Java2blog :=========================================

Java Interview Question -------------------------

**1. What do you mean by Platform independence of java?**

You can write and compile program in one Operating system and run in other operating system.

**example:**  
You can compile program in Windows and can run it in Unix.

**2. What is difference between JVM, JRE and JDK ?**

**JVM :** [**JVM**](https://java2blog.com/java-virtual-machine-architecture/) stands for Java Virtual Machine. It is virtual machine which actually runs the byte code.

**JRE :**JRE stands for Java Runtime Environment. It provides runtime environment for java code. It has JVM , libraries such as rt.jar and other files.

**JDK :**JDK stands for Java development kit. It is superset of JRE, it has JRE + compilation and debugging tools(javac and java).

**3. What are memory areas allocated in JVM?**

Memory areas allocated in [**JVM**](https://java2blog.com/java-virtual-machine-architecture/) are:

* Heap area
* Method area
* JVM language stacks
* Program counter (PC) register
* Native method stacks

**4. What are some core concepts of OOPS in Java ?**

Core concepts of [**OOPs**](https://java2blog.com/oops-interview-questions-answers-java/) are :

* Encapsulation
* Polymorphism
* Abstraction
* Inheritance

**5. What is Abstraction?**

Abstraction is achieved using [**interface**](https://www.java2blog.com/2017/04/interface-in-java-with-example.html) and [**abstract class**](https://www.java2blog.com/2017/04/abstract-class-java.html) in Java.

You can read about [**abstraction**](https://www.java2blog.com/2017/04/abstraction-java-example.html) for more details..

**6. What is encapsulation?**

You can refer about [**encapsulation**](https://www.java2blog.com/2017/05/encapsulation-java-example.html) for more details..

**7. What is Polymorphism in java?**

You can refer about [**Polymorphism**](https://www.java2blog.com/2017/05/polymorphism-java-example.html) for more details..

**8. What is inheritance in java?**

[**Inheritance**](https://java2blog.com/inheritance-java/) allows to inherit properties and methods of parent class, so you can reuse all methods and properties.

**9. What is constructor in java?**

Constructor  can be considered a special code which is used to initiaze objects.  
It has two main points

* Class and Constuctor name should match
* Constructor should not have any return type else it will be same as method.

You can read more about [**Constructor in Java**](https://www.java2blog.com/2017/05/constructor-java.html).

**10. Can we declare constructor as final?**

No, Constructor can not be declared as final. If you do so, you will get compile time error.

**11. What is immutable object in java?**

[**Immutable**](https://java2blog.com/how-to-create-immutable-class-in-java/) object is object whose state can not be changed once created. You can take String object as example for immutable object.

**12. Why String is declared final or immutable in java?**

There are various reasons to make String [**immutable**](https://java2blog.com/immutable-class-interview-questions/).

* String pool
* Thread Safe
* Security
* Class Loading
* Cache hash value

You can refer why [**String is immutable in java**](https://www.java2blog.com/2016/05/why-string-is-immutable-in-java.html) for more details.

**13. What are access modifier available in java?**

It Specifies accessibility of variables, methods , constructor of class.

There are four [**access modifier in java**](https://java2blog.com/access-modifiers-java/)

**Private** : Accessible only to the class.

**Default :** Accessible in the package.

**Protected :**Accessible in the packages and its subclasses.

**Public :** Accessible everywhere

**14. What is difference between Abstract class and interface?**

You can refer [**difference between Abstract class and interface**](https://www.java2blog.com/2014/06/difference-between-abstract-class-and.html) for more details.

**15. Can one interface implement another interface in java?**

No, One [**interface**](https://java2blog.com/interface-in-java-with-example/) can not implement another interface. It can extend it using extends keyword.

**16. What is marker interface?**

Marker interfaces  are interfaces which have no method but it is used to indicate JVM to behave specially when any class implement these interfaces.

**For example :**If you implement cloneable interface and then call .clone method of object, it will clone your object. If you do not implement cloneable interface, it will throw cloneNotSupported exception.

**17. What is method overloading and method overriding in java?**

**Method overloading :** [**Method overloading**](https://java2blog.com/method-overloading-in-java/) is concept that allows a class to have same method name but diferent method arguments. Method overloading is also known as compile time polymorphism.  
 **Method overriding :** If child class contain same method as parent class with same method signature. This is called [**method overriding**](https://java2blog.com/method-overriding-in-java/). Method overriding is also known as dynamic polymorphism.

**18. Can you override static methods in Java?**

No, you can not override [**static**](https://java2blog.com/static-keyword-in-java/) methods in Java. You can create same method in child class but it won’t be dynamic [**polymorphism**](https://java2blog.com/polymorphism-java-example/). It will be method hiding. Static methods belong at class level not at object level hence you can not override static method.

**19. Can you override private methods in Java?**

No, you can not override private methods in Java. Private methods are not visible to subclass, hence you can not override private method but you can hide it.

**20. Difference between path and classpath in java?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Path** | **classpath** |
| Locate | It allows operating system to locate executable such as javac, java | It allows classloader to locate all .class file used by program |
| Overriding | You can not override path variable with java setting | You can override classpath by using -cp with java,javac or class-path in manifest file. |
| Inclusion | You need to include bin folder of jdk (**For example jdk1.7.1/bin**) | You need to include all the classes which is required by program |
| Used by | Operating system | java classloaders |

You can refer [**difference between Path and ClassPath in java**](https://www.java2blog.com/2016/05/difference-between-path-and-classpath-in-java.html) for more details.

**21. What is difference between StringBuffer and StringBuilder in java?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **StringBuffer** | **StringBuilder** |
| Thread-safe | StringBuffer is thread safe. Two threads can not call methods of StringBuffer simultaneously. | StringBuilder is not thread safe, so two threads can call methods of StringBuilder simultaneously. |
| Performance | It is less performance efficient as it is thread-safe | It is more performance efficient as it is not thread-safe. |

**22. What are methods you should override when you put an object as key in HashMap?**

You need to implement hashcode() and equals() method if you put key as object in **[HashMap](https://java2blog.com/hashmap-in-java-with-examples/" \t "_blank)**. You can go through **[hashcode and equals method in java](https://www.java2blog.com/2014/02/hashcode-and-equals-method-in-java.html" \t "_blank)** for more details.

**23. Can you explain internal working of HashMap in java?**

* There is an Entry[]  array called table which has size 16.
* This table stores Entry class’s object. **[HashMap](https://java2blog.com/hashmap-in-java-with-examples/" \t "_blank)** class has a inner class called Entry.This Entry have key value as instance variable.

Let’s see the structure of entry class Entry Structure.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | static class Entry implements Map.Entry  {          final K key;          V value;          Entry next;          final int hash;          ...//More code goes here  } |

Whenever we try to put any key value pair in **[Hashmap](https://java2blog.com/hashmap-in-java-with-examples/" \t "_blank)**, Entry class object is instantiated for key value and that object will be stored in above-mentioned Entry[](table). Now you must be wondering, where will above created Entry object gets stored(exact position in table). The answer is, hash code is calculated for a key by calling Hashcode() method. This hashcode is used to calculate the index for above Entry[] table.  
You can read [**How HashMap works internally in java**](https://www.java2blog.com/2014/02/how-hashmap-works-in-java.html) for more details.

**24. Why java uses another hash function internally to calculate hash value apart from hashcode method which you have implemented?**

It is due to avoid large number of collisions due to bad **[hashcode](https://java2blog.com/hashcode-and-equals-method-in-java/" \t "_blank)** method written by developers.

You can refer [**hash method of HashMap**](https://www.java2blog.com/2014/02/hash-and-indexfor-method-in-hashmap.html) for more details.

**25. What if you don’t override hashcode method while putting custom objects as key in HashMap?**

As we did not implement hashcode method, each object will have different hashcode(memory address) by default, so even if we have implemented equals method correctly, it won’t work as expected.

**26. Can you explain internal working of HashSet in java?**

HashSet internally uses **[HashMap](https://java2blog.com/hashmap-in-java-with-examples/" \t "_blank)** to store elements in HashSet. It uses PRESENT as dummy object as value in that HashMap. HashSet uses HashMap to check duplicates in the HashSet.

You can refer [**How HashSet works internally in java**](https://www.java2blog.com/2014/07/how-hashset-works-in-java.html) for more details

**27. What are differences between HashMap and HashSet in java?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **HashMap** | **HashSet** |
| Interface | This is core difference among them.HashMap implements Map interface | HashSet implement Set interface |
| Method for storing data | It stores data in a form of key->value pair.So it uses put(key,value) method for storing data | It uses add(value) method for storing data |
| Duplicates | HashMap allows duplicate value but not duplicate keys | HashSet does not allow duplicate values. |
| Performance | It is faster than hashset as values are stored with unique keys | It is slower than HashMap |
| HashCode Calculation | In hash map hashcode value is calculated using key object | In this,hashcode is calculated on the basis of value object. Hashcode can be same for two value object so we have to implement equals() method.If equals() method return false then two objects are different. |

**28. Can you explain internal working of ConcurrentHashMap in java?**

[**ConcurrentHashMap**](https://java2blog.com/concurrenthashmap-in-java/) uses concept of Segments to store elements. Each Segment logically contains a **[HashMap](https://java2blog.com/hashmap-in-java-with-examples/" \t "_blank)**. ConcurrentHashMap does not lock whole object , it just lock part of it i.e. Segment.  
Structure of Segment:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | /\*\*       \* Segments are specialized versions of hash tables.  This       \* subclasses from ReentrantLock opportunistically, just to       \* simplify some locking and avoid separate construction.       \*/  static final class Segment extends ReentrantLock implements Serializable {  /\*\*        \* The per-segment table.  \*/          transient volatile HashEntry[] table;  // other methods and variables  } |

It stores a key value pair in a class called HashEntry which is similar to Entry class in **[HashMap](https://java2blog.com/hashmap-in-java-with-examples/" \t "_blank)**.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | static final class HashEntry {      final K key;    final int hash;  volatile V value;  final HashEntry next;  } |

You can refer [**internal working of ConcurrentHashMap in jav**](https://www.java2blog.com/2014/12/concurrenthashmap-in-java.html)a for more details

**29. Do we have lock while getting value from ConcurrentHashMap?**

There is no lock while getting values from **[ConcurrentHashMap](https://java2blog.com/concurrenthashmap-in-java/" \t "_blank)**.Segments are only for write operation.In case of read operation, it allows full concurrency and provides most recently updated value using volatile variables.

**30. How do you sort Collection of custom objects in java?**

We need to implement the comparable interface to custom object class(Let’s say Country) and then implement compareTo(Object o) method which will be used for sorting. It will provides default way of sorting custom objects.  
If we want to sort custom object (Lets say country) on different attributes such as name, population etc.We can implement Comparator interface and can be used for sorting.  
For more details, you can go through following links:

[**Comparable in java**](https://www.java2blog.com/2014/06/comparable-in-java.html)  
[**Comparator in java**](https://www.java2blog.com/2014/06/comparator-in-java.html)

**31. What are differences between ArrayList and LinkedList in java?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **ArrayList** | **LinkedList** |
| Internal data structure | It uses dynamic array to store elements internally | It uses doubly Linked List to store elements internally |
| Manipulation | If  We need to insert or delete element in **[ArrayList](https://java2blog.com/arraylist-in-java-with-example/" \t "_blank)**, it may take O(n), as it internally uses array and we may have to shift elements in case of insertion or deletion | If  We need to insert or delete element in LinkedList, it will take O(1), as it internally uses doubly LinkedList |
| Search | Search is faster in ArrayList as uses array internally which is index based. So here time complexity is O(1) | Search is slower in LinkedList as uses doubly Linked List internally So here time complexity is O(n) |
| Interfaces | ArrayList implements List interface only, So it can be used as List only | LinkedList implements List,Deque interfaces, so it can be used as List,Stack or Queue |

You can refer [**difference between ArrayList and LinkedList in java**](https://www.java2blog.com/2015/06/difference-between-arraylist-and.html) for more details.

**32. What is Enum in java?**

[**Java Enum**](https://java2blog.com/java-enum/) is special data type which represents list of constants values. It is a special type of java class. It can contain constant, methods and constructors etc.  
You can refer **[Enum in java](https://www.java2blog.com/2015/09/java-enum.html" \t "_blank)** for more details.

**33. How do you create custom exception in java?**

You just need to extend Exception class to create custom exception. If you want to create Unchecked exception, then you need extend Runtime Exception.

You can refer to [**create custom exception in Java**](https://www.java2blog.com/2016/07/how-to-create-custom-exception-in-java.html).

**34.What is difference between Checked Exception and Unchecked Exception?**

**Checked Exception:**Checked exceptions are those exceptions which are checked at compile. If you do not handle them , you will get compilation error.

**For example:**IOException

**Unchecked Exception :**Unchecked exceptions are those exceptions which are not checked at compile time. Java won’t complain if you do not handle the exception.

**For example:** NullPointerException, ArrayIndexOutOfBoundsException

You can refer [**difference between checked exception and unchecked exception**](https://www.java2blog.com/2016/07/difference-between-checked-and-unchecked-exception-in-java.html) for more details.

**35. Can we have try without catch block in java ?**

Yes, we can have try without catch block by using finally block. You can use try with finally. As you know finally block always executes even if you have exception or return statement in try block except in case of System.exit().  
You can refer [**Try with finally block**](https://www.java2blog.com/2016/06/can-we-have-try-without-catch-block-in-java.html) for more details.

**36. What are ways to create a thread in java ?**

There are two ways to create a thread in java

* By extending thread class
* By implementing the [**Runnable**](https://java2blog.com/java-runnable-example/) interface.

**37. What are differences between Sleep and wait in java?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Wait** | **sleep** |
| Synchonized | wait should be called from synchronized context i.e. from block or method, If you do not call it using synchronized context, it will throw IllegalMonitorStateException | It need not be called from synchronized block or methods |
| Calls on | wait method operates on Object and defined in Object class | Sleep method operates on current thread and is in java.lang.Thread |
| Release of lock | wait release lock of object on which it is called and also other locks if it holds any | Sleep method does not release lock at all |
| Wake up condition | until call notify() or notifyAll() from Object class | Until time expires or calls interrupt() |
| static | wait is non static method | sleep is static method |

You can refer [**difference between sleep and wait in java**](https://www.java2blog.com/2014/07/difference-between-sleep-and-wait-in.html) for more details.

**38. Define states of thread in java?**

There are 5 states of thread in java

**New**: When you create a thread object and it is not alive yet.

[**Runnable**](https://java2blog.com/java-runnable-example/)**:**When you call start method of thread, it goes into Runnable state. Whether it will execute immediately or execute after some times , depends on thread scheduler.

**Running :**When thread is being executed, it goes to running state.

**Blocked :**When thread waits for some resources or some other thread to complete (due to thread’s join), it goes to blocked state.

**Dead:**When thread’s run method returns, thread goes to dead state.

**39. Can we call run method directly to start a thread?**

No, you can not directly call run method to start a thread. You need to call start method to create a new thread. If you call run method directly , it won’t create a new thread and it will be in same stack as main.  
You can refer [**can we call run method directly to start a thread**](https://www.java2blog.com/2016/05/can-we-call-run-method-directly-to-start-thread.html) for more details

**40. Can we start a thread twice in java?**

No, Once you have started a thread, it can not be started again. If you try to start thread again , it will throw IllegalThreadStateException.  
You can refer [**can we start thread twice**](https://www.java2blog.com/2016/05/can-we-start-thread-twice-in-java.html) for more details

**41. What is CountDownLatch in java?**

As per java docs, CountDownLatch  is synchronisation aid that allow one or more threads to wait until set of operations being performed in other threads completes. So in other words, CountDownLatch waits for other threads to complete set of operations.  
[**CountDownLatch**](https://java2blog.com/countdownlatch-in-java/) is initialized with count. Any thread generally main threads calls latch.awaits() method, so it will wait for either count becomes zero or it’s interrupted by another thread and all other thread need to call latch.countDown() once they complete some operation.

So count is reduced by 1 whenever latch.countDown() method get called, so if count is n that means count can be used as n threads have to complete some action or some action have to be completed n times.  
You can refer **[CountDownLatch in java with example](https://www.java2blog.com/2015/08/countdownlatch-in-java.html" \t "_blank)** for more details.

**42. What is difference between CountDownLatch and CyclicBarrier?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **CountDownLatch** | **CyclicBarrier** |
| Reuse | It can not be reused once count reaches 0 | It can be reinitialized once parties reaches to 0, so it can reused |
| Method | It calls countDown() method to reduce the counter | It calls await() method to reduce the counter. |
| Common Event | It can not trigger common event when count reaches 0 | It can trigger common event (Runnable) once reaches to a barrier point. **Constructor** :CyclicBarrier(int parties, Runnable barrierAction) |
| Constructor | CountDownLatch(int count) | CyclicBarrier(int parties) |

**43. Why wait, notify and nofiyAll method belong to object class ?**

In java, we put locks on shared objects not on thread, so these methods are present in Object class. As every object have mutex(lock), it make sense to put [**wait, notify and notifyAll**](https://java2blog.com/wait-notify-and-notifyall-method-in/) methods in object class.

**44. Can you call wait, notify and notifyAll from non synchronized context?**

No, you can not call [**wait, notify and notifyAll**](https://java2blog.com/wait-notify-and-notifyall-method-in/) from non synchronized context. If you do so, it will throw IllegalMonitorStateException.

**45. What is the difference between creating String as new() and literal?**

If you create a String using new operator, it is not interned. It will  create new object in heap memory even if String object already exists with same content.

|  |  |
| --- | --- |
| 1  2  3  4  5 | String str1=new String("hello");          String str2=new String("hello");          System.out.println(str1==str2); |

It will return false as str1 and str2 will point to different object

If you create a String using assignment operator, it goes to the String constant pool and it is interned. If you create another String with same content, both will reference to same object in String constant pool.

|  |  |
| --- | --- |
| 1  2  3  4  5 | String str1="helloworld";  String str2="helloworld";  System.out.println(str1==str2); |

It will return true as str1 and str2 will point to the same object in String constant pool.

**46. What is Covariant return type in java?**

Covariant return type means if subclass overrides any method, return type of this overriding method can be subclass of return type of base class method.  
**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | package org.arpit.java2blog;    public class BaseClass {    public A m1() {    System.out.println("In BaseClass method");    return new A();  }    public static void main(String args[])  {    BaseClass b=new SubClass();    b.m1();  }  }    class SubClass extends BaseClass {  public B m1() {    System.out.println("In SubClass method");    return new B();    }  }    class A {  }    class B extends A {    } |

Above example is perfect example of covariant return type.

**47. What is garbage Collection?**

[**Garbage Collection**](https://java2blog.com/garbage-collection-java/) is a process of looking at heap memory and deleting unused object present in heap memory. Garbage Collection frees unused memory. Garbage Collection is done by JVM.

**48. What is System.gc()?**

This method is used to invoke garbage collection for clean up unreachable object but it is not guaranteed that when you invoke System.gc() , [**garbage collection**](https://java2blog.com/garbage-collection-java/) will definitely trigger.

**49. What is use of finalize() method in object class?**

Finalize method  get called when object is being collected by Garbage Collector. This method can be used to write clean code before object is collected by Garbage Collector.

**50.What is difference between final, finally and finalize in Java?**

**final :** Final is a keyword which is used with class to avoid being extended, with instance variable so they can not reassigned, with methods so that they can not be overridden.  
**finally :** Finally is a keyword used with try, catch and finally blocks. Finally block executes even if there is an exception. It is generally used to do some clean up work.  
**finalize :** Finalize is a method is used to invoke garbage collection for clean up unreachable object but it is not guaranteed that when you invoke System.gc(), garbage collection will definitely trigger.