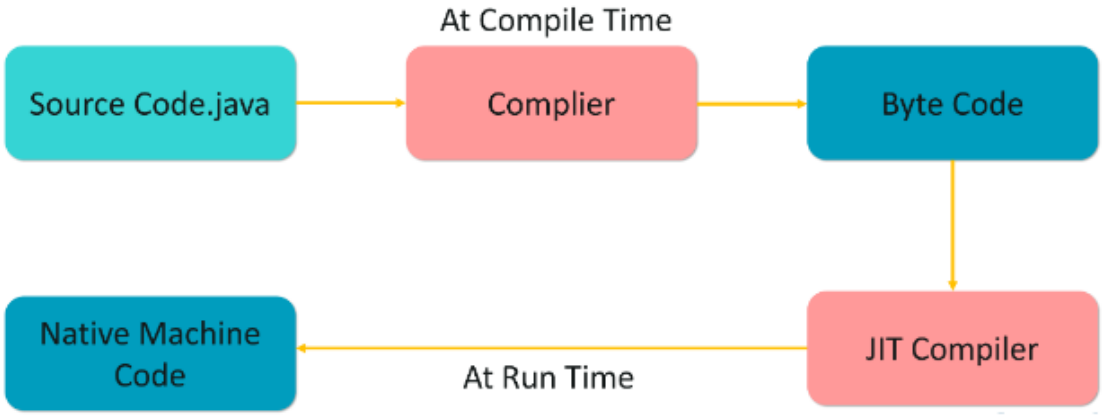
It is responsible for loading, verifying, and executing the bytecode created in Java.

Although it is platform dependent which means the software of JVM is different for different Operating Systems it plays a vital role in making Java platform Independent.




## Q 4. What is JIT?

## Ans:



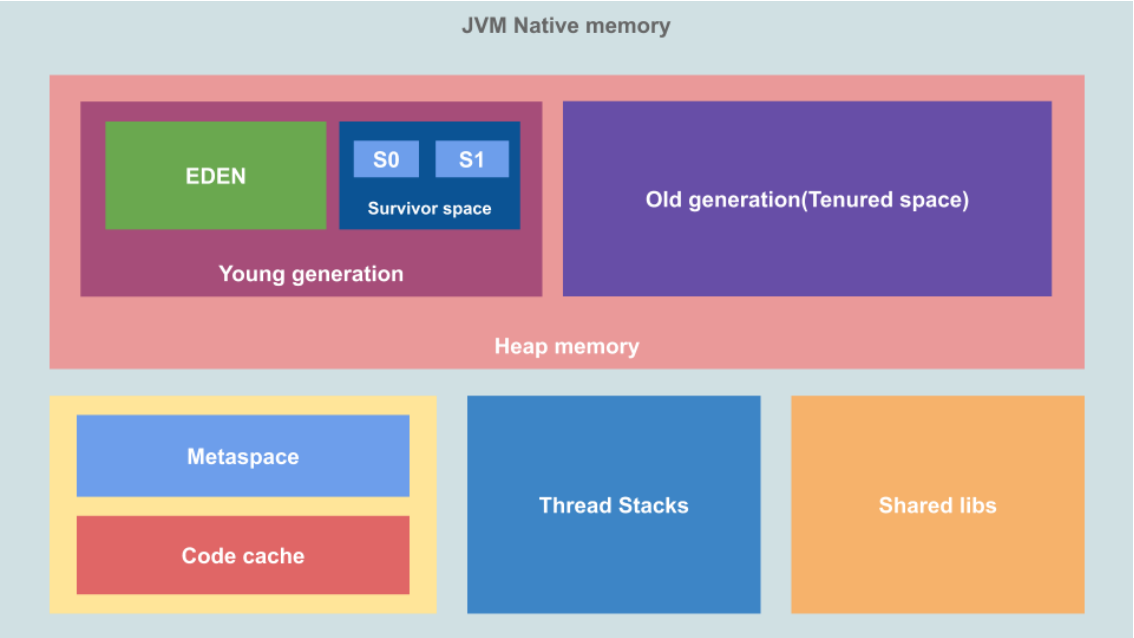

JIT stands for (Just-in-Time) compiler is a part of JRE(Java Runtime Environment), it is used for better performance of the Java applications during run-time. The use of JIT is mentioned in step by step process mentioned below:

1) Source code is compiled with **javac** compiler to form bytecode
2) Bytecode is further passed on to JVM
3) JIT is a part of JVM, JIT is responsible for compiling bytecode into native machine code at run time.

4) The JIT compiler is enabled throughout, while it gets activated when a method is invoked. For a compiled method, the JVM directly calls the compiled code, instead of interpreting it.
5) As JVM calls the compiled code that increases the performance and speed of the execution.

## Q 5. What are Memory storages available with JVM?

**Ans :**




**JVM consists of a few memory storages as mentioned below:**

- Class(Method) Area: stores class-level data of every class such as the runtime constant pool, field, and method data, and the code for methods.
- Heap: Objects are created or objects are stored. It is used to allocate memory to objects during run time.
- Stack: stores data and partial results which will be needed while returning value for method and performing dynamic linking
- Program Counter Register: stores the address of the Java virtual machine instruction currently being executed.
- Native Method Stack: stores all the native methods used in the application.

## Q 6.Difference between JVM, JRE, and JDK.

**Ans:** **JVM**: JVM also known as Java Virtual Machine is a part of JRE. JVM is a type of interpreter responsible for converting bytecode into machine-readable code. JVM itself is platform dependent but it interprets the bytecode which is the platform-independent reason why Java is platform-independent.

**JRE**: JRE stands for Java Runtime Environment, it is an installation package that provides an environment to run the Java program or application on any machine.

**JDK**: JDK stands for Java Development Kit which provides the environment to develop and execute Java programs. JDK is a package that includes two things Development Tools to provide an environment to develop your Java programs and, JRE to execute Java programs or applications.

## Q 7.What is a classloader?

**Ans :** Classloader is the part of JRE(Java Runtime Environment), during the execution of the bytecode or created .

class file classloader is responsible for dynamically loading the java classes and interfaces to JVM(Java Virtual Machine).

Because of classloaders Java run time system does not need to know about files and file systems.

## Q 8. What are the differences between Java and C++?

**Ans :**

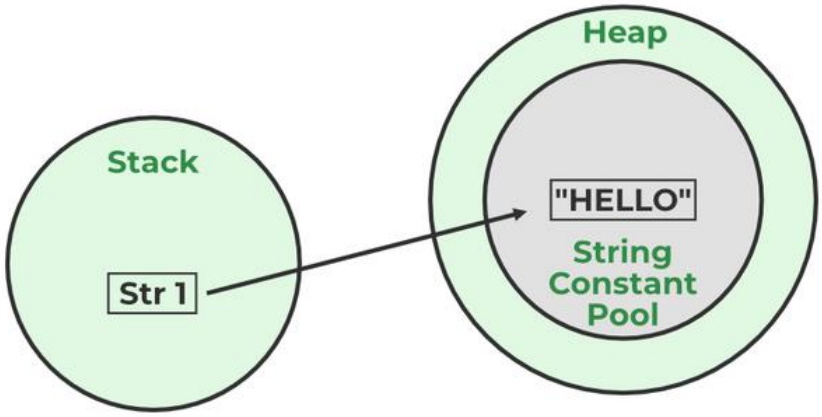
| Basis | C++ | Java |
|---|---|---|

| Basis | C++ | Java |
|---|---|---|
| **Platform** | C++ is Platform Dependent. | Java is Platform Independent. |
| **Application** | C++ is mainly used for System Programming | Java is Mainly used for Application Programming |
| **Hardware** | C++ is nearer to hardware | Java is not so interactive with hardware |
| **Global Scope** | C++ supports global and namespace scope. | Java doesn't support global scope. |
| **Not Supporting** | Functionality supported in Java but not in C++ are:<br>• thread support<br>• documentation comment<br>• unsigned right shift(>>>) | Functionality supported in C++ but not in Java are:<br>• goto<br>• Pointers<br>• Call by reference<br>• Structures and Unions<br>• Multiple Inheritance<br>• Virtual Functions |
| **OOPS** | C++ is an object-oriented language. It is not a single root hierarchy . | Java is also an object-oriented language. It is a single root hierarchy as everything gets |

| Basis | C++ | Java |
|---|---|---|
| | | derived from a single class (java.lang.Object). |
| **Inheritance Tree** | C++ always creates a new inheritance tree. | Java uses a Single inheritance tree as classes in Java are the child of object classes in Java. |

## Q 9. What is Java String Pool?

**Ans :** A Java String Pool is a place in heap memory where all the strings defined in the program are stored. A separate place in a stack is there where the variable storing the string is stored. Whenever we create a new string object, JVM checks for the presence of the object in the String pool, If String is available in the pool, the same object reference is shared with the variable, else a new object is created.



## Example:

```
String str1="Hello";
// "Hello" will be stored in String Pool
// str1 will be stored in stack memory
```

## Q 10. What will happen if we declare don ' t declare the main as static?

**Ans:** We can declare the main method without using static and without getting any errors. But, the main method will not be treated as the entry point to the application or the program.

## Q 11. Explain public static void main(String args[]) in Java.

**Ans :** Unlike any other programming language like C, C++, etc. In Java, we declared the main function as a public static void main (String args[]). The meanings of the terms are mentioned below:

1) **public**: the public is the access modifier responsible for mentioning who can access the element or the method and what is the limit. It is responsible for making the main function globally available. It is made public so that JVM can invoke it from outside the class as it is not present in the current class.
2) **static**: static is a keyword used so that we can use the element without initiating the class so to avoid the unnecessary allocation of the memory.
3) **void**: void is a keyword and is used to specify that a method doesn ' t return anything. As the main function doesn ' t return anything we use void.
4) **main**: main represents that the function declared is the main function. It helps JVM to identify that the declared function is the main function.
5) **String args[]**: It stores Java command-line arguments and is an array of type java.lang.String class.

## Q 12. What are the advantages of Packages in Java?

**Ans :** There are various advantages of defining packages in Java.

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

## Q 13. Explain different data types in Java.

**Ans:** There are 2 types of data types in Java as mentioned below:

1. Primitive Data Type
2. Non-Primitive Data Type or Object Data type

Primitive Data Type: Primitive data are single values with no special capabilities. There are 8 primitive data types:

- **boolean**: stores value true or false
- **byte**: stores an 8-bit signed two's complement integer
- **char**: stores a single 16-bit Unicode character
- **short**: stores a 16-bit signed two's complement integer
- **int**: stores a 32-bit signed two's complement integer
- **long**: stores a 64-bit two's complement integer
- **float**: stores a single-precision 32-bit IEEE 754 floating-point
- **double**: stores a double-precision 64-bit IEEE 754 floating-point

Non-Primitive Data Type: Reference Data types will contain a memory address of the variable's values because it is not able to directly store the values in the memory. Types of Non-Primitive are mentioned below:

- Strings
- Array
- Class
- Object
- Interface

## Q 14. When a byte datatype is used?

**Ans :** A byte is an 8-bit signed two-complement integer. The minimum value supported by bytes is -128 and 127 is the maximum value. It is used in conditions where we need to save memory and the limit of numbers needed is between -128 to 127.

## Q 15. What is the Wrapper class in Java?

**Ans:** Wrapper, in general, is referred to a larger entity that encapsulates a smaller entity. Here in Java, the wrapper class is an object class that encapsulates the primitive data types.

The primitive data types are the ones from which further data types could be created. For example, integers can further lead to the construction of long, byte, short, etc. On the other hand, the string cannot, hence it is not primitive.

Getting back to the wrapper class, Java contains 8 wrapper classes. They are Boolean, Byte, Short, Integer, Character, Long, Float, and Double. Further, custom wrapper classes can also be created in Java which is similar to the concept of Structure in the C programming language. We create our own wrapper class with the required data types.

## Q 16. Differentiate between instance and local variables.

**Ans :**

| Instance Variable | Local Variable |
|---|---|
| Declared outside the method, directly invoked by the method. | Declared within the method. |
| Has a default value. | No default value |

| Instance Variable | Local Variable |
|---|---|
| It can be used throughout the class. | The scope is limited to the method. |

## Q 17. What are the default values assigned to variables and instances in Java?

**Ans :** In Java When we haven ' t initialized the instance variables then the compiler initializes them with default values. The default values for instances and variables depend on their data types. Some common types of default data types are:

- The default value for numeric types (byte, short, int, long, float, and double) is 0.
- The default value for the boolean type is false.
- The default value for object types (classes, interfaces, and arrays) is null.

## Example:

```
import java.io.*;
class TW  {
        static byte b;
        static int i;
        static long l;
        static short s;
        static boolean bool;
        static char c;
        static String str;
        static Object object;
          static float f;
        static double d;        static int[] Arr;
        public static void main(String[] args)      {
                System.out.println("byte value" + b);
                System.out.println("short value" + s);
                System.out.println("int value" + i);
                System.out.println("long value" + l);
                System.out.println("boolean value" + bool);
                System.out.println("char value" + c);
                System.out.println("float value" + f);
                System.out.println("double value" + d);
                System.out.println("string value" + str);
                System.out.println("object value" + object);
                System.out.println("Array value" + Arr);
        }
}
```

## Output:

```
byte value0
short value0
int value0
long value0
boolean valuefalse
char value
float value0.0
double value0.0
string valuenull
object valuenull
Array valuenull
```

## Q 18. Why do we need wrapper classes?

**Ans :** The wrapper class is an object class that encapsulates the primitive data types, and we need them for the following reasons:

1. Wrapper classes are final and immutable
2. Provides methods like valueOf(), parseInt(), etc.
3. It provides the feature of autoboxing and unboxing.

## Q 19. What is a Class Variable?

### Ans :

In Java, a class variable (also known as a static variable) is a variable that is declared within a class but outside of any method, constructor, or block. Class variables are declared with the static keyword, and they are shared by all instances (objects) of the class as well as by the class itself. No matter how many objects are derived from a class, each class variable would only exist once.

### Example:

```
// Java program to demonstrate use of Class Variable
class TW {
        public static int ctr = 0;
        public TW() { ctr++; }
        public static void main(String[] args)
        {
                TW obj1 = new TW();
                TW obj2 = new TW();
                TW obj3 = new TW();
                System.out.println("Number of objects created
are "
                                                + TW.ctr);
        }
}
```

### Output:

```
Number of objects created are 3
```

## Q 20. What is the default value stored in Local Variables?

**Ans :** There is no default value stored with local variables. Also, primitive variables and objects don't have any default values.

## Q 21. Explain the difference between instance variable and a class variable.

## Ans:

### ➢ Instance Variable :

A class variable without a static modifier known as an instance variable is typically shared by all instances of the class.

These variables can have distinct values among several objects.

The contents of an instance variable are completely independent of one object instance from another because they are related to a specific object instance of the class.

## Example:

```
// Java Program to demonstrate Instance Variable
import java.io.*;
class TW {
        private String name;
        public void setName(String name) { this.name = name; }
        public String getName() { return name; }
        public static void main(String[] args)
        {
                TW obj = new TW();
                obj.setName("John");
                System.out.println("Name " + obj.getName());
        }
}
```

## Output:

```
Name John
```

### ➢ Class Variable:

Class Variable variable can be declared anywhere at the class level using the keyword static.

These variables can only have one value when applied to various objects.

These variables can be shared by all class members since they are not connected to any specific object of the class.

## Example:

```
// Java Program to demonstrate Class Variable
import java.io.*;
class TW {
        // class variable
        private static final double PI = 3.14159;
        private double radius;
        public TW(double radius) { this.radius = radius; }
        public double getArea() { return PI * radius * radius; }
        public static void main(String[] args)
        {
                TW obj = new TW(5.0);
                System.out.println("Area of circle: "
                                                + obj.getArea());
        }
}
```

## Output:

```
Area of circle: 78.53975
```

## Q 22. What is a static variable?

**Ans :** The static keyword is used to share the same variable or method of a given class. Static variables are the variables that once declared then a single copy of the variable is created and shared among all objects at the class level.

## Q 23. What are the super most classes for all the streams?

**Ans :** All the stream classes can be divided into two types of classes that are ByteStream classes and CharacterStream Classes.

The ByteStream classes are further divided into InputStream classes and OutputStream classes. CharacterStream classes are also divided into Reader classes and Writer classes.

The SuperMost classes for all the InputStream classes is java.io.InputStream and for all the output stream classes is java.io.OutPutStream.

Similarly, for all the reader classes, the super-most class is java.io.Reader, and for all the writer classes, it is java.io.Writer.

## Q 24. What is the purpose of using BufferedInputStream and BufferedOutputStream classes?

**Ans :** When we are working with the files or stream then to increase the Input/Output performance of the program we need to use the BufferedInputStream and BufferedOutputStream classes.
These both classes provide the capability of buffering which means that the data will be stored in a buffer before writing to a file or reading it from a stream.
It also reduces the number of times our OS needs to interact with the network or the disk.
Buffering allows programs to write a big amount of data instead of writing it in small chunks. This also reduces the overhead of accessing the network or the disk.

## Q 25. What are FilterStreams?

**Ans :** **Stream filter or Filter Streams** returns a stream consisting of the elements of this stream that match the given predicate.

While working filter() it doesn't actually perform filtering but instead creates a new stream that, when traversed, contains the elements of initial streams that match the given predicate.

## Q 26. What is covariant return type?

**Ans :** The covariant return type specifies that the return type may vary in the same direction as the subclass.

It is possible to have different return types for an overriding method in the child class, but the child's return type should be a subtype of the parent's return type and because of that overriding method becomes variant with respect to the return type.

We use covariant return type because of the following reasons:

- Avoids confusing type casts present in the class hierarchy and makes the code readable, usable, and maintainable.
- Gives liberty to have more specific return types when overriding methods.
- Help in preventing run-time ClassCastExceptions on returns.

## Q 27. What's the difference between the methods sleep() and wait()?

**Ans :**

| Sleep() | Wait() |
|---|---|
| The sleep() method belongs to the thread class. | Wait() method belongs to the object class. |
| Sleep does not release the lock that | wait() release the lock which allows |

| Sleep() | Wait() |
|---|---|
| the current thread holds. | other threads to acquire it. |
| This method is a static method. | This method is not a static method. |
| Sleep() does not throw an InterruptedException. | InterruptedException is shown if the thread is interrupted while waiting. |
| Mainly used to delay a thread for some specific time duration. | Mainly used to pause a thread until notified by another thread. |
| Sleep() Has Two Overloaded Methods:<br>• sleep(long millis)millis: milliseconds<br>• sleep(long millis, int nanos) nanos: Nanoseconds | Wait() Has Three Overloaded Methods:<br>• wait()<br>• wait(long timeout)<br>• wait(long timeout, int nanos) |

## Q 28. What are the differences between String and StringBuffer?

**Ans :**

| String | StringBuffer |
|---|---|
| Store of a sequence of characters. | Provides functionality to work with the strings. |
| It is immutable. | It is mutable (can be modified and other string |

| String | StringBuffer |
|---|---|
| | operations could be performed on them.) |
| No thread operations in a string. | It is thread-safe (two threads can't call the methods of StringBuffer simultaneously) |

## Q 29. What are the differences between StringBuffer and StringBuilder?

**Ans :**

| StringBuffer | StringBuilder |
|---|---|
| StringBuffer provides functionality to work with the strings. | StringBuilder is a class used to build a mutable string. |
| It is thread-safe (two threads can't call the methods of StringBuffer simultaneously) | It is not thread-safe (two threads can call the methods concurrently) |
| Comparatively slow as it is synchronized. | Being non-synchronized, implementation is faster |

## Q 31. On which memory arrays are created in Java?

**Ans :** Arrays in Java are created in heap memory. When an array is created with the help of a new keyword, memory is allocated in the heap to store the elements of the array.
In Java, the heap memory is managed by the Java Virtual Machine(JVM) and it is also shared between all threads of the Java Program.
The memory which is no longer in use by the program, JVM uses a garbage collector to reclaim the memory.
Arrays in Java are created dynamically which means the size of the array is determined during the runtime of the program.
The size of the array is specified during the declaration of the array and it cannot be changed once the array is created.

## Q 32. How to copy an array in Java?

**Ans :** In Java there are multiple ways to copy an Array based on the requirements.

- **clone() method in Java:** This method in Java is used to create a shallow copy of the given array which means that the new array will share the same memory as the original array.

```
int[] Arr = { 1, 2, 3, 5, 0};

int[] tempArr = Arr.clone();
```

- **arraycopy() method:** To create a deep copy of the array we can use this method which creates a new array with the same values as the original array.

```
int[] Arr = {1, 2, 7, 9, 8};

int[] tempArr = new int[Arr.length];

System.arraycopy(Arr, 0, tempArr, 0, Arr.length);
```

- **copyOf() method:** This method is used to create a new array with a specific length and copies the contents of the original array to the new array.

```
int[] Arr = {1, 2, 4, 8};
```

```
int[] tempArr = Arrays.copyOf(Arr, Arr.length);
```

- **copyOfRange() method:** This method is very similar to the copyOf() method in Java, but this method also allows us to specify the range of the elements to copy from the original array.

```
int[] Arr = {1, 2, 4, 8};

int[] temArr = Arrays.copyOfRange(Arr, 0, Arr.length);
```

## Q 33. Is it possible to make an array volatile?

**Ans :** In Java, it is not possible to make a volatile.

Volatile keywords in Java can only be applied to individual variables but not to arrays or collections.

The value of the Variable is always read from and written to the main memory when it is defined as volatile rather than being cached in a thread ' s local memory.

This makes it easier to make sure that all threads that access the variable can see changes made to it.

## Q 34. What are the advantages and disadvantages of an array?

**Ans : The advantages of Arrays are:**

- Direct and effective access to any element in the collection is made possible by arrays. An array ' s elements can be accessed using an O(1) operation, which means that the amount of time needed to do so is constant and independent of the array ' s size.
- Data can be stored effectively in memory using arrays. The size of an array is known at compile time since its elements are stored in contiguous memory regions.
- Due to the fact that the data is stored in contiguous memory areas, arrays provide quick data retrieval.
- Arrays are easy to implement and understand, making them an ideal choice for beginners learning computer programming.

### Disadvantages of Arrays are:

- Arrays are created with a predetermined size that is chosen at that moment. This means that if the array's size needs to be extended, a new array will need to be made, and the data will need to be copied from the old array to the new array, which can take a lot of time and memory.
- There may be unused memory space in an array's memory space if the array is not completely occupied. If you have poor recall, this can be a problem.
- Compared to other data structures like linked lists and trees, arrays might be rigid due to their fixed size and limited support for sophisticated data types.
- Because an array's elements must all be of the same data type, it does not support complex data types like objects and structures.

## Q 35. What is an object-oriented paradigm?

**Ans :** Paradigm literally means a pattern or a method. Programming paradigms are the methods to solve a program that is of four types namely, Imperative, logical, functional, and object-oriented. When objects are used as base entities upon which the methods are applied, encapsulation or inheritance functionalities are performed, it is known as an object-oriented paradigm.

## Q 36.What is the difference between an object-oriented programming language and an object-based programming language?

| Object-Oriented Programming Language | Object-Based Programming Language |
|---|---|
| Object-oriented programming | The scope of object-based |

| Object-Oriented Programming Language | Object-Based Programming Language |
|---|---|
| language covers larger concepts like inheritance, polymorphism, abstraction, etc. | programming is limited to the usage of objects and encapsulation. |
| It supports all the built-in objects | It doesn’ t support all the built-in objects |
| Examples: Java, C#, etc. | Examples: Java script, visual basics, etc. |

## Q 37. What are Classes in Java?

**Ans :** In Java, Classes are the collection of objects sharing similar characteristics and attributes.

Classes represent the blueprint or template from which objects are created.

Classes are not real-world entities but help us to create objects which are real-world entities.

## Q 38. What is the difference between static (class) method and instance method?

**Ans :**

| Static(Class) method | Instance method |
|---|---|
| Static method is associated with a class rather than an object. | The instance method is associated with an object rather than a class. |

| Static(Class) method | Instance method |
|---|---|
| Static methods can be called using the class name only without creating an instance of a class. | The instance method can be called on a specific instance of a class using the object reference. |
| Static methods do not have access to **this** keyword. | Instance methods have access to **this** keyword. |
| This method can access only static members of the class | This method can access both static and non-static methods of the class. |
| Static methods can be overridden. | Instance methods cannot be overridden. |

## Q 39. What is an object?

**Ans :** The object is a real-life entity that has certain properties and methods associated with it. The object is also defined as the instance of a class. An object can be declared using a new keyword.

## Q 40. What are the different ways to create objects in Java?

**Ans :** Methods to create objects in Java are mentioned below:

1. Using new keyword
2. Using new instance
3. Using clone() method
4. Using deserialization
5. Using the newInstance() method of the Constructor class

## Q 41. What are the advantages and disadvantages of object cloning?

**Ans :** There are many advantages and disadvantages of using object cloning as mentioned below:

### Advantages:

- In Java, the '=' assignment operator cannot be used for cloning as it simply creates a copy of reference variables. To overcome such discrepancy the clone() method of Object class can be used over the assignment operator.
- The clone() method is a protected method of class Object which means that only the Employee class can clone Employee objects. This means no class other than Employee can clone Employee objects since it does not know the Employee class' attributes.
- Code size decreases as repetition decreases.

### Disadvantages:

- As the Object.clone() method is protected, so need to provide our own clone() and indirectly call Object.clone() from it.
- If we don' t have any methods then we need to provide a Cloneable interface as we need to provide JVM information so that we can perform a clone() on our object.

## Q 42. What are the advantages of passing this into a method instead of the current class object itself?

**Ans :** There are a few advantages of passing this into a method instead of the current class object itself these are:

- this is the final variable because of which 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.

## Q 43. What do you understand by copy constructor in Java?

**Ans :** The copy constructor is the type of constructor in which we pass another object as a parameter because which properties of both objects seem the same, that is why it seems as if constructors create a copy of an object.

## Q 44. What are the differences between the constructors and methods?

**Ans :** Java constructors are used for initializing objects. During creation, constructors are called to set attributes for objects apart from this few basic differences between them are:

1) Constructors are only called when the object is created but other methods can be called multiple times during the life of an object.
2) Constructors do return anything whereas other methods can return anything.
3) Constructors are used to setting up the initial state but methods are used to perform specific actions.

## Q 44. Give some features of the Interface.

**Ans :**An Interface in Java programming language is defined as an abstract type used to specify the behavior of a class. An interface in Java is a blueprint of a behavior. A Java interface contains static constants and abstract methods.

Features of the Interface are mentioned below:

- The interface can help to achieve total abstraction.
- Allows us to use multiple inheritances in Java.
- Any class can implement multiple interfaces even when one class can extend only one class.
- It is also used to achieve loose coupling.

## Q 45. Difference between an Error and an Exception.

**Ans :**

| Errors | Exceptions |
|---|---|
| Recovering from Errors is not possible. | Recover from exceptions by either using a try-catch block or throwing exceptions back to the caller. |
| Errors are all unchecked types in Java. | It includes both checked as well as unchecked types that occur. |
| Errors are mostly caused by the environment in which the program is running. | The program is mostly responsible for causing exceptions. |
| Errors can occur at compile time as | All exceptions occur at runtime but |

| Errors | Exceptions |
|---|---|
| well as run time. Compile Time: Syntax Error, Run Time: Logical Error. | checked exceptions are known to the compiler while unchecked are not. |
| They are defined in java.lang.Error package. | They are defined in java.lang.Exception package |
| **Examples**: java.lang.StackOverflowError, java.lang.OutOfMemoryError | **Examples**: Checked Exceptions: SQLException, IOException Unchecked Exceptions: ArrayIndexOutOfBoundException, NullPointerException, ArithmeticException. |

## Q 46. Explain Runtime Exceptions.

**Ans :** Runtime Exceptions are exceptions that occur during the execution of a code, as opposed to compile-time exceptions that occur during compilation. Runtime exceptions are unchecked exceptions, as they aren't accounted for by the JVM.

### Examples of runtime exceptions in Java include:

- NullPointerException: This occurs when an application attempts to use a null object reference.
- ArrayIndexOutOfBoundsException: This occurs when an application attempts to access an array index that is out of bounds.
- ArithmeticException: This occurs when an application attempts to divide by zero.

- IllegalArgumentException: This occurs when a method is passed on an illegal or inappropriate argument.

Unlike checked exceptions, runtime exceptions do not require a declaration in the throws clause or capture in a try-catch block. However, handling runtime exceptions is advisable in order to provide meaningful error messages and prevent a system crash. Because runtime exceptions provide more specific information about the problem than checked exceptions, they enable developers to detect and correct programming errors more easily and quickly.

## Q 47. What is the difference between Checked Exception and Unchecked Exception?

### Ans :

**Checked Exception:**

Checked Exceptions are the exceptions that are checked during compile time of a program.

In a program, if some code within a method throws a checked exception, then the method must either handle the exception or must specify the exception using the throws keyword.

Checked exceptions are of two types:

1) Fully checked exceptions: all its child classes are also checked, like IOException, and InterruptedException.
2) Partially checked exceptions: some of its child classes are unchecked, like an Exception.

**Unchecked Exception:**

Unchecked are the exceptions that are not checked at compile time of a program.

Exceptions under Error and RuntimeException classes are unchecked exceptions, everything else under throwable is checked.

## Q 48. What will happen if you put System.exit(0) on the try or catch block? Will finally block execute?

**Ans :** System.exit(int) has the capability to throw SecurityException.

So, if in case of security, the exception is thrown then finally block will be executed otherwise JVM will be closed while calling System.

exit(0) because of which finally block will not be executed.

## Q 49. What do you understand by Object Cloning and how do you achieve it in Java?

**Ans :** It is the process of creating an exact copy of any object. In order to support this, a java class has to implement the Cloneable interface of java.lang package and override the clone() method provided by the Object class the syntax of which is:

Protected Object clone() throws CloneNotSupportedException{ return (Object)super.clone();}In case the Cloneable interface is not implemented and just the method is overridden, it results in CloneNotSupportedException in Java.

## Q 50. What are the advantages of multithreading?

**Ans :** There are multiple advantages of using multithreading which are as follows:

- Responsiveness: User Responsiveness increases because multithreading interactive application allows running code even when the section is blocked or executes a lengthy process.
- Resource Sharing: The process can perform message passing and shared memory because of multithreading.
- Economy: We are able to share memory because of which the processes are economical.

- Scalability: Multithreading on multiple CPU machines increases parallelism.
- Better Communication: Thread synchronization functions improves inter-process communication.
- Utilization of multiprocessor architecture
- Minimized system resource use

# ABOUT US

**At TopperWorld, we are on a mission to empower college students with the knowledge, tools, and resources they need to succeed in their academic journey and beyond.**

## Our Vision

- **Our vision is to create a world where every college student can easily access high-quality educational content, connect with peers, and achieve their academic goals.**
- **We believe that education should be accessible, affordable, and engaging, and that's exactly what we strive to offer through our platform.**

## Unleash Your Potential

- **In an ever-evolving world, the pursuit of knowledge is essential. TopperWorld serves as your virtual campus, where you can explore a**

diverse array of online resources tailored to your specific college curriculum.

- Whether you're studying science, arts, engineering, or any other discipline, we've got you covered.
- Our platform hosts a vast library of e-books, quizzes, and interactive study tools to ensure you have the best resources at your fingertips.

## ➢ The TopperWorld Community

- Education is not just about textbooks and lectures; it's also about forming connections and growing together.
- TopperWorld encourages you to engage with your fellow students, ask questions, and share your knowledge.
- We believe that collaborative learning is the key to academic success.

## ➢ Start Your Journey with TopperWorld

- Your journey to becoming a top-performing college student begins with TopperWorld.
- Join us today and experience a world of endless learning possibilities.
- Together, we'll help you reach your full academic potential and pave the way for a brighter future.
- Join us on this exciting journey, and let's make academic success a reality for every college student.

# "UNLOCK YOUR POTENTIAL"

With- TOPPERWORLD

Explore More


topperworld.in

DSA TUTORIAL

C TUTORIAL

C++ TUTORIAL

JAVA TUTORIAL

PYTHON TUTORIAL

Follow Us On

E-mail

topperworld.in@gmail.com