Prime alog:

    private boolean isPrime(int number){

        if (number < 2 || (number % 2 == 0 && number != 2)){

            return false;

        }

        else if (number == 2){

            return true;

        }

        else {

            for (int i=2; i <= (number/2); ++i){

                if (number % i == 0){

                    return false;

                }

            }

            return true;

        }

    }

Java nested class:

https://jenkov.com/tutorials/java/nested-classes.html

In Java nested classes are classes that are defined inside another class.

The purpose of a nested class is to clearly group the nested class with its surrounding class, signaling that these two classes are to be used together. Or perhaps that the nested class is only to be used from inside its enclosing (owning) class.

Java developers often refer to *nested classes* as *inner classes*, but inner classes (non-static nested classes) are only one out of several different types of nested classes in Java.

In Java nested classes are considered members of their enclosing class. Thus, a nested class can be declared public, package (no access modifier), protected and private (see [**access modifiers**](https://jenkov.com/tutorials/java/access-modifiers.html) for more info). Therefore nested classes in Java can also be inherited by subclasses as explained in my tutorial about [**Java inheritance**](https://jenkov.com/tutorials/java/inheritance.html).

You can create several different types of nested classes in Java. The different Java nested class types are:

* Static nested classes
* Non-static nested classes
* Local classes
* Anonymous classes

All these types of nested classes will be covered in the following sections.

**Static Nested Classes**

Static nested classes are declared in Java like this:

public class Outer {

public static class Nested {

}

}

In order to create an instance of the Nested class you must reference it by prefixing it with the Outer class name, like this:

Outer.Nested instance = new Outer.Nested();

In Java a static nested class is essentially a normal class that has just been nested inside another class. Being static, a static nested class can only access instance variables of the enclosing class via a reference to an instance of the enclosing class.

**Non-static Nested Classes (Inner Classes)**

Non-static nested classes in Java are also called *inner classes*. Inner classes are associated with an instance of the enclosing class. Thus, you must first create an instance of the enclosing class to create an instance of an inner class. Here is an example inner class definition:

public class Outer {

public class Inner {

}

}

Here is how you create an instance of the Inner class:

Outer outer = new Outer();

Outer.Inner inner = outer.new Inner();

Notice how you put new after the reference to the outer class in order to create an instance of the inner class.

Non-static nested classes (inner classes) have access to the fields of the enclosing class, even if they are declared private. Here is an example of that:

public class Outer {

private String text = "I am private!";

public class Inner {

public void printText() {

System.out.println(text);

}

}

}

Notice how the printText() method of the Inner class references the private text field of the Outer class. This is perfectly possible. Here is how you would call the printText() method:

Outer outer = new Outer();

Outer.Inner inner = outer.new Inner();

**inner.printText();**

**Inner Class Shadowing**

If a Java inner class declares fields or methods with the same names as field or methods in its enclosing class, the inner fields or methods are said to *shadow* over the outer fields or methods. Here is an example:

public class Outer {

private String text = "I am Outer private!";

public class Inner {

private String text = "I am Inner private";

public void printText() {

System.out.println(text);

}

}

}

In the above example both the Outer and Inner class contains a field named text. When the Inner class refers to text it refers to its own field. When Outer refers to text it also refers to its own field.

Java makes it possible though, for the Inner class to refer to the text field of the Outer class. To do so it has to prefix the text field reference with Outer.this. (the outer class name + .this. + field name) like this:

public class Outer {

private String text = "I am Outer private!";

public class Inner {

private String text = "I am Inner private";

public void printText() {

System.out.println(text);

System.out.println(**Outer.this.text**);

}

}

}

Now the Inner.printText() method will print both the Inner.text and Outer.text fields.

**Local Classes**

Local classes in Java are like inner classes (non-static nested classes) that are defined inside a method or scope block ({ ... }) inside a method. Here is an example:

class Outer {

public void printText() {

class Local {

}

Local local = new Local();

}

}

Local classes can only be accessed from inside the method or scope block in which they are defined.

Local classes can access members (fields and methods) of its enclosing class just like regular inner classes.

Local classes can also access local variables inside the same method or scope block, provided these variables are declared final.

From Java 8 local classes can also access local variables and parameters of the method the local class is declared in. The parameter will have to be declared final or be *effectually final*. Effectually final means that the variable is never changed after it is initialized. Method parameters are often effectually final.

Local classes can also be declared inside static methods. In that case the local class only has access to the static parts of the enclosing class. Local classes cannot contain all kinds of static declarations (constants are allowed - variables declared static final), because local classes are non-static in nature - even if declared inside a static method.

The same shadowing rules apply for local classes as for inner classes.

**Anonymous Classes**

Anonymous classes in Java are nested classes without a class name. They are typically declared as either subclasses of an existing class, or as implementations of some [**interface**](https://jenkov.com/tutorials/java/interfaces.html).

Anonymous classes are defined when they are instantiated. Here is an example that declares an anonymous subclass of a superclass called SuperClass:

public class SuperClass {

public void doIt() {

System.out.println("SuperClass doIt()");

}

}

SuperClass instance = new SuperClass() {

public void doIt() {

System.out.println("Anonymous class doIt()");

}

};

instance.doIt();

Running this Java code would result in Anonymous class doIt() being printed to System.out. The anonymous class subclasses (extends) SuperClass and overrides the doIt() method.

A Java anonymous class can also implement an interface instead of extending a class. Here an example:

public interface MyInterface {

public void doIt();

}

MyInterface instance = new MyInterface() {

public void doIt() {

System.out.println("Anonymous class doIt()");

}

};

instance.doIt();

As you can see, an anonymous class implementing an interface is pretty similar to an anonymous class extending another class.

An anonymous class can access members of the enclosing class. It can also access local variables which are declared final or effectively final (since Java 8).

You can declare fields and methods inside an anonymous class, but you cannot declare a constructor. You can declare a static initializer for the anonymous class instead, though. Here is an example:

final Strint textToPrint = "Text...";

MyInterface instance = new MyInterface() {

private String text;

//static initializer

{ this.text = textToPrint; }

public void doIt() {

System.out.println(this.text);

}

};

instance.doIt();

The same shadowing rules apply to anonymous classes as to inner classes.

**Nested Class Benefits**

The benefits of Java nested classes are that you can group classes together that belong together. You could do so already by putting them in the same package, but putting one class inside another makes an even stronger grouping.

A nested class is typically only used *by* or *with* its enclosing class. Sometimes a nested class is only visible to the enclosing class, is only used internally, and is thus never visible outside the enclosing class. Other times the nested class is visible outside its enclosing class, but can only be used in conjunction with the enclosing class.

An example would be a Cache class. Inside the Cache class you might declare a CacheEntry class which can contain information about a specific cache entry (cached value, time inserted, number of times accessed etc.). Users of the Cache class may never see the CacheEntry class, if they have no need to obtain information about the CacheEntry itself, but only the cached value. However, the Cache class may choose to make the CacheEntry class visible to the outside world, so they can access more than just the cached value (for instance information about when the value was last refreshed etc.).

Here are two Cache implementation skeletons illustrating the points:

public class Cache {

private Map<String, CacheEntry> cacheMap = new HashMap<String, CacheEntry>();

private class CacheEntry {

public long timeInserted = 0;

public object value = null;

}

public void store(String key, Object value){

CacheEntry entry = new CacheEntry();

entry.value = value;

entry.timeInserted = System.currentTimeMillis();

this.cacheMap.put(key, entry);

}

public Object get(String key) {

CacheEntry entry = this.cacheMap.get(key);

if(entry == null) return null;

return entry.value;

}

}

public class Cache {

private Map<String, CacheEntry> cacheMap = new HashMap<String, CacheEntry>();

**public** class CacheEntry {

public long timeInserted = 0;

public object value = null;

}

public void store(String key, Object value){

CacheEntry entry = new CacheEntry();

entry.value = value;

entry.timeInserted = System.currentTimeMillis();

this.cacheMap.put(key, entry);

}

public Object get(String key) {

CacheEntry entry = this.cacheMap.get(key);

if(entry == null) return null;

return entry.value;

}

**public CacheEntry getCacheEntry(String key) {**

**return this.cacheMap.get(key);**

**}**

}

Print all methods in the class:

public static void main(String[] args){

Class student = Student.class;

Method[] methods = student.getDeclaredMethods();

ArrayList<String> methodList = new ArrayList<>();

for(Method m : methods){

methodList.add(m.getName());

}

Collections.sort(methodList);

for(String name: methodList){

System.out.println(name);

}

}

Problem statement :

**Sample Input**

5

amy 100

david 100

heraldo 50

aakansha 75

aleksa 150

**Sample Output**

aleksa 150

amy 100

david 100

aakansha 75

heraldo 50

class Checker implements Comparator<Player>{

@Override

public int compare(Player a, Player b){

if (a.score == b.score){

return a.name.compareTo(b.name);//alphabetically

} else {

return b.score - a.score;//decreasing

}

}

}

Generics :

import java.io.IOException;

import java.lang.reflect.Method;

class Printer

{

   //Write your code here

     <T> void printArray(T[] elements){

       for (T element : elements){

           System.out.println(element);

       }

   }

}

public class Solution {

    public static void main( String args[] ) {

        Printer myPrinter = new Printer();

        Integer[] intArray = { 1, 2, 3 };

        String[] stringArray = {"Hello", "World"};

        myPrinter.printArray(intArray);

        myPrinter.printArray(stringArray);

        int count = 0;

        for (Method method : Printer.class.getDeclaredMethods()) {

            String name = method.getName();

            if(name.equals("printArray"))

                count++;

        }

        if(count > 1)System.out.println("Method overloading is not allowed!");

    }

}

our Output (stdout)

* **1**
* **2**
* **3**
* **Hello**
* **World**

Expected Output

* **1**
* **2**
* **3**
* **Hello**
* **World**

Hashset:

import java.io.\*;

import java.util.\*;

import java.text.\*;

import java.math.\*;

import java.util.regex.\*;

public class Solution {

 public static void main(String[] args) {

        Scanner s = new Scanner(System.in);

        int t = s.nextInt();

        String [] pair\_left = new String[t];

        String [] pair\_right = new String[t];

        for (int i = 0; i < t; i++) {

            pair\_left[i] = s.next();

            pair\_right[i] = s.next();

        }

//Write your code here

   HashSet<String> pairs = new HashSet<String>(t);

    for(int i = 0; i < t; i++)

    {

        pairs.add("(" + pair\_left[i] + ", " + pair\_right[i] + ")" );

        System.out.println(pairs.size());

    }

    }

**Sample Input**

5

john tom

john mary

john tom

mary anna

mary anna

**Sample Output**

1

2

2

3

3

Compartor:

import java.util.\*;

// Write your Checker class here

class Checker implements Comparator<Player>{

    @Override

    public int compare(Player a, Player b){

        if (a.score == b.score){

            return a.name.compareTo(b.name);//alphabetically

        } else {

            return b.score - a.score;//decreasing

        }

    }

}

class Player{

    String name;

    int score;

    Player(String name, int score){

        this.name = name;

        this.score = score;

    }

}

class Solution {

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();

        Player[] player = new Player[n];

        Checker checker = new Checker();

        for(int i = 0; i < n; i++){

            player[i] = new Player(scan.next(), scan.nextInt());

        }

        scan.close();

        Arrays.sort(player, checker);

        for(int i = 0; i < player.length; i++){

            System.out.printf("%s %s\n", player[i].name, player[i].score);

        }

    }

}

**Sample Input**

5

amy 100

david 100

heraldo 50

aakansha 75

aleksa 150

**Sample Output**

aleksa 150

amy 100

david 100

aakansha 75

heraldo 50

* [1. What is ORM?](https://java2blog.com/hibernate-interview-questions-and-answers/#1_What_is_ORM)
* [2. What is Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#2_What_is_Hibernate)
* [3. What are advantages of Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#3_What_are_advantages_of_Hibernate)
* [4. Explain architecture of Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#4_Explain_architecture_of_Hibernate)
* [5. What are some core interfaces of hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#5_What_are_some_core_interfaces_of_hibernate)
* [6. Explain brief about Session interface used in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#6_Explain_brief_about_Session_interface_used_in_hibernate)
* [7. Are session objects thread safe?](https://java2blog.com/hibernate-interview-questions-and-answers/#7_Are_session_objects_thread_safe)
* [8. Explain brief about SessionFactory object used in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#8_Explain_brief_about_SessionFactory_object_used_in_hibernate)
* [9. What is Configuration class in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#9_What_is_Configuration_class_in_hibernate)
* [10. Explain brief about Criteria API in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#10_Explain_brief_about_Criteria_API_in_Hibernate)
* [11. Explain brief about Query interface in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#11_Explain_brief_about_Query_interface_in_Hibernate)
* [12. What are differences between openSession and getCurrentSession in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#12_What_are_differences_between_openSession_and_getCurrentSession_in_hibernate)
* [13. What are differences between get and load methods in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#13_What_are_differences_between_get_and_load_methods_in_hibernate)
* [14. Can you declare Entity class as final in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#14_Can_you_declare_Entity_class_as_final_in_hibernate)
* [15. Differentiate between first level cache and second level cache?](https://java2blog.com/hibernate-interview-questions-and-answers/#15_Differentiate_between_first_level_cache_and_second_level_cache)
* [16. What are states of object in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#16_What_are_states_of_object_in_hibernate)
* [17. Does entity class in hibernate require no arg constructor?](https://java2blog.com/hibernate-interview-questions-and-answers/#17_Does_entity_class_in_hibernate_require_no_arg_constructor)
* [18. What is HQL?](https://java2blog.com/hibernate-interview-questions-and-answers/#18_What_is_HQL)
* [19. What are differences between save and saveOrUpdate method of session object?](https://java2blog.com/hibernate-interview-questions-and-answers/#19_What_are_differences_between_save_and_saveOrUpdate_method_of_session_object)
* [20. What are two types of Collections in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#20_What_are_two_types_of_Collections_in_hibernate)
* [21. What are differences between sorted and ordered collection in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#21_What_are_differences_between_sorted_and_ordered_collection_in_hibernate)
* [22. What is lazy loading in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#22_What_is_lazy_loading_in_hibernate)
* [23. What is LazyInitializationException in Hibernate? Why do you get it?](https://java2blog.com/hibernate-interview-questions-and-answers/#23_What_is_LazyInitializationException_in_Hibernate_Why_do_you_get_it)
* [24. What are different collection types available in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#24_What_are_different_collection_types_available_in_hibernate)
* [25. How can you log sql queries executed by Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#25_How_can_you_log_sql_queries_executed_by_Hibernate)
* [26. Can you execute native sql in hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#26_Can_you_execute_native_sql_in_hibernate)
* [27. What are inheritance mapping strategies are available in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#27_What_are_inheritance_mapping_strategies_are_available_in_Hibernate)
* [28. How to make an class immutable in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#28_How_to_make_an_class_immutable_in_Hibernate)
* [29. What is autamatic dirty checking in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#29_What_is_autamatic_dirty_checking_in_Hibernate)
* [30. What do you understand by Hibernate tuning?](https://java2blog.com/hibernate-interview-questions-and-answers/#30_What_do_you_understand_by_Hibernate_tuning)
* [31. What types of joins can you use in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#31_What_types_of_joins_can_you_use_in_Hibernate)
* [32. What is dialect in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#32_What_is_dialect_in_Hibernate)
* [33. Can you share some of the databases supported by Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#33_Can_you_share_some_of_the_databases_supported_by_Hibernate)
* [34. What is Named queries in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#34_What_is_Named_queries_in_Hibernate)
* [35. What is Query cache in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#35_What_is_Query_cache_in_Hibernate)
* [36. What are benefits of Criteria API in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#36_What_are_benefits_of_Criteria_API_in_Hibernate)
* [37. What is cascading in Hiberante and can you list types of cascading?](https://java2blog.com/hibernate-interview-questions-and-answers/#37_What_is_cascading_in_Hiberante_and_can_you_list_types_of_cascading)
* [38. How to integrate log4j with Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#38_How_to_integrate_log4j_with_Hibernate)
* [39. What is hibernate configuration file?](https://java2blog.com/hibernate-interview-questions-and-answers/#39_What_is_hibernate_configuration_file)
* [40. Can you list down important annotations used for Hibernate mapping?](https://java2blog.com/hibernate-interview-questions-and-answers/#40_Can_you_list_down_important_annotations_used_for_Hibernate_mapping)
* [41. What are the design patterns used in Hibernate?](https://java2blog.com/hibernate-interview-questions-and-answers/#41_What_are_the_design_patterns_used_in_Hibernate)
* [42. Given a Customer class, you need to persist customer data in the database with customer ID as primary key. Please list down the changes you need to make?](https://java2blog.com/hibernate-interview-questions-and-answers/#42_Given_a_Customer_class_you_need_to_persist_customer_data_in_the_database_with_customer_ID_as_primary_key_Please_list_down_the_changes_you_need_to_make)

**1. What is ORM?**

ORM stands for Object Relational mapping. It is programming paradigm which is used to persist java objects  to database tables.

**2. What is Hibernate?**

Hibernate is pure ORM tool which is used to save old java objects to database tables. The main goal of hibernate to avoid old JDBC code and focus more on business logic. You need to write less code with it.

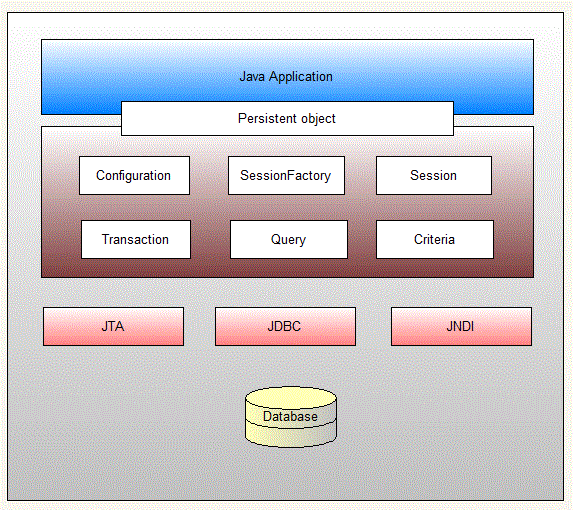
**3. What are advantages of Hibernate?**

Advantages of Hibernate are:

* Lazy Loading
* Caching
* You do not need to maintain JDBC code , Hibernate takes care of it.
* You need to write less code
* It provides high level object oriented API

**4. Explain architecture of Hibernate?**

Following is detail architecture of hibernate with core classes.



Hibernate is like a bridge between java code and relation database and provide object oriented API to deal with JDBC tasks.

**5. What are some core interfaces of hibernate?**

* Session
* SessionFactory
* Configuration
* Transaction
* Query and Criteria interface.

**6. Explain brief about Session interface used in hibernate?**

Session interface is primarily used by hibernate application. Session is light weight,short lived objects which are inexpensive to create and destroy. It allows you to create query objects to retrieve persistent objects.It wraps JDBC connection Factory for Transaction.It holds a mandatory (first-level) cache of persistent objects, used when navigating the object graph or looking up objects by identifier .

**7. Are session objects thread safe?**

Session objects are not thread safe and need to be used in single thread. Every thread should have their own session object and close it once it is done with the work.

**8. Explain brief about SessionFactory object used in hibernate?**

SessionFactory is heavy weight object and it should be created one per database. SessionFactory object is shared by multiple sessions.

**9. What is Configuration class in hibernate?**

Configuration class is used to load required hibernate configuration. It is used to bootstrap hibernate and it is used to locate to hibernate mapping file.

**10. Explain brief about Criteria API in Hibernate?**

Criteria API is a API for retrieving entities by composing Criterion objects also referred as Criterion query.

Criteria API is elegant way for building dynamic queries on the persistence database.

Let’s understand with the help of example.  
You have Employee class with two attributes i.e. name and age.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50 | **package** org.arpit.java2blog;    */\**  *\* This is our model class and it corresponds to Employee table in database*  *\*/*  @Entity  @Table(name="EMPLOYEE")  public **class** Employee {        @Id      @Column(name="id")      @GeneratedValue(strategy=GenerationType.IDENTITY)  **int** id;        @Column(name="name")  **String** name;        @Column(name="age")  **int** age;        public Employee(**String** name) {  **this**.name = name;      }        public **int** getId() {  **return** id;      }        public **void** setId(**int** id) {  **this**.id = id;      }        public **String** getName() {  **return** name;      }        public **void** setName(**String** name) {  **this**.name = name;      }        public **int** getAge() {  **return** age;      }        public **void** setAge(**int** age) {  **this**.age = age;      }  } |

You want to get list of employees whose name start with A and age is greater than 30.

You can write query as below.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Criteria criteria = session.createCriteria(Employee.**class**);  criteria.add(Restrictions.like("name","A%");  criteria.add(Restrictions.gt("age",30);  List<Employee> employeeList = criteria.list(); |

**11. Explain brief about Query interface in Hibernate?**

Query interface is object oriented representation of Hibernate Query. You can get query object bu calling Session.createQuery() method.

Here is simple example to execute Native query using Query APIs.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | SQLQuery query = session.createSQLQuery("select name, age from Employee");  List<**Object**[]> rows = query.list();  **for**(**Object**[] row : rows){      Employee e = **new** Employee();      e.setName(row[0].toString());      e.setAge(**Integer**.parseInt(row[1].toString()));      System.out.println(e);  } |

**12. What are differences between openSession and getCurrentSession in hibernate?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **openSession** | **getCurrentSession** |
| Session object | It always create new Session object | It creates a new Session if not exists , else use same session which is in current hibernate context |
| Flush and close | You need to explicitly flush and close session objects | You do not need to flush and close session objects, it will be automatically taken care by Hibernate internally |
| Performance | In single threaded environment , It is slower than getCurrentSession | In single threaded environment , It is faster than getOpenSession |
| Configuration | You do not need to configure any property to call this method | You need to configure additional property “hibernate.current\_session\_context\_class” to call getCurrentSession method, otherwise it will throw exceptions. |

You can also refer [difference between opensession and getCurrentSession in hibernate](https://java2blog.com/difference-opensession-getcurrentsession-hibernate/).

**13. What are differences between get and load methods in hibernate?**

This is one of the most asked hibernate interview questions.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **get** | **load** |
| Database retrieval | It always hits the database | It does not hit database |
| If null | If it does not get the object with id, it returns null | If it does get the object with id, it throws ObjectNotFoundException |
| Proxy | It returns real object | It returns proxy object |
| Use | If you are not sure if object with id exists or not, you can use get | If you are sure about existence of object, you can use load |

Read here: [Difference between get and load in Hibernate](https://java2blog.com/difference-between-get-and-load-in-hibernate/)

**14. Can you declare Entity class as final in hibernate?**

Yes, you can declare entity class as final but it is not considered as a good practice because hibernate uses proxy pattern for lazy initialisation, If you declare it as final then hibernate won’t be able to create sub class and won’t be able to use proxy pattern, so it will limit performance and improvement options.

**15. Differentiate between first level cache and second level cache?**

This is one of most important hibernate interview questions.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **First level Cache** | **Second level Cache** |
| Association | It is associated at Session level. | It is associated at SessionFactory level and is generally exists one per application. |
| Default | It is enabled by default | It is not enabled by default, you need to enable it explicitly |

**16. What are states of object in hibernate?**

**Transient :** If object is in transient state, it means it was never associated with session and just created.

**Persistent :** If object is in persistent state, it means it is associated with session and you just saved or retrieved object from database.

**Detached :**If object is in detached state, it means session is closed and object is no more part of session. If you call merge or update, object goes back to persistent state.

**17. Does entity class in hibernate require no arg constructor?**

Yes, Entity class in hibernate requires no arg constructor because Hibernate use reflection to create instance of entity class and it mandates no arg constructor in Entity class.

**18. What is HQL?**

HQL stands for Hibernate query language. It is very simple, efficient and object oriented query languages which simplifies complex SQL queries. Instead of table, you use object to write queries.

**19. What are differences between save and saveOrUpdate method of session object?**

save(): stores object in database. It generates identifier for tQhe object and returns it. If object already exists in database, it will throw an error.

saveOrUpdate(): SaveOrUpdate method save the object if identifies does not exist. If it exists , it calls update method.

**20. What are two types of Collections in hibernate?**

* Sorted Collection
* Ordered Collection

**21. What are differences between sorted and ordered collection in hibernate?**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Sorted Collection** | **Ordered Collection** |
| Sorting | Sorted collection uses java’s sorting API to sort the collection. | Ordered Collections uses order by clause while retrieval of objects |
| Default | It is enabled by default | It is not enabled by default, you need to enable it explicitly |

**22. What is lazy loading in hibernate?**

It is used for performance improvement. Lazy loading means when you load parent, child objects won’t get loaded until requested. Infact hibernate loads child objects automatically when you access them. It improves performance as you are not loading whole object at once.

**23. What is LazyInitializationException in Hibernate? Why do you get it?**

This exception generally occurs when you use lazy loading and child objects are accessed after closing session.

That’s all about hibernate interview questions.

**24. What are different collection types available in hibernate?**

There are 5 collection types available in Hibernate for one to many relationship mappings.

* Bag
* Set
* List
* Map
* Array

**25. How can you log sql queries executed by Hibernate?**

You can set hibernate.show\_sql to true for logging sql queries in Hibernate configuration file.

|  |  |
| --- | --- |
| 1  2  3 | <property name="hibernate.show\_sql">**true**</property> |

2.4M

How to Upgrade to Linux Mint 20 from 19.3 [Step by Step]

**26. Can you execute native sql in hibernate?**

Yes, you can use execute native sql with the help of SQLQuery object in Hibernate.

Here is the example to fetch list of employees from database and create corrresponding Employee objects.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | *// Crate session object*  SessionFactory sf = HibernateUtil.getSessionFactory();  Session session = sf.getCurrentSession();    *// Get list of employees with help of SQLQuery.*  Transaction tx = session.beginTransaction();  SQLQuery sqlQuery = session.createSQLQuery("select id, name, age from Employee");  List<**Object**[]> rows = sqlQuery.list();  **for**(**Object**[] row : rows){      Employee employee = **new** Employee();      employee.setId(**Long**.parseLong(row[0].toString()));      employee.setName(row[1].toString());      employee.setAge(**Integer**.parseInt(row[2].toString()));      System.out.println(employee);  } |

**27. What are inheritance mapping strategies are available in Hibernate?**

There are three strategies supported by Hibernate. You can use xml files or JPA annotation to implement them.

* [Table per Hierarchy](https://java2blog.com/hibernate-inheritancetable-per-class/ “Table per Hierarchy”)
* [Table per concrete class](https://java2blog.com/hibernate-inheritancetable-per-concrete/ “Table per concrete class”)
* [Table per subclass](https://java2blog.com/hibernate-inheritancetable-per-subclass/ “Table per subclass”)

**28. How to make an class immutable in Hibernate?**

You can mark class as mutable=false and the class will become immutable.

**29. What is autamatic dirty checking in Hibernate?**

if object is modified in the transaction, then its state will be updated automatically when you committ the transaction.

Here is an example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | SessionFactory factory = cfg.buildSessionFactory();  Session session = factory.openSession();  Transaction tx=session.beginTransaction();    Employee e1 = (Employee) session.get(Employee.**class**, **Integer**.valueOf(10001));    e1.setAge(32);    tx.commit();  session.close(); |

Here, we are updating age of employe after getting employee instance and this is updated automatically when we have committed the transaction.

**30. What do you understand by Hibernate tuning?**

Hibernate tuning is process of optimizing performance of Hibernate application.

Some of the performance tuning strategies are:

* Data Caching
* Session management
* SQL optimization

**31. What types of joins can you use in Hibernate?**

There are multiple ways to use join in Hibernate.

* Using relationships such as one-to-one, one-to-many or many-to-many
* Using joins in native SQL query
* Using joins in HQL

**32. What is dialect in Hibernate?**

Dialect specifies type of database used in Hibernate, so that Hibernate can generate type of SQL statements.

For example:  
Dialect for mysql database: org.hibernate.dialect.MySQL5Dialect  
Dialect for sqlserver database: org.hibernate.dialect.SQLServer2005Dialect

**33. Can you share some of the databases supported by Hibernate?**

Here are some of the databases supported by Hibernate:

* MySQL
* SQLServer
* PostgreSQL
* FrontBaase
* Oracle
* Sybase SQL Server

**34. What is Named queries in Hibernate?**

Named queries helps you to group HQL/SQL statements at single location. You can refer it by name in code when you want to use them. It helps you to avoid code mess that can happend because of scattered queriest through the project.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58 | **package** org.arpit.java2blog;  @NamedQueries(      {          @NamedQuery(          name = "getEmployeeByName",          query = "from Employee e where e.name = :name"          )      }  )    */\**  *\* This is our model class and it corresponds to Employee table in database*  *\*/*  @Entity  @Table(name="EMPLOYEE")  public **class** Employee {        @Id      @Column(name="id")      @GeneratedValue(strategy=GenerationType.IDENTITY)  **int** id;        @Column(name="name")  **String** name;        @Column(name="age")  **int** age;        public Employee(**String** name) {  **this**.name = name;      }        public **int** getId() {  **return** id;      }        public **void** setId(**int** id) {  **this**.id = id;      }        public **String** getName() {  **return** name;      }        public **void** setName(**String** name) {  **this**.name = name;      }        public **int** getAge() {  **return** age;      }        public **void** setAge(**int** age) {  **this**.age = age;      }  } |

You can execute the query in main class as below:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | TypedQuery query = session.getNamedQuery("getEmployeeByName");      query.setParameter("name","John");        List<Employee> employees=query.getResultList();      System.out.println(employees); |

**35. What is Query cache in Hibernate?**

If you have queries that run over and over again,with same parameters, then query caching can you performance improvements in the application.

Benifit from caching query results is very limited and dependent on the usage of the application. This is reason Hibernate disables query level cache by default.

To enable, you need to do following:

1. Set hibernate.cache.use\_query\_cache in hibernate config file.

|  |  |
| --- | --- |
| 1  2  3 | <property name="hibernate.cache.use\_query\_cache">**true**</property> |

1. You need to enable query cache for specific queries. Here is an example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | List<Employees> employee = session.createQuery("from Employee e where e.name = :name")          .setEntity("employee", employee)          .setMaxResults(15)          .setCacheable(**true**)          .setCacheRegion("employeeReg")          .list(); |

**36. What are benefits of Criteria API in Hibernate?**

Criteria API can be used to fetch entity from database using object oriented apporach.  
Here are the advantages of criteria APIs.

* Criteria API is cleanm nice and Object oriented API.
* Criteria API provides Projection that can be used for aggregate functions like sum(), min() etc.
* You can write more flexible and dynamic queries as compared to HQL.
* It has addOrder() method which can be used for ordering the results.

**37. What is cascading in Hiberante and can you list types of cascading?**

Most of the time, if Entity relationships depend on existence of other entity.  
For example:  
In case of Employee-Address relationship, if Employee is removed from database, then Address does not make sense of its own. So when you remove Employee from database, then its associated Address should also be removed.

You can use Cascading to achieve this. When you perform an action on entity, same action can be performed on associated entity.

Here are the Cascading types supported by Hibernate:

|  |  |
| --- | --- |
| **Cascade Operation** | **Description** |
| ALL | All the operations will be applied to parent entity’s associated entity. All operation includes DETACH, MERGE, PERSIST, REFRESH, REMOVE etc. |
| DETACH | If parent entitiy is detached from context, then the associated entity will also be detached. |
| MERGE | If parent entitiy is merged into the context, then the associated entity will also be merged. |
| PERSIST | If parent entitiy is persisted into the context, then the associated entity will also be persisted. |
| REFRESH | If parent entitiy is refreshed in the curent persistence context, then the associated entity will also be persisted. |
| REMOVE | If parent entitiy is removed from the curent persistence context, then the associated entity will also be removed. |

**38. How to integrate log4j with Hibernate?**

For log4j configuration, you can follow following steps:

1. Add log4j dependencies for maven project. If it is not maven project, then add required log4j jars to classpath.
2. Create [log4j.xml](https://java2blog.com/log4j-xml-configuration-example/ “log4j.xml”) or log4j.properties and put in the classpath.
3. Use DOMConfigurator or PropertyConfigurator to configure log4j in static block for standalone application. If you have web application, then you can use ServletContextListener to configure it.

**39. What is hibernate configuration file?**

Hibernate configuration file contains database configurations such as database url, username, password and dialect etc and is used to initialize SessionFactory. It also contains mapping files and entity class details.

**40. Can you list down important annotations used for Hibernate mapping?**

Here are some important annotations that can be used for Hibernate mapping.

@Entity : It is used to define class as entity bean.  
@Table : It is used to define table name in database corrresponding to Entity.  
@Id : It is used to define primary key in the entity bean.  
@Column: It is used to define column properties in database corrresponding to entity bean property.  
@OneToOne, @ManyToOne, @ManyToMany: These annotations are used to define relationships between various entities.  
@JoinColumn : It is used to specify a mapped column for joining an entity association.

Here is an example:  
We are using two entities Country and Capital with one to one relationship.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59 | **package** org.arpit.java2blog;  import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.Id;  import javax.persistence.JoinColumn;  import javax.persistence.OneToOne;  import javax.persistence.Table;    @Entity  @Table(name="COUNTRY")  public **class** Country {        @Id      @Column(name="Country\_Name")  **String** countryName ;        @OneToOne      @JoinColumn(name="Capital\_Name")      Capital capital;        @Column(name="Country\_Population")  **long** countryPopulation;        public Country()      {        }        public Country(**String** countryName, **long** countryPopulation) {  **this**.countryName = countryName;  **this**.countryPopulation = countryPopulation;  }        public **long** getCountryPopulation() {  **return** countryPopulation;  }        public **void** setCountryPopulation(**long** countryPopulation) {  **this**.countryPopulation = countryPopulation;  }        public **String** getCountryName() {  **return** countryName;      }        public **void** setCountryName(**String** countryName) {  **this**.countryName = countryName;      }        public Capital getCapital() {  **return** capital;  }        public **void** setCapital(Capital capital) {  **this**.capital = capital;  }  } |

Capital.java

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45 | **package** org.arpit.javapostsforlearning;import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.Id;  import javax.persistence.Table;    @Entity  @Table(name="CAPITAL")  public **class** Capital {         @Id       @Column(name="Capital\_Name")  **String** capitalName;         @Column(name="Capital\_Population")  **long** capitalPopulation;         public Capital()       {         }       public Capital(**String** capitalName, **long** capitalPopulation) {  **super**();  **this**.capitalName = capitalName;  **this**.capitalPopulation = capitalPopulation;       }         public **String** getCapitalName() {  **return** capitalName;       }         public **void** setCapitalName(**String** capitalName) {  **this**.capitalName = capitalName;       }         public **long** getCapitalPopulation() {  **return** capitalPopulation;       }         public **void** setCapitalPopulation(**long** capitalPopulation) {  **this**.capitalPopulation = capitalPopulation;       }    } |

**41. What are the design patterns used in Hibernate?**

Here are some of the design patterns used in Hibernate:

* [Proxy pattern](https://java2blog.com/proxy-design-pattern-in-java) for lazy loading
* Factory pattern in SessionFactory
* Query object for Criteria API
* Data Mapper: A pattern in which layer of matters that flows data between object and databases while keeping them independent of each other
* Domain Model pattern: An object model of the domain that incorporates both data and behavior.

**42. Given a Customer class, you need to persist customer data in the database with customer ID as primary key. Please list down the changes you need to make?**

Here is definition of table CUSTOMER.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | CREATE TABLE CUSTOMER (  id **int**(11) **NOT** **NULL** AUTO\_INCREMENT,  Customer\_Name varchar(255) **DEFAULT** **NULL**,  email varchar(255) **DEFAULT** **NULL**,  PRIMARY KEY (id)  ) |

Here is the definition of Customer class

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44 | **package** org.arpit.java2blog.model;    */\**  *\* This is our model class and it corresponds to Customer table in database*  *\*/*  =  public **class** Customer{    **int** id;    **String** customerName;    **String** email;        public Customer() {  **super**();      }      public Customer(**String** customerName,**String** email) {  **super**();  **this**.customerName=customerName;  **this**.email=email;      }      public **String** getCustomerName() {  **return** customerName;      }      public **void** setCustomerName(**String** customerName) {  **this**.customerName = customerName;      }      public **String** getEmail() {  **return** email;      }      public **void** setEmail(**String** email) {  **this**.email = email;      }      public **int** getId() {  **return** id;      }      public **void** setId(**int** id) {  **this**.id = id;      }    } |

Here are the changes that needs to be done:

1. Annotate the Customer class with @Entity to declare it as Hibernate entity
2. Use @Table annotation to map class to table name in databse
3. Use @id and @GeneratedValue annotation to make id as primary key
4. Use @Column to map attribute to corresponding database column

Here is the code:

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57 | **package** org.arpit.java2blog.model;    import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.GeneratedValue;  import javax.persistence.GenerationType;  import javax.persistence.Id;  import javax.persistence.Table;    */\**  *\* This is our model class and it corresponds to Customer table in database*  *\*/*  @Entity  @Table(name="CUSTOMER")  public **class** Customer{        @Id      @Column(name="id")      @GeneratedValue(strategy=GenerationType.IDENTITY)  **int** id;        @Column(name="Customer\_Name")  **String** customerName;        @Column(name="email")  **String** email;        public Customer() {  **super**();      }      public Customer(**String** customerName,**String** email) {  **super**();  **this**.customerName=customerName;  **this**.email=email;      }      public **String** getCustomerName() {  **return** customerName;      }      public **void** setCustomerName(**String** customerName) {  **this**.customerName = customerName;      }      public **String** getEmail() {  **return** email;      }      public **void** setEmail(**String** email) {  **this**.email = email;      }      public **int** getId() {  **return** id;      }      public **void** setId(**int** id) {  **this**.id = id;      }    } |

# Spring boot interview questions for 10 years experience

Table of Contents [[hide](https://java2blog.com/spring-boot-interview-questions/)]

* [1. What is Spring boot?](https://java2blog.com/spring-boot-interview-questions/#1_What_is_Spring_boot)
* [2. Why did you use Spring boot in your application?](https://java2blog.com/spring-boot-interview-questions/#2_Why_did_you_use_Spring_boot_in_your_application)
* [3. Can you list advantages of Spring boot?](https://java2blog.com/spring-boot-interview-questions/#3_Can_you_list_advantages_of_Spring_boot)
* [4. What are disadvantages of Spring boot?](https://java2blog.com/spring-boot-interview-questions/#4_What_are_disadvantages_of_Spring_boot)
* [5. How can you override default properties in Spring boot Project?](https://java2blog.com/spring-boot-interview-questions/#5_How_can_you_override_default_properties_in_Spring_boot_Project)
* [5. How can you run Spring boot application on custom port?](https://java2blog.com/spring-boot-interview-questions/#5_How_can_you_run_Spring_boot_application_on_custom_port)
* [6. What is Spring boot starter and how it is useful?](https://java2blog.com/spring-boot-interview-questions/#6_What_is_Spring_bootstarter_and_how_it_is_useful)
* [7. Can we use Spring boot with applications which are not using Spring?](https://java2blog.com/spring-boot-interview-questions/#7_Can_we_use_Spring_boot_with_applications_which_are_not_using_Spring)
* [8. What is name of the configuration file which you use in Spring boot?](https://java2blog.com/spring-boot-interview-questions/#8_What_is_name_of_the_configuration_file_which_you_use_in_Spring_boot)
* [9. What is DevTools in Spring boot?](https://java2blog.com/spring-boot-interview-questions/#9_What_is_DevTools_in_Spring_boot)
* [10. What is actuator in Spring boot?](https://java2blog.com/spring-boot-interview-questions/#10_What_is_actuator_in_Spring_boot)
* [11. How can you implement Spring security in Spring boot application?](https://java2blog.com/spring-boot-interview-questions/#11_How_can_you_implement_Spring_security_in_Spring_boot_application)
* [12. Have you used @SpringBootApplication annotation in Spring boot project?](https://java2blog.com/spring-boot-interview-questions/#12_Have_you_usedSpringBootApplication_annotation_in_Spring_boot_project)
* [13. What are embedded containers which are supported by Spring Boot?](https://java2blog.com/spring-boot-interview-questions/#13_What_are_embedded_containers_which_are_supported_by_Spring_Boot)
* [14. Have you used ActiveMQ in Spring Boot application? Do you know how to configure external ActiveMQ?](https://java2blog.com/spring-boot-interview-questions/#14_Have_you_used_ActiveMQ_in_Spring_Boot_application_Do_you_know_how_to_configure_external_ActiveMQ)
* [15. How can you configure logging in Spring boot application?](https://java2blog.com/spring-boot-interview-questions/#15_How_can_you_configure_logging_in_Spring_boot_application)
* [16. Which embedded servers are used in a Spring Boot application](https://java2blog.com/spring-boot-interview-questions/#16_Which_embedded_servers_are_used_in_a_Spring_Boot_application)
* [17. What are the different ways you can leverage to run and deploy a Spring Boot application](https://java2blog.com/spring-boot-interview-questions/#17_What_are_the_different_ways_you_can_leverage_to_run_and_deploy_a_Spring_Boot_application)
* [18. Does Spring Boot replace Spring MVC, Spring REST, etc…](https://java2blog.com/spring-boot-interview-questions/#18_Does_Spring_Boot_replace_Spring_MVC_Spring_REST_etc8230)
* [19. Does Spring Boot code run faster than regular Spring code](https://java2blog.com/spring-boot-interview-questions/#19_Does_Spring_Boot_code_run_faster_than_regular_Spring_code)
* [20. What is Spring Boot starter parent?](https://java2blog.com/spring-boot-interview-questions/#20_What_is_Spring_Boot_starter_parent)
* [21. What are the benefits of Spring Boot starter parent](https://java2blog.com/spring-boot-interview-questions/#21_What_are_the_benefits_of_Spring_Boot_starter_parent)
* [22. What are some of the endpoints exposed by Spring Boot Actuator?](https://java2blog.com/spring-boot-interview-questions/#22_What_are_some_of_the_endpoints_exposed_by_Spring_Boot_Actuator)
* [23. How can you expose all Spring Boot actuator endpoints over HTTP](https://java2blog.com/spring-boot-interview-questions/#23_How_can_you_expose_all_Spring_Boot_actuator_endpoints_over_HTTP)
* [24. How do you inject custom application properties?](https://java2blog.com/spring-boot-interview-questions/#24_How_do_you_inject_custom_application_properties)
* [25. What logger does Spring Boot use?](https://java2blog.com/spring-boot-interview-questions/#25_What_logger_does_Spring_Boot_use)
* [26. How do you integrate Hibernate and JPA with Spring Boot?](https://java2blog.com/spring-boot-interview-questions/#26_How_do_you_integrate_Hibernate_and_JPA_with_Spring_Boot)
* [27. What is Spring Boot auto data source configuration?](https://java2blog.com/spring-boot-interview-questions/#27_What_is_Spring_Boot_auto_data_source_configuration)
* [28. What is JPA?](https://java2blog.com/spring-boot-interview-questions/#28_What_is_JPA)
* [29. Which are the JPA vendor implementations?](https://java2blog.com/spring-boot-interview-questions/#29_Which_are_the_JPA_vendor_implementations)
* [30. What are the benefits of JPA?](https://java2blog.com/spring-boot-interview-questions/#30_What_are_the_benefits_of_JPA)
* [31. What are the various DAO techniques that can be used in Spring Boot?](https://java2blog.com/spring-boot-interview-questions/#31_What_are_the_various_DAO_techniques_that_can_be_used_in_Spring_Boot)
* [32. What is JPQL?](https://java2blog.com/spring-boot-interview-questions/#32_What_is_JPQL)
* [33. What are the benefits of Spring Data JPA?](https://java2blog.com/spring-boot-interview-questions/#33_What_are_the_benefits_of_Spring_Data_JPA)
* [34. What is JpaRepository?](https://java2blog.com/spring-boot-interview-questions/#34_What_is_JpaRepository)
* [35. How does Spring Data REST work in the background? Spring Data REST scans your project for JpaRepository and exposes REST APIs for each entity type for your JpaRepository. In the code below, if you have a custom repository that extends JpaRepository and an entity type of type customer, It will expose a customer’s endpoint. By default, Spring Data REST will create endpoints based on entity type by making use of the simple pluralized form, meaning the first character of the entity type is lowercase then just adds an “s” to the entity. The entity type is Customer, and it will create the /customers endpoint, and that’s the basic approach for how they will expose these REST endpoints. 12345      public interface CustomerRepository extends JpaRepository<Customer, Long>{     }  Here is example of Spring boot CrudRepository. 36. What is HATEOAS, and how does it apply in Spring Data REST?](https://java2blog.com/spring-boot-interview-questions/#35_How_does_Spring_Data_REST_work_in_the_background_Spring_Data_REST_scans_your_project_for_JpaRepository_and_exposes_REST_APIs_for_each_entity_type_for_your_JpaRepository_In_the_code_below_if_you_have_a_custom_repository_that_extends_JpaRepository_and_an_entity_type_of_type_customer_It_will_expose_a_customer8217s_endpoint_By_default_Spring_Data_REST_will_create_endpoints_based_on_entity_type_by_making_use_of_the_simple_pluralized_form_meaning_the_first_character_of_the_entity_type_is_lowercase_then_just_adds_an_8220s8221_to_the_entity_The_entity_type_is_Customer_and_it_will_create_the_customers_endpoint_and_that8217s_the_basic_approach_for_how_they_will_expose_these_REST_endpoints_____________12345__nbspnbspnbspnbspnbsppublic_interface_CustomerRepository_extends_JpaRepositoryltCustomer_Longgtnbspnbspnbspnbspnbspnbsp_______Here_is_example_of_Spring_boot_CrudRepository_36_What_is_HATEOAS_and_how_does_it_apply_in_Spring_Data_REST)
* [37. Which are the advanced features of Spring Data REST?](https://java2blog.com/spring-boot-interview-questions/#37_Which_are_the_advanced_features_of_Spring_Data_REST)
* [38. What is Thymeleaf?](https://java2blog.com/spring-boot-interview-questions/#38_What_is_Thymeleaf)
* [39. Where is the Thymeleaf template processed?](https://java2blog.com/spring-boot-interview-questions/#39_Where_is_the_Thymeleaf_template_processed)
* [40. Which are the few Thymeleaf use cases?](https://java2blog.com/spring-boot-interview-questions/#40_Which_are_the_few_Thymeleaf_use_cases)

In this post, we will see top 15 Spring Boot interview questions with answers.If you want to read more about Spring boot, you can go through [Spring boot tutorial](https://www.java2blog.com/spring-boot-tutorial/).  
If you are looking for below queries then this post will help you as well.

* Spring boot interview questions for 3 years experience
* Spring boot interview questions for 5 years experience
* Spring boot interview questions for 7 years experience

## 1. What is Spring boot?

Spring Boot makes it easier for you to create production ready applications in no time. It is an opinionated view to create Spring application quickly. It follows convention over configuration. In simple terms, it comes with default configurations for most of the Spring projects, you don’t need to do much to bootstrap any spring application.

## 2. Why did you use Spring boot in your application?

As discussed earlier, Spring boot makes it easier for you to create Spring application, it can save a lot of time and efforts.

For example: Let’s say you want to create Spring boot project with activeMQ. You can simply use "spring–boot–starter–activemq" as artifact Id, it will take all the defaults and create Spring application with ActiveMQ configured. Let’s say you don’t want to use inbuilt activeMQ, you can simply override "spring.activemq.broker-url" in application.properties to use external ActiveMQ.

## 3. Can you list advantages of Spring boot?

Advantages of Spring boot are:

* It provides a lot of default configurations which help you to create Spring application faster.
* It comes with embedded tomcat or jetty server, so you don’t have to deploy jar.
* It reduces development code by avoiding a lot of boilerplate code.
* It increases productivity as you can create Spring application quickly.
* It provides a lot of starter project for easy maven integration.You don’t have to worry about version mismatch.
* You can quickly create using sample project using [spring boot initializer](https://start.spring.io/)

## 4. What are disadvantages of Spring boot?

If you want to convert your old spring application to Spring boot application, it may not be straight forward and can be time consuming.

## 5. How can you override default properties in Spring boot Project?

Spring boot provides a lot of properties which can be overridden by specifying them in application.properties.

**For example:** You want to specify prefix and suffix in Spring MVC applications. You can simply do it by putting below properties in application.properties.

spring.mvc.view.prefix: /WEB-INF/  
spring.mvc.view.suffix: .jsp

## 5. How can you run Spring boot application on custom port?

You can simply put server.port properties in application.properties.

**For example:server.port=8050**

## 6. What is Spring boot starter and how it is useful?

Spring boot comes with a lot of starters which is set of convenient dependency descriptors which you can include in your pom.xml.

**For example:** Let’s say you want to work Spring MVC application, you can simply include "**spring–boot–starter–web**" as dependency in pom.xml .

## 7. Can we use Spring boot with applications which are not using Spring?

No, it is not possible as of now. Spring boot is limited to Spring applications only.

## 8. What is name of the configuration file which you use in Spring boot?

Configuration file used in Spring boot projects is application.properties. It is very important file as it is used to override all default configurations.

## 9. What is DevTools in Spring boot?

Spring boot comes with DevTools which is introduced to increase the productivity of developer. You don’t need to redeploy your application every time you make the changes.Developer can simply reload the changes without restart of the server. It avoids pain of redeploying application every time when you make any change. This module will be disabled in production environment.

## 10. What is actuator in Spring boot?

[Spring boot actuator](https://lightrun.com/best-practices/getting-started-with-spring-boot-actuator/) is one of the most important features of Spring boot. It is used to access current state of running application in production environment. There are various metrics which you can use to check current state of the application.

Spring boot actuator provides restful web services end points which you can simply use and check various metrics.  
For example:  
**/metrics :**This restful end point will show you metrics such as free memory, processors, uptime and many more properties,

Spring boot actuator will help you to monitor your application in production environment.Restful end points can be sensitive, it means it will have restricted access and will be shown only to authenticated users. You can change this property by overriding it in application.properties.

## 11. How can you implement Spring security in Spring boot application?

Implementation of Spring security in Spring boot application requires very little configuration. You need to add **spring-boot-starter-security**starter in pom.xml.You need to create Spring config class which will extend WebSecurityConfigurerAdapter and override required method to achieve security in Spring boot application.

You can read more about [Spring boot security example](https://www.java2blog.com/2017/05/spring-boot-spring-security-example.html).

## 12. Have you used @SpringBootApplication [annotation in Spring](https://java2blog.com/annotation-based-configuration-in-spring/) boot project?

@SpringBootApplication annotation was introduced in Spring Boot 1.2.0. This annotation is equivalent to declaring these 3 annotations.

* @Configuration
* @EnableAutoConfiguration
* @ComponentScan

For example:

When you create your main class with Spring boot, you have to use below annotations before Spring boot 1.2.0.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | @Configuration  @EnableAutoConfiguration  @ComponentScan  public **class** SpringBootHelloWorldApplication {  ...  } |

But after Spring boot 1.2.0, you just need to use @SpringBootApplication annotation which will cover above 3 annotations

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | @SpringBootApplication  public **class** SpringBootHelloWorldApplication {  ...  } |

## 13. What are embedded containers which are supported by Spring Boot?

Spring boot contains embedded Tomcat, Jetty and undertow servers.

## 14. Have you used ActiveMQ in Spring Boot application? Do you know how to configure external ActiveMQ?

Spring Boot comes with embedded ActiveMQ.We need to use "spring–boot–starter–activemq" dependency in pom.xml and it will take care of all defaults and will configure ActiveMQ in the project.

If you want to configure external ActiveMQ then you need to just put "spring.activemq.broker-url" in application.properties and provide the URL of external ActiveMQ.

## 15. How can you configure logging in Spring boot application?

2.4M

How to Upgrade to Linux Mint 20 from 19.3 [Step by Step]

Spring Boot comes with support for Java Util Logging, Log4J2 and Logback and it will be pre-configured as Console output.  
Hence,You can simply specify logging.level in application.properties.

|  |  |
| --- | --- |
| 1  2  3 | logging.level.spring.framework=Debug |

It will set Spring framework logs to debug level.  
Let’s say you want to put logs to the file.You can specify logger.file in application.properties.

|  |  |
| --- | --- |
| 1  2  3 | logging.file={java.io.tmpdir}/application.log |

If you want to do logging configuration explicitly, You can also create logback.xml in main/java/resources folder and specify logging configuration in the file. Spring Boot will pick this file and configure logging accordingly.

## 16. Which embedded servers are used in a Spring Boot application

Spring Boot provides an embedded HTTP server so that you can get started quickly and has support for Tomcat, Jetty, and Undertow.

## 17. What are the different ways you can leverage to run and deploy a Spring Boot application

Spring Boot apps can be run as standalone because they include an embedded server enabling it to be run from the IDE or the command line.

For example, if we have an app.jar application that includes your code and embedded server, in the command line, type java -jar app.jar, and it will run your application and spin up the server.

If you want to deploy the application traditionally, you can deploy a war file to an external server like Tomcat, JBoss, and WebSphere, and it can work just like you used it in the past.

For example, if we have a Tomcat deployed somewhere on your corporate network, you can deploy that Spring Boot app with the .war file extension.

As a war file, you only have your code included, and there is no need to have the embedded server because now you are deploying it in a traditional sense.

There is already a Tomcat server installed running elsewhere, and we are simply deploying our war file to that server.

## 18. Does Spring Boot replace Spring MVC, Spring REST, etc…

No, Instead, Spring Boot uses those technologies in the background. Spring Boot uses Spring Core, Spring AOP, Spring MVC, Spring REST, and other technologies in the background, so there is no competition among them. Spring Boot is mainly about configuration, and once you do your configs, then you can do the regular Spring coding just as you have @Component, @Controller, [@Autowired](https://java2blog.com/autowired-annotation-in-spring/) etc.

## 19. Does Spring Boot code run faster than regular Spring code

No, behind the scenes, Spring Boot uses the same code as Spring Framework, and its sole purpose is about making it easier to get started minimizing configuration.

## 20. What is Spring Boot starter parent?

[Spring Boot starter parent](https://java2blog.com/spring-boot-starters/) is a special starter that provides Maven defaults. In your pom.xml you will have an entry for parent, and you will give the actual groupId artifactId and version which is included in the pom.xml directly when using the Spring initializer.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | <**parent**>      <groupId>org.springframework.boot</groupId>      <artifactId>spring-boot-starter-**parent**</artifactId>      <version>2.5.3</version>      <relativePath/> <!-- lookup **parent** from repository -->    </**parent**> |

The maven defaults defined in the starter parent are default compiler level which is Java 8, UTF-8 source encoding, and other features out there.

To override a default, set it as a property.

|  |  |
| --- | --- |
| 1  2  3  4  5 | <properties>      <java.version>11</java.version>  </properties> |

For Spring Boot starter dependencies, there is no need to list the version as they inherit the version from the starter parent, which is great for maintenance and also helps to make sure that all the dependencies that you are using are compatible.

## 21. What are the benefits of Spring Boot starter parent

Default Maven configuration for the Java version, UTF-encoding, and others.

Dependency management by using the version on parent only and all Spring Boot starter dependencies extend version from the parent.

Provides default configuration of the Spring Boot plugin as shown below.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <build>      <plugins>        <plugin>          <groupId>org.springframework.boot</groupId>          <artifactId>spring-boot-maven-plugin</artifactId>        </plugin>      </plugins>    </build> |

## 22. What are some of the endpoints exposed by Spring Boot Actuator?

All the endpoints are prefixed /actuator and each request returns a JSON response from the server to authorized users only.

/health checks the status of your application to see if your app is up or down.

/info gives you information about your application but is empty by default, and you need to customize the info endpoint by updating the application.properties with your app info.

|  |  |
| --- | --- |
| 1  2  3  4  5 | info.app.name = My **Super** Cool App  info.app.description = An app that does magic  info.app.version = 2.1.0 |

/auditevents inspect events for your application.

/beans return a list of all beans registered in the Spring application context.

/mappingd returns a list of all @RequestMapping paths.

## 23. How can you expose all Spring Boot actuator endpoints over HTTP

Use a wildcard “\*” to expose all endpoints, and you can also expose the individual endpoints using a comma-delimited list.

|  |  |
| --- | --- |
| 1  2  3 | management.endpoints.web.exposure.include=\* |

## 24. How do you inject custom application properties?

Your app needs to be configurable to prevent hard coding of values by reading values from a properties file.

By default, Spring Boot reads information from a standard properties file located at src/main/resources/application.properties.

You can define any custom properties in this file, and your Spring Boot app can access properties using @Value annotation.

## 25. What logger does Spring Boot use?

[Spring Boot](https://java2blog.com/spring-boot-tutorial/) provides support for Logback, Log4j2, and Java Util Logging in its default configuration but uses the Apache Commons logging for all internal logging.

## 26. How do you integrate Hibernate and JPA with Spring Boot?

Spring Boot will automatically configure your data source for you based on entries from the Maven POM file. In your POM file, you give a reference to your JDBC Driver, Spring Data (ORM) and setup database connection properties in application.properties, and Spring Boot will use this information to create a data source for you.

## 27. What is Spring Boot auto data source configuration?

Based on the configurations provided, Spring Boot automatically creates Beans for DataSource and EntityManager, and then you can inject these into your data access objects (DAO). EntityManager is a class from the Java persistence API (JPA). In Spring Boot, Hibernate is the default implementation of JPA. The EntityManager is similar to HibernateSessionFactory, and EntityManager can serve as a wrapper for a Hibernate Session object.

## 28. What is JPA?

JPA is a standard API for Object-to-Relational-Mapping and acts as a specification that defines a set of interfaces and requires an implementation to be usable.

## 29. Which are the JPA vendor implementations?

One of the implementations is Hibernate, just like java coding, they take those interfaces and provide an implementation of those interfaces. Another implementation is EclipseLink, which also have their implementation of the JPA specification. There are other implementations, but Hibernate is the most popular implementation of the JPA specification.

## 30. What are the benefits of JPA?

By having a standard API, you are not locked to vendor implementation, so you can maintain portable, flexible code by coding to JPA specifications. Theoretically, you can switch vendor implementations. For example, if Hibernate stops supporting their products, you can switch to EclipseLink without vendor lock-in because you are coding to the actual JPA standard API.

## 31. What are the various DAO techniques that can be used in Spring Boot?

To implement a Data Access Object in Spring boot, you can use EntityManager but leverage native Hibernate API, EntityManager, and standard JPA API, and finally Spring Data JPA which is the most common nowadays.

## 32. What is JPQL?

JPQL stands for Jakarta Persistence Query Language which is a query language for JPA and has the following syntax.

|  |  |
| --- | --- |
| 1  2  3 | SELECT a FROM Author a ORDER BY a.firstName, a.lastName |

## 33. What are the benefits of Spring Data JPA?

When you create a DAO and plug in your entity type, and primary key, Spring Boot will give you CRUD implementations for free, which help us minimize our boiler-plate DAO code. Depending on the use case, you can get more than a 70% reduction in code.

## 34. What is JpaRepository?

Spring Data provides an interface called JpaRepository and exposes methods, some of which are inherited from parents, and these are CRUD methods that you can use. Some of these methods include findAll(), findById(), save(), deleteById() and others.

## 35. How does Spring Data REST work in the background? Spring Data REST scans your project for JpaRepository and exposes REST APIs for each entity type for your JpaRepository.

**In the code below, if you have a custom repository that extends JpaRepository and an entity type of type customer, It will expose a customer’s endpoint.**

**By default, Spring Data REST will create endpoints based on entity type by making use of the simple pluralized form, meaning the first character of the entity type is lowercase then just adds an “s” to the entity.**

**The entity type is Customer, and it will create the /customers endpoint, and that’s the basic approach for how they will expose these REST endpoints.**

|  |  |
| --- | --- |
| **1**  2  3  4  5 | public **interface** CustomerRepository **extends** JpaRepository<Customer, **Long**>{        } |

## Here is example of [Spring boot CrudRepository](https://java2blog.com/spring-boot-crudrepository/).

## 36. What is HATEOAS, and how does it apply in Spring Data REST?

**HATEOAS stands for Hypermedia as the Engine of Application State and Spring Data REST is HATEOAS compliant, meaning that it provides information to access REST interfaces and you can also think of it as meta-data for REST data. For a collection, meta-data includes page size, total elements, and pages.**

## 37. Which are the advanced features of Spring Data REST?

**There is support for pagination, sorting, searching and you can also extend and add custom queries with JPQL. You can also customize the REST API by making use of the Query Domain-specific Language or the Query DSL.**

## 38. What is Thymeleaf?

**Thymeleaf is a Java templating engine commonly used to generate HTML views on web pages apps. However, it is a general-purpose templating engine meaning you can use it outside of web apps.**

## 39. Where is the Thymeleaf template processed?

**In a web app, the Thymeleaf template is processed on the server, results included in HTML, and returned to the browser.**

## 40. Which are the few Thymeleaf use cases?

**We can make use of an email template where when a customer signs up for service; then we will send them a welcome email.**

**We can also use a CSV template to generate a monthly CSV then upload it to Google drive.**

**We can use a pdf template to generate a travel confirmation PDF then send it via an email attachment.**

**That’s all about Spring boot interview questions for 10 years experience**

# Top 100+ Java coding interview questions

I have been posting [data structure](https://java2blog.com/data-structures-java/) and coding interview questions on various topics such as Array, Queue, Stack, Binary tree, LinkedList, String, Number, [ArrayList](https://java2blog.com/arraylist-in-java-with-example/), etc. So I am consolidating a list of java coding interview questions to create an index post. I will keep adding links to this post whenever I will add new java coding interview question.

These are frequently asked java coding interview questions.

Table of Contents [[hide](https://java2blog.com/java-coding-interview-questions/)]

* [String](https://java2blog.com/java-coding-interview-questions/#String)
  + [Question 1 : How to reverse a String in java? Can you write a program without using any java inbuilt methods?](https://java2blog.com/java-coding-interview-questions/#Question_1_How_to_reverse_a_String_in_java_Can_you_write_a_program_without_using_any_java_inbuilt_methods)
  + [Question 2 : Write a java program to check if two Strings are anagram in java?](https://java2blog.com/java-coding-interview-questions/#Question_2_Write_a_java_program_to_check_if_two_Strings_are_anagram_in_java)
  + [Question 3 : Write a program to check if String has all unique characters in java?](https://java2blog.com/java-coding-interview-questions/#Question_3_Write_a_program_to_check_if_String_has_all_unique_characters_in_java)
  + [Question 4 : How to check if one String is rotation of another String in java?](https://java2blog.com/java-coding-interview-questions/#Question_4_How_to_check_if_one_String_is_rotation_of_another_String_in_java)
  + [Question 5 : How to find duplicate characters in String in java?](https://java2blog.com/java-coding-interview-questions/#Question_5_How_to_find_duplicate_characters_in_String_in_java)
  + [Question 6 : Find first non repeated character in String in java?](https://java2blog.com/java-coding-interview-questions/#Question_6_Find_first_non_repeated_character_in_String_in_java)
  + [Question 7 : Find all substrings of String in java?](https://java2blog.com/java-coding-interview-questions/#Question_7_Find_all_substrings_of_String_in_java)
  + [Question 8 : Find length of String without using any inbuilt method in java?](https://java2blog.com/java-coding-interview-questions/#Question_8_Find_length_of_String_without_using_any_inbuilt_method_in_java)
  + [Question 9 : Write a program to print all permutations of String in java?](https://java2blog.com/java-coding-interview-questions/#Question_9_Write_a_program_to_print_all_permutations_of_String_in_java)
* [Array](https://java2blog.com/java-coding-interview-questions/#Array)
  + [Question 10 : Write java Program to Find Smallest and Largest Element in an Array.](https://java2blog.com/java-coding-interview-questions/#Question_10_Write_java_Program_to_Find_Smallest_and_Largest_Element_in_an_Array)
  + [Question 11 : Find missing number in the array.](https://java2blog.com/java-coding-interview-questions/#Question_11_Find_missing_number_in_the_array)
  + [Question 12 : Search an element in rotated and sorted array.](https://java2blog.com/java-coding-interview-questions/#Question_12_Search_an_element_in_rotated_and_sorted_array)
  + [Question 13 : Find minimum element in a sorted and rotated array.](https://java2blog.com/java-coding-interview-questions/#Question_13_Find_minimum_element_in_a_sorted_and_rotated_array)
  + [Question 14: Find second largest number in an array](https://java2blog.com/java-coding-interview-questions/#Question_14_Find_second_largest_number_in_an_array)
  + [Question 15 : Find the number occurring odd number of times in an array](https://java2blog.com/java-coding-interview-questions/#Question_15_Find_the_number_occurring_odd_number_of_times_in_an_array)
  + [Question 16 : Find minimum number of platforms required for railway station](https://java2blog.com/java-coding-interview-questions/#Question_16_Find_minimum_number_of_platforms_required_for_railway_station)
  + [Question 17 : Find a Pair Whose Sum is Closest to zero in Array](https://java2blog.com/java-coding-interview-questions/#Question_17_Find_a_Pair_Whose_Sum_is_Closest_to_zero_in_Array)
  + [Question 18 : Given a sorted array and a number x, find the pair in array whose sum is closest to x](https://java2blog.com/java-coding-interview-questions/#Question_18_Given_a_sorted_array_and_a_number_x_find_the_pair_in_array_whose_sum_is_closest_to_x)
  + [Question 19 : Find all pairs of elements from an array whose sum is equal to given number](https://java2blog.com/java-coding-interview-questions/#Question_19_Find_all_pairs_of_elements_from_an_array_whose_sum_is_equal_to_given_number)
  + [Question 20: Given an array of 0’s and 1’s in random order, you need to separate 0’s and 1’s in an array.](https://java2blog.com/java-coding-interview-questions/#Question_20_Given_an_array_of_0s_and_1s_in_random_order_you_need_to_separate_0s_and_1s_in_an_array)
  + [Question 21 : Separate odd and even numbers in an array](https://java2blog.com/java-coding-interview-questions/#Question_21_Separate_odd_and_even_numbers_in_an_array)
  + [Question 22 : Given an array containing zeroes, ones and twos only. Write a function to sort the given array in O(n) time complexity.](https://java2blog.com/java-coding-interview-questions/#Question_22_Given_an_array_containing_zeroes_ones_and_twos_only_Write_a_function_to_sort_the_given_array_in_On_time_complexity)
  + [Question 23 : Find local minima in array](https://java2blog.com/java-coding-interview-questions/#Question_23_Find_local_minima_in_array)
  + [Question 24 : Sliding window maximum in java](https://java2blog.com/java-coding-interview-questions/#Question_24_Sliding_window_maximum_in_java)
  + [Question 25 : Count number of occurrences (or frequency) of each element in a sorted array](https://java2blog.com/java-coding-interview-questions/#Question_25_Count_number_of_occurrences_or_frequency_of_each_element_in_a_sorted_array)
  + [Question 26 : Find subarrays with given sum in an array.](https://java2blog.com/java-coding-interview-questions/#Question_26_Find_subarrays_with_given_sum_in_an_array)
  + [Question 27 : Find peak element in the array.](https://java2blog.com/java-coding-interview-questions/#Question_27_Find_peak_element_in_the_array)
  + [Question 28 : Find leaders in an array.](https://java2blog.com/java-coding-interview-questions/#Question_28_Find_leaders_in_an_array)
  + [Question 29 : Count 1’s in sorted Binary Array.](https://java2blog.com/java-coding-interview-questions/#Question_29_Count_1s_in_sorted_Binary_Array)
  + [Question 30 : Find first repeating element in an array of integers.](https://java2blog.com/java-coding-interview-questions/#Question_30_Find_first_repeating_element_in_an_array_of_integers)
  + [Question 31 : Check if Array Elements are Consecutive.](https://java2blog.com/java-coding-interview-questions/#Question_31_Check_if_Array_Elements_are_Consecutive)
  + [Question 32 : Permutations of array in java.](https://java2blog.com/java-coding-interview-questions/#Question_32_Permutations_of_array_in_java)
  + [Question 33 : Rotate an array by K positions.](https://java2blog.com/java-coding-interview-questions/#Question_33_Rotate_an_array_by_K_positions)
  + [Question 34 : Stock Buy Sell to Maximize Profit.](https://java2blog.com/java-coding-interview-questions/#Question_34_Stock_Buy_Sell_to_Maximize_Profit)
  + [Question 35 : Find maximum difference between two elements such that larger element appears after the smaller number.](https://java2blog.com/java-coding-interview-questions/#Question_35_Find_maximum_difference_between_two_elements_such_that_larger_element_appears_after_the_smaller_number)
  + [Question 36 : Search in a row wise and column wise sorted matrix.](https://java2blog.com/java-coding-interview-questions/#Question_36_Search_in_a_row_wise_and_column_wise_sorted_matrix)
  + [Question 37 : Largest sum contiguous subarray.](https://java2blog.com/java-coding-interview-questions/#Question_37_Largest_sum_contiguous_subarray)
  + [Question 38 : Find the Contiguous Subarray with Sum to a Given Value in an array.](https://java2blog.com/java-coding-interview-questions/#Question_38_Find_the_Contiguous_Subarray_with_Sum_to_a_Given_Value_in_an_array)
  + [Question 39 : Longest Common Prefix in an array of Strings in java.](https://java2blog.com/java-coding-interview-questions/#Question_39_Longest_Common_Prefix_in_an_array_of_Strings_in_java)
  + [Question 40 : Find all subsets of set (power set) in java.](https://java2blog.com/java-coding-interview-questions/#Question_40_Find_all_subsets_of_set_power_set_in_java)
* [Stack](https://java2blog.com/java-coding-interview-questions/#Stack)
  + [Question 41:  Implement a stack using array.](https://java2blog.com/java-coding-interview-questions/#Question_41_Implement_a_stack_using_array)
  + [Question 42: Implement a stack using Linked List.](https://java2blog.com/java-coding-interview-questions/#Question_42_Implement_a_stack_using_Linked_List)
  + [Question 43:  Implement a stack using two queues.](https://java2blog.com/java-coding-interview-questions/#Question_43_Implement_a_stack_using_two_queues)
  + [Question 44 : Sort an stack using another stack](https://java2blog.com/java-coding-interview-questions/#Question_44_Sort_an_stack_using_another_stack)
* [Queue](https://java2blog.com/java-coding-interview-questions/#Queue)
  + [Question 45:  Implement Queue using Array in java.](https://java2blog.com/java-coding-interview-questions/#Question_45_Implement_Queue_using_Array_in_java)
  + [Question 46:  Implement a stack using two queues .](https://java2blog.com/java-coding-interview-questions/#Question_46_Implement_a_stack_using_two_queues)
* [Linked List](https://java2blog.com/java-coding-interview-questions/#Linked_List)
  + [Question 47 : Implement singly linked list in java.](https://java2blog.com/java-coding-interview-questions/#Question_47_Implement_singly_linked_list_in_java)
  + [Question 48: How to reverse linked list in java.](https://java2blog.com/java-coding-interview-questions/#Question_48_How_to_reverse_linked_list_in_java)
  + [Question 49: How to find middle element of linked list.](https://java2blog.com/java-coding-interview-questions/#Question_49_How_to_find_middle_element_of_linked_list)
  + [Question 50 : How to find nth element from end of linked list .](https://java2blog.com/java-coding-interview-questions/#Question_50_How_to_find_nth_element_from_end_of_linked_list)
  + [Question 51 : How to detect a loop in linked list. If linked list has loop, find the start node for the loop.](https://java2blog.com/java-coding-interview-questions/#Question_51_How_to_detect_a_loop_in_linked_list_If_linked_list_has_loop_find_the_start_node_for_the_loop)
  + [Question 52: How to check if linked list is palindrome or not?](https://java2blog.com/java-coding-interview-questions/#Question_52_How_to_check_if_linked_list_is_palindrome_or_not)
  + [Question 53 :  Find intersection of two linked lists?](https://java2blog.com/java-coding-interview-questions/#Question_53_Find_intersection_of_two_linked_lists)
  + [Question 54 :  How to reverse a linked list in pairs?](https://java2blog.com/java-coding-interview-questions/#Question_54_How_to_reverse_a_linked_list_in_pairs)
  + [Question 55 :  Implement Doubly linked list in java?](https://java2blog.com/java-coding-interview-questions/#Question_55_Implement_Doubly_linked_list_in_java)
* [Binary Tree](https://java2blog.com/java-coding-interview-questions/#Binary_Tree)
  + [Question 56 : How can you traverse binary tree?](https://java2blog.com/java-coding-interview-questions/#Question_56_How_can_you_traverse_binary_tree)
  + [Question 57 : Write an algorithm to do level order traversal of binary tree?](https://java2blog.com/java-coding-interview-questions/#Question_57_Write_an_algorithm_to_do_level_order_traversal_of_binary_tree)
  + [Question 58 :  Write an algorithm to do spiral order traversal of binary tree?](https://java2blog.com/java-coding-interview-questions/#Question_58_Write_an_algorithm_to_do_spiral_order_traversal_of_binary_tree)
  + [Question 59 : How can you print leaf nodes of binary tree?](https://java2blog.com/java-coding-interview-questions/#Question_59_How_can_you_print_leaf_nodes_of_binary_tree)
  + [Question 60 : How to count leaf nodes of binary tree.](https://java2blog.com/java-coding-interview-questions/#Question_60_How_to_count_leaf_nodes_of_binary_tree)
  + [Question 61 : How to print all paths from root to leaf in binary tree.](https://java2blog.com/java-coding-interview-questions/#Question_61_How_to_print_all_paths_from_root_to_leaf_in_binary_tree)
  + [Question 62 : How to find level of node in binary tree](https://java2blog.com/java-coding-interview-questions/#Question_62_How_to_find_level_of_node_in_binary_tree)
  + [Question 63 : How to find maximum element in binary tree.](https://java2blog.com/java-coding-interview-questions/#Question_63_How_to_find_maximum_element_in_binary_tree)
  + [Question 64 : How to find lowest common ancestor(LCA) in binary tree.](https://java2blog.com/java-coding-interview-questions/#Question_64_How_to_find_lowest_common_ancestorLCA_in_binary_tree)
  + [Question 65 : How to do boundary traversal of binary tree.](https://java2blog.com/java-coding-interview-questions/#Question_65_How_to_do_boundary_traversal_of_binary_tree)
  + [Question 66 : How to print vertical sum of binary tree?](https://java2blog.com/java-coding-interview-questions/#Question_66_How_to_print_vertical_sum_of_binary_tree)
  + [Question 67 : Count subtrees with Sum equal to target in binary tree?](https://java2blog.com/java-coding-interview-questions/#Question_67_Count_subtrees_with_Sum_equal_to_target_in_binary_tree)
* [Binary Search tree](https://java2blog.com/java-coding-interview-questions/#Binary_Search_tree)
  + [Question 68 : What is binary search tree?](https://java2blog.com/java-coding-interview-questions/#Question_68_What_is_binary_search_tree)
  + [Question 69 : Can you write algorithm to insert a node in binary search tree.](https://java2blog.com/java-coding-interview-questions/#Question_69_Can_you_write_algorithm_to_insert_a_node_in_binary_search_tree)
  + [Question 70 : Can you write algorithm to delete a node in binary search tree.](https://java2blog.com/java-coding-interview-questions/#Question_70_Can_you_write_algorithm_to_delete_a_node_in_binary_search_tree)
  + [Question 71 :  How can you find minimum and maximum elements in binary search tree?](https://java2blog.com/java-coding-interview-questions/#Question_71_How_can_you_find_minimum_and_maximum_elements_in_binary_search_tree)
  + [Question 72 : How to find lowest common ancestor(LCA) in binary search tree.](https://java2blog.com/java-coding-interview-questions/#Question_72_How_to_find_lowest_common_ancestorLCA_in_binary_search_tree)
  + [Question 73 : Find inorder successor in a Binary search Tree](https://java2blog.com/java-coding-interview-questions/#Question_73_Find_inorder_successor_in_a_Binary_search_Tree)
  + [Question 74 : Convert sorted array to balanced BST](https://java2blog.com/java-coding-interview-questions/#Question_74_Convert_sorted_array_to_balanced_BST)
  + [Question 75 : Convert sorted Linked List to balanced BST](https://java2blog.com/java-coding-interview-questions/#Question_75_Convert_sorted_Linked_List_to_balanced_BST)
  + [Question 76 : Check if a binary tree is binary search tree or not in java](https://java2blog.com/java-coding-interview-questions/#Question_76_Check_if_a_binary_tree_is_binary_search_tree_or_not_in_java)
* [Sorting](https://java2blog.com/java-coding-interview-questions/#Sorting)
  + [Question 77 : Write an algorithm to implement bubble sort?](https://java2blog.com/java-coding-interview-questions/#Question_77_Write_an_algorithm_to_implement_bubble_sort)
  + [Question 78 : Write an algorithm to implement insertion sort sort?](https://java2blog.com/java-coding-interview-questions/#Question_78_Write_an_algorithm_to_implement_insertion_sort_sort)
  + [Question 79 : Write an algorithm to implement selection sort sort?](https://java2blog.com/java-coding-interview-questions/#Question_79_Write_an_algorithm_to_implement_selection_sort_sort)
  + [Question 80 : Can you write algorithm for merge sort and also do you know complexity of merge sort?](https://java2blog.com/java-coding-interview-questions/#Question_80_Can_you_write_algorithm_for_merge_sort_and_also_do_you_know_complexity_of_merge_sort)
  + [Question 81 : Do you know how to implement Heap sort?](https://java2blog.com/java-coding-interview-questions/#Question_81_Do_you_know_how_to_implement_Heap_sort)
  + [Question 82 : Implement quick sort in java?](https://java2blog.com/java-coding-interview-questions/#Question_82_Implement_quick_sort_in_java)
  + [Question 83 : Implement shell sort in java?](https://java2blog.com/java-coding-interview-questions/#Question_83_Implement_shell_sort_in_java)
  + [Question 84 : Implement Counting sort in java?](https://java2blog.com/java-coding-interview-questions/#Question_84_Implement_Counting_sort_in_java)
  + [Question 85 : What is binary search? Can you write an algorithm to find an element in sorted array using binary search?](https://java2blog.com/java-coding-interview-questions/#Question_85_What_is_binary_search_Can_you_write_an_algorithm_to_find_an_element_in_sorted_array_using_binary_search)
* [Graph](https://java2blog.com/java-coding-interview-questions/#Graph)
  + [Question 86 : Write algorithm to do depth first search in a graph.](https://java2blog.com/java-coding-interview-questions/#Question_86_Write_algorithm_to_do_depth_first_search_in_a_graph)
  + [Question 87 : Write algorithm to do breadth first search in a graph.](https://java2blog.com/java-coding-interview-questions/#Question_87_Write_algorithm_to_do_breadth_first_search_in_a_graph)
  + [Question 88 : Explain Dijkstra algorithm from source to all other vertices.](https://java2blog.com/java-coding-interview-questions/#Question_88_Explain_Dijkstra_algorithm_from_source_to_all_other_vertices)
  + [Question 89 : Explain Bellman Ford algorithm to find shortest distance](https://java2blog.com/java-coding-interview-questions/#Question_89_Explain_Bellman_Ford_algorithm_to_find_shortest_distance)
  + [Question 90 : Explain Kruskal’s algorithm for finding minimum spanning tree](https://java2blog.com/java-coding-interview-questions/#Question_90_Explain_Kruskal8217s_algorithm_for_finding_minimum_spanning_tree)
* [Dynamic Programming](https://java2blog.com/java-coding-interview-questions/#Dynamic_Programming)
  + [Question 91 : Given two String, find longest common substring.](https://java2blog.com/java-coding-interview-questions/#Question_91_Given_two_String_find_longest_common_substring)
  + [Question 92 : Given two Strings A and B. Find the length of the Longest Common Subsequence (LCS) of the given Strings.](https://java2blog.com/java-coding-interview-questions/#Question_92_Given_two_Strings_A_and_B_Find_the_length_of_the_Longest_Common_Subsequence_LCS_of_the_given_Strings)
  + [Question 93 : Given a matrix, we need to count all paths from top left to bottom right of MxN matrix. You can either move down or right.](https://java2blog.com/java-coding-interview-questions/#Question_93_Given_a_matrix_we_need_to_count_all_paths_from_top_left_to_bottom_right_ofMxN_matrix_You_can_either_move_down_or_right)
  + [Question 94 : Edit Distance Problem in java](https://java2blog.com/java-coding-interview-questions/#Question_94_Edit_Distance_Problem_in_java)
  + [Question 95: Coin change problem in java](https://java2blog.com/java-coding-interview-questions/#Question_95_Coin_change_problem_in_java)
  + [Question 96 : Minimum number of jumps to reach last index](https://java2blog.com/java-coding-interview-questions/#Question_96_Minimum_number_of_jumps_to_reach_last_index)
* [Miscellaneous](https://java2blog.com/java-coding-interview-questions/#Miscellaneous)
  + [Question 97 : What is an algorithm and how to calculate complexity of algorithms.](https://java2blog.com/java-coding-interview-questions/#Question_97_What_is_an_algorithm_and_how_to_calculate_complexity_of_algorithms)
  + [Question 98 : Implement trie data structure in java.](https://java2blog.com/java-coding-interview-questions/#Question_98_Implement_trie_data_structure_in_java)
  + [Question 99 : Count Factorial Trailing Zeroes in java.](https://java2blog.com/java-coding-interview-questions/#Question_99_Count_Factorial_Trailing_Zeroes_in_java)
  + [Question 100 : Largest Rectangular Area in a Histogram.](https://java2blog.com/java-coding-interview-questions/#Question_100_Largest_Rectangular_Area_in_a_Histogram)
  + [Question 101 : Check for balanced parentheses in an expression in java.](https://java2blog.com/java-coding-interview-questions/#Question_101_Check_for_balanced_parentheses_in_an_expression_in_java)
  + [Question 102 : What is Memoization.](https://java2blog.com/java-coding-interview-questions/#Question_102_What_is_Memoization)

If you want to practice and improve [data structure](https://java2blog.com/data-structures-java/) and algorithm programs, this post will be very helpful to you. I will recommend you to try it yourself first and then check the solution.

## String

### Question 1 : How to reverse a String in java? Can you write a program without using any java inbuilt methods?

**Solution:** There are many ways to do it, some of them are:

* Using for loop
* Using recursion
* Using StringBuffer

Please refer to the solution at [reverse a String in java](https://www.java2blog.com/java-program-to-reverse-string/)

### Question 2 : Write a java program to check if two Strings are anagram in java?

**Solution:**Two string are anagrams if they have same characters but in different order. For example: Angel and Angle are anagrams  
There are few ways to check if Strings are anagrams. Some of them are:

1. Using String methods
2. Using array.sort

Check solution at [check if two Strings are anagram in java.](https://www.java2blog.com/check-if-two-strings-are-anagrams-in-java-example-program/)

### Question 3 : Write a program to check if String has all unique characters in java?

**Solution:**Here are some ways to check if String contains all unique characters

* By using HashSet
* Using indexOf and lastIndexOf methods of String
* By Using ascii value of characters.

Please refer to complete solution at [check if String has all unique characters](https://www.java2blog.com/check-if-string-has-all-unique-characters-in-java/).

### Question 4 : How to check if one String is rotation of another String in java?

**Solution: Let’s** say you want to check whether str1 and str2 is rotation of one another or not.

1. Create a new String with str3= str1 + str1
2. Check if str3 [contains](https://www.java2blog.com/java-string-contains-example/) str2 or not.
3. if str3 [contains](https://www.java2blog.com/java-string-contains-example/) str2 then str2 is rotation of str1 else it is not

You can find complete solution at [check if one String is rotation of another in java](https://www.java2blog.com/check-if-one-string-is-rotation-of-another-java/).

### Question 5 : How to find duplicate characters in String in java?

**Solution:**Here is a solution to find duplicate characters in String.

1. Create a [HashMap](https://www.java2blog.com/hashmap-in-java-with-examples/) and character of String will be inserted as key and its count as value.
2. If [Hashamap](https://www.java2blog.com/hashmap-in-java-with-examples/) already contains char,increase its count by 1, else put char in HashMap.
3. If value of Char is more than 1, that means it is duplicate character in that String.

Please refer to solution at[program to find duplicate characters in a String](https://www.java2blog.com/find-duplicate-characters-in-string-java/).

### Question 6 : Find first non repeated character in String in java?

**Solution:**There are may ways to find it.  
Some of them are:

* Using [LinkedHashMap](https://java2blog.com/linkedhashmap-in-java-with-example/)
* Using indexOf and lastIndexOf methods.

Please find complete solution at [find first non repeated character in  a String](https://www.java2blog.com/find-first-non-repeated-character-in/).

### Question 7 : Find all substrings of String in java?

**Solution:** Java program to find all [substrings](https://java2blog.com/find-all-substrings-of-string-in-java/) of a String.  
For example: If input is “abb”  then output should be “a”, “b”,”b”, “ab”, “bb”, “abb”

We will use String class’s subString method to find all subString.  
Please refer to complete solution at [find all subStrings of String.](https://www.java2blog.com/find-all-substrings-of-string-in-java/)

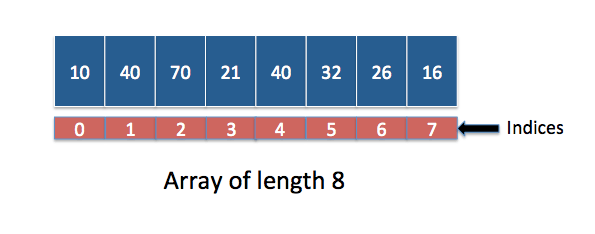
### Question 8 : Find length of String without using any inbuilt method in java?

**Solution:** You can use try catch block for catching StringIndexOutOfBoundException and when this exception aries, you can simply return i(Index at which you will get the exception)  
Please refer to complete solution at [find length of String without inbuilt methods](https://www.java2blog.com/find-length-of-string-without-using/).

### Question 9 : Write a program to print all permutations of String in java?

**Solution:** Take out first character of String and insert into different places of permutations of remaining String recursively. Please find complete solution at [how to find all permutations of String in java](https://www.java2blog.com/find-all-permutations-of-string-in-java/).

## Array



You may be asked lot of java coding interview questions on Array. You can practice following coding questions on Array to ace coding interview.

### Question 10 : Write java Program to Find Smallest and Largest Element in an Array.

You are given an integer array containing 1 to n but one of the number from 1 to n in the array is missing. You need to provide an optimum solution to find the missing number. Number can not be repeated in the arry.  
For example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **int**[] arr1={7,5,6,1,4,2};  Missing numner : 3  **int**[] arr2={5,3,1,2};  Missing numner : 4 |

**Solution**: [Java Program to Find Smallest and Largest Element in an Array](https://java2blog.com/java-program-to-find-smallest-and-largest-number-in-array/)

### Question 11 : Find missing number in the array.

You are given an integer array containing 1 to n but one of the number from 1 to n in the array is missing. You need to provide optimum solution to find the missing number. Number cannot be repeated in the arry.  
For example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **int**[] arr1={7,5,6,1,4,2};  Missing numner : 3  **int**[] arr2={5,3,1,2};  Missing numner : 4 |

**Solution**: [Find missing number in the array.](https://www.java2blog.com/java-program-to-find-missing-number-in-array/)

### Question 12 : Search an element in rotated and sorted array.

You are given an sorted and rotated array as below:

|  |  |
| --- | --- |
| 1  2  3 | **int** arr[]={16,19,21,25,3,5,8,10}; |

If you note that array is sorted and rotated. You need to search an element in above array in o(log n) time complexity.  
**Solution**: [Search element in rotated and sorted array](https://www.java2blog.com/search-element-in-sorted-and-rotated-array-java/)

### Question 13 : Find minimum element in a sorted and rotated array.

You are given an sorted and rotated array as below:

|  |  |
| --- | --- |
| 1  2  3  4 | **int** arr[]={16,19,21,25,3,5,8,10};  Minimum element **in** the **array** : 3 |

If you note that array is sorted and rotated. You need to i an element in above array in o(log n) time complexity.  
**Solution**: [Find minimum element in a sorted and rotated array](https://java2blog.com/find-minimum-element-in-sorted-and-rotated-array-java/)

### Question 14: Find second largest number in an array

You are given an sorted and rotated array as below:

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4 | **int**[] arr1={7,5,6,1,4,2};  Second largest element **in** the **array** : 6 |

**Solution :**[java program to find second largest number in an array](https://www.java2blog.com/find-second-largest-number-in-array-java/).

### Question 15 : Find the number occurring odd number of times in an array

You are given a array of integer. All numbers occur even number of times except one. You need to find the number which occurs odd number of time. You need to solve it with o(n) time complexity and o(1) space complexity.  
For example:

|  |  |
| --- | --- |
| 1  2  3  4 | **int** **array**[] = **new** **int**[]{20, 40, 50, 40, 50, 20, 30, 30, 50, 20, 40, 40, 20};  Number which occurs odd number of times **is** : 50 |

**Solution :**[java program to find number occurring odd number of times in an array](https://www.java2blog.com/find-number-occurring-odd-number-of-times-in-array/).

### Question 16 : Find minimum number of platforms required for railway station

You are given arrival and departure time of trains reaching to a particular station. You need to find minimum number of platforms required to accommodate the trains at any point of time.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | arrival[] = {1:00, 1:40, 1:50, 2:00, 2:15, 4:00}  departure[] = {1:10, 3:00, 2:20, 2:30, 3:15, 6:00}  No. of platforms required **in** above scenario = 4 |

Please note that arrival time is in chronological order.

**Solution :**[Find minimum number of platforms required for railway station](https://www.java2blog.com/minimum-number-of-platforms-required-for-railway-station/).

### Question 17 : Find a Pair Whose Sum is Closest to zero in Array

Given array of +ve and -ve integers ,we need to find a pair whose sum is closed to Zero in Array.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4 | **array**[]={1,3,-5,7,8,20,-40,6};  The pair whose sum **is** closest **to** zero :  -5 **and** 6 |

**Solution :**[Find a Pair Whose Sum is Closest to zero in Array in java](https://www.java2blog.com/find-pair-whose-sum-is-closest-to-zero-in-array/).

### Question 18 : Given a sorted array and a number x, find the pair in array whose sum is closest to x

Given a sorted array, we need to find a pair whose sum is closed to number X in Array.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4 | **array**[]={-40,-5,1,3,6,7,8,20};  The pair whose sum **is** closest **to** 5 :  1 **and** 3 |

**Solution :**[Find a Pair Whose Sum is Closest to X in Array in java](https://www.java2blog.com/given-sorted-array-number-x-find-pair-closest-to-x-array/).

### Question 19 : Find all pairs of elements from an array whose sum is equal to given number

Given a  array,we need to find all pairs whose sum is equal to number X.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4 | **array**[]={ -40, -5, 1, 3, 6, 7, 8, 20 };  Pair of elements whose sum **is** equal **to** 15 :  7, 8 **and** -5, 20 |

**Solution :**[Find all pairs of elements from an array whose sum is equal to given number](https://www.java2blog.com/find-all-pairs-elements-from-array-sum-equals-given-number-java/).

### Question 20: Given an array of 0’s and 1’s in random order, you need to separate 0’s and 1’s in an array.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | arr[] = {0,1,0,0,1,1,1,0,1}  **Array** after separating 0 **and** 1 numbers :  {0,0,0,0,1,1,1,1,1} |

**Solution :**[Separate 0s and 1s in array](https://java2blog.com/separate-0s-and-1s-in-array/).

### Question 21 : Separate odd and even numbers in an array

Given an array of integers, you need to segregate odd and even numbers in an array.  
Please note: Order of elements can be changed.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | arr[] = {12, 17, 70, 15, 22, 65, 21, 90}  **Array** after separating odd **and** even numbers :  {12, 90, 70, 22, 15, 65, 21, 17} |

**Solution :**[Separate 0s and 1s in array](https://java2blog.com/separate-odd-and-even-numbers-in-array/).

### Question 22 : Given an array containing zeroes, ones and twos only. Write a function to sort the given array in O(n) time complexity.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Input :  [1, 2, 2, 0, 0, 1, 2, 2, 1]    Output :  [0, 0, 1, 1, 1, 2, 2, 2, 2] |

**Solution :**[Sort an array of 0s, 1s and 2s](https://java2blog.com/sort-array-of-0s-1s-and-2s/).

### Question 23 : Find local minima in array

A local minima is less than its neighbours

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | Input :    **int** [] arr = {10, 5, 3, 6, 13, 16, 7};  Output: 2    **int** []arr = {11,12,13,14};  Output: 11    **int** []arr = {10};  Output: 10    **int** []arr = {8,6};  Output: 6 |

### Question 24 : Sliding window maximum in java

Given an Array of integers and an Integer k, Find the maximum element of from all the contiguous subarrays of size K.

**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Input :  Input : **int**[] arr = {2,6,-1,2,4,1,-6,5}  **int** k = 3  output : 6,6,4,4,4,5 |

**Solution :**[Find the local minima in array](https://java2blog.com/sliding-window-maximum-java/).

### Question 25 : Count number of occurrences (or frequency) of each element in a sorted array

Given a Sorted Array of integers containing duplicates. Find the frequency of every unique element present in the array.  
Frequency is defined as the number of occurrence of any element in the array.

**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | Input :  Input:  **int**[] arr = {1, 1, 1, 3, 3, 4, 5, 5, 6, 6};  Output:  Frequency of 1 **is** : 3  Frequency of 3 **is** : 2  Frequency of 4 **is** : 1  Frequency of 5 **is** : 2  Frequency of 6 **is** : 2 |

**Solution :**[Count number of occurrences (or frequency) of each element in a sorted array](https://java2blog.com/count-occurences-frequency-each-element-sorted-array/).

### Question 26 : Find subarrays with given sum in an array.

Given an Array of non negative Integers and a number. You need to print all the starting and ending indices of Subarrays having their sum equal to the given integer.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | Input :  Input-**int**[] arr = {2, 3, 6, 4, 9, 0, 11};  **int** num = 9  Output-  starting index : 1, Ending index : 2  starting index : 5, Ending index : 5  starting index : 5, Ending index : 6 |

**Solution :**[Find subarrays with given sum in an array](https://java2blog.com/find-subarrays-given-sum-array/).

### Question 27 : Find peak element in the array.

Peak Element is the element of the array which is GREATER THAN / EQUAL TO its neighbours, that is, for an element at i th index, the neighbour elements at index i-1 & i+1 must be greater than equal to element at i th position.

**Solution :**[Find peak element in the array](https://java2blog.com/find-subarrays-given-sum-array/).

### Question 28 : Find leaders in an array.

We need to print all the leaders present in the array. Element is the leader if it is greater than right side of elements.

|  |  |
| --- | --- |
| 1  2  3  4 | arr[]={14, 12, 70, 15, 99, 65, 21, 90}  Here 99 **and** 90 are leader elements |

**For example:**

**Solution :**[Find leaders in an array](https://java2blog.com/find-leaders-in-array/).

### Question 29 : Count 1’s in sorted Binary Array.

Print number of 1’s in a given sorted Binary Array.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Input :  **int**[] arr = {0,0,0,1,1,1,1};  output : 4  **int**[] arr = {0,0,1};  output : 1 |

**Solution :**[Count 1’s in sorted Binary Array](https://java2blog.com/count-1s-sorted-binary-array/).

### Question 30 : Find first repeating element in an array of integers.

Find the first repeating element in array of integers.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Input :  Input: **array**[] = {10, 7, 8, 1, 8, 7, 6}  Output: 7 [7 **is** the first element actually repeats] |

**Solution :**[Find first repeating element in an array of integers](https://java2blog.com/find-first-repeating-element-array-integers/).

### Question 31 : Check if Array Elements are Consecutive.

Given an array, we need to check if array contains consecutive elements.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | Input: **array**[] = {5, 3, 4, 1, 2}  Output: **true**  **As** **array** contains consecutive elements from 1 **to** 5  Input: **array**[] = {47, 43, 45, 44, 46}  Output: **true**  **As** **array** contains consecutive elements from 43 **to** 47  Input: **array**[] = {6, 7, 5, 6}  Output: **false**  **As** **array** does **not** contain consecutive elements. |

**Solution :**[Check if Array Elements are Consecutive](https://java2blog.com/check-array-elements-consecutive/).

### Question 32 : Permutations of array in java.

Given array of distinct integers, print all permutations of the array.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **array** : [10, 20, 30]    Permuations are :    [10, 20, 30]  [10, 30, 20]  [20, 10, 30]  [20, 30, 10]  [30, 10, 20]  [30, 20, 10] |

**Solution :**[Permutations of array in java](https://java2blog.com/permutations-array-java/).

### Question 33 : Rotate an array by K positions.

**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5 | N=6 **and** k=2  **If** Arr[] = {1, 2, 3, 4, 5, 6} **and** k=2  **then** rotated **array** will be  {5, 6, 1, 2,  3,  4} |

**Solution :**[Rotate an array by K positions](https://java2blog.com/rotate-array-by-k-positions/).

### Question 34 : Stock Buy Sell to Maximize Profit.

Given an array of integers representing stock price on single day, find max profit that can be earned by 1 transaction.  
So you need to find pair (buyDay,sellDay) where buyDay < = sellDay and it should maximise the profit.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5 | **int** arr[]={14, 12, 70, 15, 99, 65, 21, 90};  Max profit can be gain by buying at 1th day(0 based indexing) **and** sell at 4th day.  Max profit = 99-12 =87 |

**Solution :**[Stock Buy Sell to Maximize Profit](https://java2blog.com/stock-buy-sell-to-maximize-profit/).

### Question 35 : Find maximum difference between two elements such that larger element appears after the smaller number.

Given array of integers, find Maximum difference between two elements such that larger element appears after the smaller number  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4 | **int** arr[]={14, 12, 70, 15, 95, 65, 22, 30};  Max Difference =95-12 = 83 |

**Solution :**[Maximum difference between two elements such that larger element appears after the smaller number](https://java2blog.com/maximum-difference-between-two-elements-in-array/).

### Question 36 : Search in a row wise and column wise sorted matrix.

Given row wise and column wise sorted matrix ,we need to search element with minimum time complexity.

**Solution :**[Search in a row wise and column wise sorted matrix](https://java2blog.com/search-in-row-wise-and-column-wise-sorted-matrix/).

### Question 37 : Largest sum contiguous subarray.

Largest sum contiguous subarray is the task of finding the contiguous subarray within a one-dimensional array of numbers which has the largest sum.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3 | **for** the sequence of values −2, 1, −3, 4, −1, 2, 1, −5, 4; the contiguous subarray with the largest sum **is** 4, −1, 2, 1, with sum 6 |

**Solution :**[Largest sum contiguous subarray](https://java2blog.com/largest-sum-contiguous-subarray/).

### Question 38 : Find the Contiguous Subarray with Sum to a Given Value in an array.

Given an array of positive integer and given value X, find Contiguous sub array whose sum is equal to X.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | arr[]={14, 12, 70, 15, 99, 65, 21, 90};  X =97.  Sum found between index 1 **to** 3  Elements are 12, 17 **and** 15 |

**Solution :**[Find the Contiguous Subarray with Sum to a Given Value in an array](https://java2blog.com/find-contiguous-subarray-with-sum-to-given-value-array/).

### Question 39 : Longest Common Prefix in an array of Strings in java.

Given an array of positive integer and given value X, find Contiguous sub array whose sum is equal to X.  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4 | **String**[] strArr={"java2blog","javaworld","javabean","javatemp"};  So Longest common prefix **in** above **String** **array** will be “java” **as** all above **string** starts with “java”. |

**Solution :**[Longest Common Prefix in an array of Strings in java](https://java2blog.com/longest-common-prefix-in-array-of-strings-java/).

### Question 40 : Find all subsets of set (power set) in java.

Given a set of distinct integers, arr, return all possible subsets (the power set).  
**For example :**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | Input: nums = [1,2,3]  Output:  [  [3],  [1],  [2],  [1,2,3],  [1,3],  [2,3],  [1,2],  []  ] |

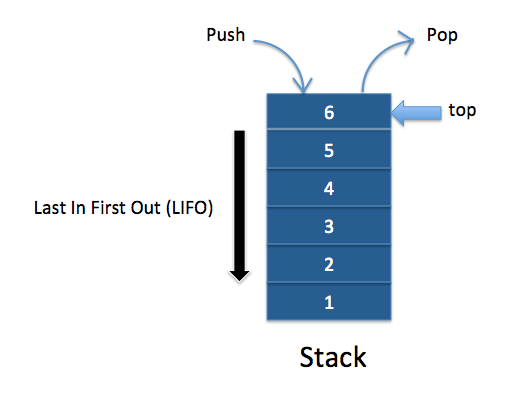
23.9M

180

Learn Java Programming with Beginners Tutorial

**Solution :**[Find all subsets of set in java](https://java2blog.com/longest-common-prefix-in-array-of-strings-java/).

## Stack



### Question 41:  Implement a stack using array.

You need to implement Stack using array. You need to write push and pop methods to demonstrate Stack behavior(Last In First Out).  
**Solution**:[Java Program to implement stack using array.](https://www.java2blog.com/implement-stack-using-array-in-java/)

### Question 42: Implement a stack using Linked List.

You need to implement Stack using Linked List. You need to write push and pop methods to demonstrate Stack behavior(Last In First Out).  
**Solution**: [Java Program to implement stack using Linked List](https://www.java2blog.com/implement-stack-using-linked-list-in-java/)

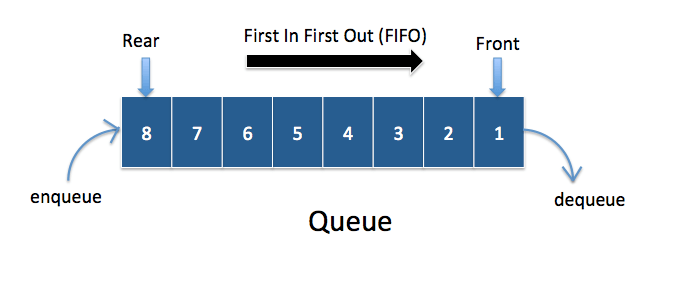
### Question 43:  Implement a stack using two queues.

You need to use two queues to implement stack behavior.You need to write push and pop methods to demonstrate Stack behavior(Last In First Out).  
**Solution**:[Java Program to implement stack using two queues](https://www.java2blog.com/implement-stack-using-two-queues-in-java/)

### Question 44 : Sort an stack using another stack

You need to sort an stack using another stack. You can use push and pop operation of stack to do so,  
**Solution**: [Sort a stack using another stack.](https://www.java2blog.com/sort-stack-using-another-stack/)

## Queue



### Question 45:  Implement Queue using Array in java.

You need to use array to implement queue.  
**Solution**:[Implement Queue using Array in java](https://java2blog.com/implement-queue-using-array-in-java/)

### Question 46:  Implement a stack using two queues .

You need to use Linked list to implement queue.  
**Solution**:[Java Program to implement queue using linked list](https://java2blog.com/implement-queue-using-linked-list-in-java/)

## Linked List

### Question 47 : Implement singly linked list in java.

You need to implement singly linked list data structures.You need to write simple program to demonstrate insert , delete operations.



**Solution**: [Java program to implement singly linked list in java.](https://www.java2blog.com/implement-singly-linked-list-in-java/)

### Question 48: How to reverse linked list in java.

You need to write iterative and recursive solution to reverse linked list.  
**Solution**[: Java program to reverse linked list in java.](https://www.java2blog.com/how-to-reverse-linked-list-in-java/)

### Question 49: How to find middle element of linked list.

You need to write java program to find middle element of linked list in most optimize way.



**Solution**: [Java program to find middle element of linked list](https://www.java2blog.com/find-middle-element-of-linkedlist-in/).

### Question 50 : How to find nth element from end of linked list .

You need to write java program to find nth  element of linked list in most optimize way.  
In question 6, Node 7 is 3rd from last of linked list.  
**Solution**:[How to find nth element from end of linked list.](https://www.java2blog.com/find-nth-element-from-end-of-linked-list/)

### Question 51 : How to detect a loop in linked list. If linked list has loop, find the start node for the loop.

You need to write a java program to detect whether any loop exists in linked list and if loop exists , you need to find start node for the linked list.  
**Solution**: [How to detect loop in linked list](https://www.java2blog.com/how-to-detect-loop-in-linkedlist-in/).  
[How to find start node of loop in linked list.](https://www.java2blog.com/find-start-node-of-loop-in-linkedlist/)

### Question 52: How to check if linked list is palindrome or not?

A palindrome is a word, phrase, number, or other sequence of symbols or elements that reads the same forward or reversed. For example: 12121 is palindrome as it reads same forward or reversed. madam is also a palindrome . So we need write java programs to check if linked list is palindrome or not.  
**Solution**: [Java program to check if linked list is palindrome.](https://www.java2blog.com/how-to-check-if-linked-list-is/)

### Question 53 :  Find intersection of two linked lists?

Given two [singly linked lists](https://java2blog.com/implement-singly-linked-list-in-java/), find if two linked lists intersect. If they intersect, find intersection point.

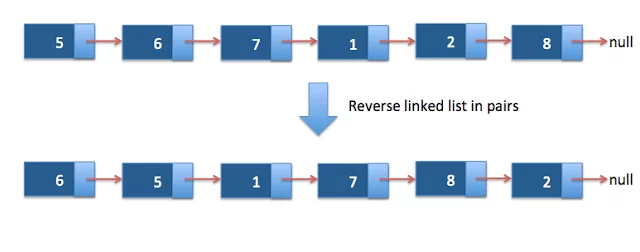
A picture containing text, jack, electronics

Description automatically generated

Solution  : [Intersection of two linked list](https://java2blog.com/intersection-two-linked-lists/)

### Question 54 :  How to reverse a linked list in pairs?

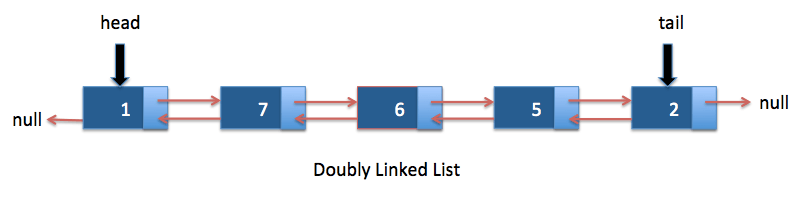
You need to write a java program to reverse linked list in pairs.



Solution  : [Java program to reverse linked list in pair.](https://www.java2blog.com/java-program-to-reverse-linked-list-in/)

### Question 55 :  Implement Doubly linked list in java?

You need to write a java program to implement doubly linked list in java.



Solution  : [Doubly Linked List in java](https://java2blog.com/doubly-linked-list-java/)

## Binary Tree

### Question 56 : How can you traverse binary tree?

There are three ways to traverse binary tree.

* [PreOrder](https://www.java2blog.com/binary-tree-preorder-traversal-in-java/)
* [InOrder](https://www.java2blog.com/binary-tree-inorder-traversal-in-java/)
* [PostOrder](https://www.java2blog.com/binary-tree-postorder-traversal-in-java/).

### Question 57 : Write an algorithm to do level order traversal of binary tree?

You need to write java program to do level order traversal of binary tree. You can use queue data structure to do level order traversal.

Diagram

Description automatically generated

**Solution**: [Level order traversal of binary tree.](https://www.java2blog.com/binary-tree-level-order-traversal-in/)

### Question 58 :  Write an algorithm to do spiral order traversal of binary tree?

You need to write java program to do spiral level order traversal of binary tree

Diagram

Description automatically generated

**Solution**: [Sprial order or zigzag traversal of binary tree.](https://www.java2blog.com/spiralzigzag-level-order-traversal-of/)

### Question 59 : How can you print leaf nodes of binary tree?

You need to write java program to print all leaf nodes of binary tree.

Diagram

Description automatically generated

Leaf nodes for above binary tree will be 5 , 30 , 55 ,70  
**Solution**:[Print leaf nodes of binary tree.](https://www.java2blog.com/how-to-print-leaf-nodes-of-binary-tree/)

### Question 60 : How to count leaf nodes of binary tree.

You need to write java program to count leaf nodes of binary tree.  
Count of Leaf nodes for binary tree used in Question 15 are **5**.  
**Solution**: [Count leaf nodes of binary tree.](https://www.java2blog.com/program-to-count-leaf-nodes-in-binary/)

### Question 61 : How to print all paths from root to leaf in binary tree.

You need to write a program to print all paths from root to leaf.

Diagram

Description automatically generated

**Solution**: [Print all paths from root to leaf in binary tree.](https://www.java2blog.com/print-all-paths-from-root-to-leaf-in/)

### Question 62 : How to find level of node in binary tree

Given a node, you need to find level of a node. For example : Level of node will 3 for node 70 used in Question 14.  
**Solution**: [Find level of node in binary tree.](https://www.java2blog.com/get-level-of-node-in-binary-tree-in-java/)

### Question 63 : How to find maximum element in binary tree.

You need to write a java program to find maximum element in binary tree.  
**Solution**: [Find maximum element in binary tree.](https://java2blog.com/find-maximum-element-binary-tree-java/)

### Question 64 : How to find lowest common ancestor(LCA) in binary tree.

You need to write a program to find LCA in binary tree.

Diagram

Description automatically generated

**Solution**: [Program to find LCA in binary tree.](https://www.java2blog.com/lowest-common-ancestor-of-binary-tree/)

### Question 65 : How to do boundary traversal of binary tree.

Write a java program to do boundary traversal of binary tree as shown in below image.

Diagram

Description automatically generated

Solution : [Boundary traversal of binary tree.](https://www.java2blog.com/boundary-traversal-of-binary-tree-in/)

### Question 66 : How to print vertical sum of binary tree?

You need to find sum of nodes which lies in same column.

A picture containing chart

Description automatically generated

Solution : [How to print vertical sum of binary tree](https://www.java2blog.com/vertical-sum-of-binary-tree-in-java/).

### Question 67 : Count subtrees with Sum equal to target in binary tree?

Given a [Binary tree](https://java2blog.com/binary-tree-in-java/) and an integer. You need to find the number of subtrees having the sum of all of its nodes equal to given Integer, that is, Target sum.

Solution : [Count subtrees with Sum equal to target in binary tree](https://www.java2blog.com/vertical-sum-of-binary-tree-in-java/).

## Binary Search tree

### Question 68 : What is binary search tree?

Binary search tree is a special type of [binary tree](https://www.java2blog.com/binary-tree-in-java/) which have following properties.

* Nodes which are smaller than root will be in left subtree.
* Nodes which are greater than root will be right subtree.
* It should not have duplicate nodes
* Both left and right subtree also should be binary search tree.

### Question 69 : Can you write algorithm to insert a node in binary search tree.

**Solution**: [Insert node in binary search tree](https://java2blog.com/binary-search-tree-in-java/)

### Question 70 : Can you write algorithm to delete a node in binary search tree.

**Solution**: [Delete node in binary search tree](https://java2blog.com/how-to-delete-node-from-binary-search-tree-java/)

### Question 71 :  How can you find minimum and maximum elements in binary search tree?

**Solution**: Leftmost and rightmost nodes of binary search tree are minimum and maximum nodes respectively

[Minimum and maximum elements in binary search tree](https://java2blog.com/find-minimum-and-maximum-elements-binary-search-tree-java/).

### Question 72 : How to find lowest common ancestor(LCA) in binary search tree.

You need to write a program to find LCA in binary search tree.

Diagram

Description automatically generated

**Solution**: [Program to find LCA in binary search tree.](https://www.java2blog.com/lowest-common-ancestor-of-binary-search/)

### Question 73 : Find inorder successor in a Binary search Tree

You need to write a program to find inorder successor in a Binary search tree.

**Solution**: [Inorder Successor in a Binary Search Tree](https://java2blog.com/inorder-successor-binary-search-tree/)

### Question 74 : Convert sorted array to balanced BST

**Solution**: [Convert sorted sorted array to balanced BST](https://java2blog.com/convert-sorted-array-to-balanced-binary-search-tree/)

### Question 75 : Convert sorted Linked List to balanced BST

**Solution**: [Convert sorted Linked List to balanced BST](https://java2blog.com/convert-sorted-linkedlist-to-balanced-bst/)

### Question 76 : Check if a binary tree is binary search tree or not in java

**Solution**: [Check if a binary tree is binary search tree or not in java](https://java2blog.com/check-if-binary-tree-is-binary-search-tree-java/)

## Sorting

### Question 77 : Write an algorithm to implement bubble sort?

**Solution**: [Bubble sort in java](https://java2blog.com/implement-bubble-sort-in-java/)

### Question 78 : Write an algorithm to implement insertion sort sort?

**Solution**: [Insertion sort in java](https://java2blog.com/implement-insertion-sort-in-java/)

### Question 79 : Write an algorithm to implement selection sort sort?

**Solution**: [Selection sort in java](https://java2blog.com/selection-sort-in-java/)

### Question 80 : Can you write algorithm for merge sort and also do you know complexity of merge sort?

**Solution**: [Merge sort in java](https://java2blog.com/implement-merge-sort-in-java/)

### Question 81 : Do you know how to implement Heap sort?

**Solution**: [implement Heap sort in java](https://java2blog.com/heap-sort-in-java/)

### Question 82 : Implement quick sort in java?

**Solution**: [implement Quick sort in java](https://java2blog.com/quick-sort-in-java/)

### Question 83 : Implement shell sort in java?

**Solution**: [implement Shell sort in java](https://java2blog.com/shell-sort-in-java/)

### Question 84 : Implement Counting sort in java?

**Solution**: [implement Counting sort in java](https://java2blog.com/counting-sort-in-java/)

### Question 85 : What is binary search? Can you write an algorithm to find an element in sorted array using binary search?

**Solution**:[Binary search algorithm in java](https://java2blog.com/binary-search-java/)

## Graph

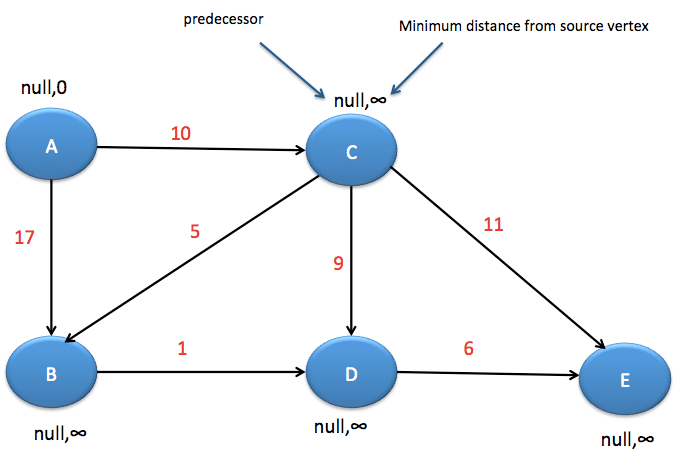
### Question 86 : Write algorithm to do depth first search in a graph.

**Solution**: [Depth first search in java](https://www.java2blog.com/depth-first-search-in-java/)

### Question 87 : Write algorithm to do breadth first search in a graph.

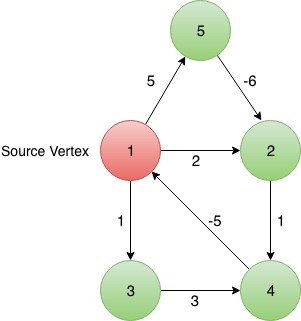
Solution : [breadth first search in java](https://www.java2blog.com/breadth-first-search-in-java/)

### Question 88 : Explain Dijkstra algorithm from source to all other vertices.



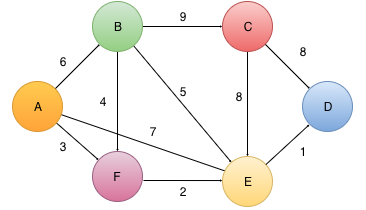
Solution : [Dijkstra’s algorithm in java](https://java2blog.com/dijkstra-java/)

### Question 89 : Explain Bellman Ford algorithm to find shortest distance



Solution : [Bellman ford algorithm in java](https://java2blog.com/bellman-ford-algorithm-java/)

### Question 90 : Explain Kruskal’s algorithm for finding minimum spanning tree



Solution : [Kruskal’s algorithm](https://java2blog.com/kruskals-algorithm-minimum-spanning-tree/)

## Dynamic Programming

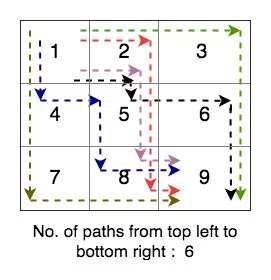
### Question 91 : Given two String, find longest common substring.

Solution: [Longest common substring in java](https://java2blog.com/longest-common-substring-java/).

### Question 92 : Given two Strings A and B. Find the length of the Longest Common Subsequence (LCS) of the given Strings.

Solution: [Longest common subsequence in java](https://java2blog.com/longest-common-subsequence-java/)

### Question 93 : Given a matrix, we need to count all paths from top left to bottom right of MxN matrix. You can either move down or right.



Solution: [Count all paths in matrix](https://java2blog.com/count-paths-top-left-bottom-right-mxn-matrix/)

Given two strings string1 and string2, String1 is to be converted into String2 with the given operations available in the minimum number of steps. Using any one of the given operations contributes to the increment of steps by one.

Allowed Operations are :  
(i) **Remove** : This operation allows the Removal any one character from String.  
(ii) **Insert** : This operation allows the Insertion of one character at any spot in the String.  
(iii) **Replace** : This operation allows the replacement of any one character in the string with  
any other character.

Solution: [Edit distance problem in java](https://java2blog.com/edit-distance-problem/).

Given an Amount to be paid and the currencies to pay with. There is infinite supply of every currency using combination of which, the given amount is to be paid. Print the number of ways by which the amount can be paid.

Solution: [Coin change problem in java](https://java2blog.com/coin-change-problem-java/)

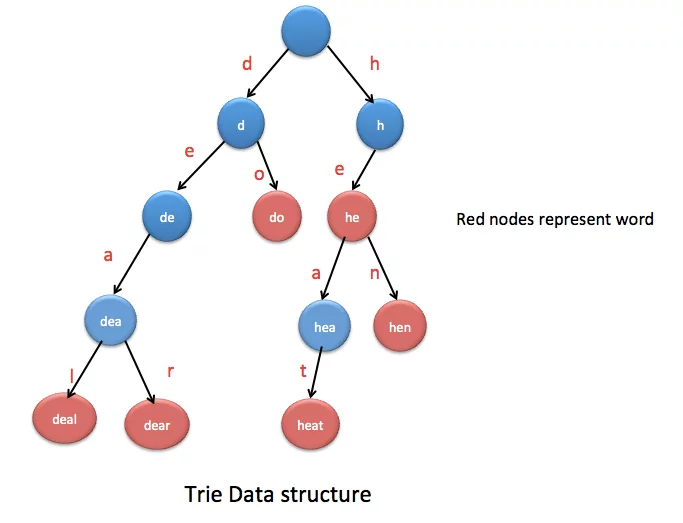
Solution: [Minimum number of jumps to reach last index](https://java2blog.com/minimum-number-jumps-reach-last-index/).

## Miscellaneous

### Question 97 : What is an algorithm and how to calculate complexity of algorithms.

Solution : [How to calculate Complexity of algorithm](https://www.java2blog.com/introduction-to-complexity-of-algorithm/)

### Question 98 : Implement trie data structure in java.

  
Solution : [Implement trie data structure in java.](https://java2blog.com/trie-data-structure-in-java/)

### Question 99 : Count Factorial Trailing Zeroes in java.

Solution : [Count Factorial Trailing Zeroes in java](https://java2blog.com/count-factorial-trailing-zeroes-java/)

### Question 100 : Largest Rectangular Area in a Histogram.

Solution : [Count Largest Rectangular Area in a Histogram](https://java2blog.com/largest-rectangular-area-histogram/)

# Core java interview questions and answers

Core java interview questions play a vital role in [java/j2EE interviews](https://java2blog.com/java-interview-questions/). Whether you are fresher or experienced, you are going to face core java interview questions. You can also go through [top java interview programs](https://www.java2blog.com/2015/08/java-interview-programs.html) for practicing java programs.

Here I am providing some important core java interview questions with answers.

Table of Contents [[hide](https://java2blog.com/core-java-interview-questions-and-answers/)]

* [1. What do you mean by Platform independence of java?](https://java2blog.com/core-java-interview-questions-and-answers/#1_What_do_you_mean_by_Platform_independence_of_java)
* [2. What is difference between JVM, JRE and JDK ?](https://java2blog.com/core-java-interview-questions-and-answers/#2_What_is_difference_between_JVM_JRE_and_JDK)
* [3. What are memory areas allocated in JVM?](https://java2blog.com/core-java-interview-questions-and-answers/#3_What_are_memory_areas_allocated_in_JVM)
* [4. What are some core concepts of OOPS in Java ?](https://java2blog.com/core-java-interview-questions-and-answers/#4_What_are_some_core_concepts_of_OOPS_in_Java)
* [5. What is Abstraction?](https://java2blog.com/core-java-interview-questions-and-answers/#5_What_is_Abstraction)
* [6. What is encapsulation?](https://java2blog.com/core-java-interview-questions-and-answers/#6_What_is_encapsulation)
* [7. What is Polymorphism in java?](https://java2blog.com/core-java-interview-questions-and-answers/#7_What_is_Polymorphism_in_java)
* [8. What is inheritance in java?](https://java2blog.com/core-java-interview-questions-and-answers/#8_What_is_inheritance_in_java)
* [9. What is constructor in java?](https://java2blog.com/core-java-interview-questions-and-answers/#9_What_is_constructor_in_java)
* [10. Can we declare constructor as final?](https://java2blog.com/core-java-interview-questions-and-answers/#10_Can_we_declare_constructor_as_final)
* [11. What is immutable object in java?](https://java2blog.com/core-java-interview-questions-and-answers/#11_What_is_immutable_object_in_java)
* [12. Why String is declared final or immutable in java?](https://java2blog.com/core-java-interview-questions-and-answers/#12_Why_String_is_declared_final_or_immutable_in_java)
* [13. What are access modifier available in java?](https://java2blog.com/core-java-interview-questions-and-answers/#13_What_are_access_modifier_available_in_java)
* [14. What is difference between Abstract class and interface?](https://java2blog.com/core-java-interview-questions-and-answers/#14_What_is_difference_between_Abstract_class_and_interface)
* [15. Can one interface implement another interface in java?](https://java2blog.com/core-java-interview-questions-and-answers/#15_Can_one_interface_implement_another_interface_in_java)
* [16. What is marker interface?](https://java2blog.com/core-java-interview-questions-and-answers/#16_What_is_marker_interface)
* [17. What is method overloading and method overriding in java?](https://java2blog.com/core-java-interview-questions-and-answers/#17_What_is_method_overloading_and_method_overriding_in_java)
* [18. Can you override static methods in Java?](https://java2blog.com/core-java-interview-questions-and-answers/#18_Can_you_override_static_methods_in_Java)
* [19. Can you override private methods in Java?](https://java2blog.com/core-java-interview-questions-and-answers/#19_Can_you_override_private_methods_in_Java)
* [20. Difference between path and classpath in java?](https://java2blog.com/core-java-interview-questions-and-answers/#20_Difference_between_path_and_classpath_in_java)
* [21. What is difference between StringBuffer and StringBuilder in java?](https://java2blog.com/core-java-interview-questions-and-answers/#21_What_is_difference_between_StringBuffer_and_StringBuilder_in_java)
* [22. What are methods you should override when you put an object as key in HashMap?](https://java2blog.com/core-java-interview-questions-and-answers/#22_What_are_methods_you_should_override_when_you_put_an_object_as_key_in_HashMap)
* [23. Can you explain internal working of HashMap in java?](https://java2blog.com/core-java-interview-questions-and-answers/#23_Can_you_explain_internal_working_of_HashMap_in_java)
* [24. Why java uses another hash function internally to calculate hash value apart from hashcode method which you have implemented?](https://java2blog.com/core-java-interview-questions-and-answers/#24_Why_java_uses_another_hash_function_internally_to_calculate_hash_value_apart_from_hashcode_method_which_you_have_implemented)
* [25. What if you don’t override hashcode method while putting custom objects as key in HashMap?](https://java2blog.com/core-java-interview-questions-and-answers/#25_What_if_you_don8217t_override_hashcode_method_while_putting_custom_objects_as_key_in_HashMap)
* [26. Can you explain internal working of HashSet in java?](https://java2blog.com/core-java-interview-questions-and-answers/#26_Can_you_explain_internal_working_of_HashSet_in_java)
* [27. What are differences between HashMap and HashSet in java?](https://java2blog.com/core-java-interview-questions-and-answers/#27_What_are_differences_between_HashMap_and_HashSet_in_java)
* [28. Can you explain internal working of ConcurrentHashMap in java?](https://java2blog.com/core-java-interview-questions-and-answers/#28_Can_you_explain_internal_working_of_ConcurrentHashMap_in_java)
* [29. Do we have lock while getting value from ConcurrentHashMap?](https://java2blog.com/core-java-interview-questions-and-answers/#29_Do_we_have_lock_while_getting_value_from_ConcurrentHashMap)
* [30. How do you sort Collection of custom objects in java?](https://java2blog.com/core-java-interview-questions-and-answers/#30_How_do_you_sort_Collection_of_custom_objects_in_java)
* [31. What are differences between ArrayList and LinkedList in java?](https://java2blog.com/core-java-interview-questions-and-answers/#31_What_are_differences_between_ArrayList_and_LinkedList_in_java)
* [32. What is Enum in java?](https://java2blog.com/core-java-interview-questions-and-answers/#32_What_is_Enum_in_java)
* [33. How do you create custom exception in java?](https://java2blog.com/core-java-interview-questions-and-answers/#33_How_do_you_create_custom_exception_in_java)
* [34.What is difference between Checked Exception and Unchecked Exception?](https://java2blog.com/core-java-interview-questions-and-answers/#34What_is_difference_between_Checked_Exception_and_Unchecked_Exception)
* [35. Can we have try without catch block in java ?](https://java2blog.com/core-java-interview-questions-and-answers/#35Can_we_have_try_without_catch_block_in_java)
* [36. What are ways to create a thread in java ?](https://java2blog.com/core-java-interview-questions-and-answers/#36_What_are_ways_to_create_a_thread_in_java)
* [37. What are differences between Sleep and wait in java?](https://java2blog.com/core-java-interview-questions-and-answers/#37_What_are_differences_between_Sleep_and_wait_in_java)
* [38. Define states of thread in java?](https://java2blog.com/core-java-interview-questions-and-answers/#38_Define_states_of_thread_in_java)
* [39. Can we call run method directly to start a thread?](https://java2blog.com/core-java-interview-questions-and-answers/#39_Can_we_call_run_method_directly_to_start_a_thread)
* [40. Can we start a thread twice in java?](https://java2blog.com/core-java-interview-questions-and-answers/#40_Can_we_start_a_thread_twice_in_java)
* [41. What is CountDownLatch in java?](https://java2blog.com/core-java-interview-questions-and-answers/#41_What_is_CountDownLatch_in_java)
* [42. What is difference between CountDownLatch and CyclicBarrier?](https://java2blog.com/core-java-interview-questions-and-answers/#42_What_is_difference_between_CountDownLatch_and_CyclicBarrier)
* [43. Why wait, notify and nofiyAll method belong to object class ?](https://java2blog.com/core-java-interview-questions-and-answers/#43_Why_wait_notify_and_nofiyAll_method_belong_to_object_class)
* [44. Can you call wait, notify and notifyAll from non synchronized context?](https://java2blog.com/core-java-interview-questions-and-answers/#44_Can_you_call_wait_notify_and_notifyAll_from_non_synchronized_context)
* [45. What is the difference between creating String as new() and literal?](https://java2blog.com/core-java-interview-questions-and-answers/#45_What_is_the_difference_between_creating_String_as_new_and_literal)
* [46. What is Covariant return type in java?](https://java2blog.com/core-java-interview-questions-and-answers/#46_What_is_Covariant_return_type_in_java)
* [47. What is garbage Collection?](https://java2blog.com/core-java-interview-questions-and-answers/#47_What_is_garbage_Collection)
* [48. What is System.gc()?](https://java2blog.com/core-java-interview-questions-and-answers/#48_What_is_Systemgc)
* [49. What is use of finalize() method in object class?](https://java2blog.com/core-java-interview-questions-and-answers/#49_What_is_use_of_finalize_method_in_object_class)
* [50.What is difference between final, finally and finalize in Java?](https://java2blog.com/core-java-interview-questions-and-answers/#50What_is_difference_between_final_finally_and_finalize_in_Java)

### 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.  
**For 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/). You can go through [hashcode and equals method in java](https://www.java2blog.com/2014/02/hashcode-and-equals-method-in-java.html) 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/) 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*  } |

114.5K

Europeans have their data shared 376 times a day in advertising sales, according to a report

Whenever we try to put any key value pair in [Hashmap](https://java2blog.com/hashmap-in-java-with-examples/), 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/) 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/) 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/). 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/).

|  |  |
| --- | --- |
| 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/).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/), 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) 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](https://java2blog.com/exception-thread-main-java-lang-nullpointerexception/ “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) 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.

**Question 1:  
What can be output of below code :**

|  |  |
| --- | --- |
| 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 | **package** org.arpit.java2blog;    import java.util.HashSet;    **class** Country {  private **String** name;  Country(**String** name ){  **this**.name = name;  }  public **String** toString() {  **return** name;  }  }    public **class** CollClient {    public static **void** main(**String**[] str) {    HashSet myMap = **new** HashSet();  **String** s1 = **new** **String**("India");  **String** s2 = **new** **String**("India");    Country s3 = **new** Country("France");    Country s4 = **new** Country("France");    myMap.add(s1);    myMap.add(s2);    myMap.add(s3);    myMap.add(s4);    System.out.println(myMap);  }  } |

**options are :**  
A)France France India  
B)India India France France  
C)India France  
D)France France  
 **Answer:** A) France France India  
**Explanation:**  
As String class overrides hashcode and equals method, it won’t allow the same string twice in HashSet, but we did not implement hashcode and equals method for Country class, so each object will have different hashcode hence can be inserted in [HashSet](https://java2blog.com/how-hashset-works-in-java/).

**More explanation:**  
[hashcode and equals method in java](https://www.java2blog.com/2014/02/hashcode-and-equals-method-in-java.html)

**Question 2:**

|  |  |
| --- | --- |
| 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 | **package** com.org.arpit.java2blog;    import java.util.HashSet;    public **class** Employee {  public **String** name;    @Override  public **int** hashCode() {  **return** 31;  }    @Override  public **boolean** equals(**Object** obj) {  **return** **true**;  }    public static **void** main(**String** args[]) {    Employee employeeOne = **new** Employee();    Employee employeeTwo = **new** Employee();    employeeOne.name = "John";    employeeTwo.name = "Martin";    HashSet employeeSet = **new** HashSet();    employeeSet.add(employeeOne);    employeeSet.add(employeeTwo);    System.out.println(employeeSet.size());  }  } |

What will be output of above program:  
A. Compilation fails  
B. 1  
C. 2  
D. An Exception is thrown at run time.

**Answer:** B. 1

**Explanation :**  
As equals method always return true and hashcode return constant as 31. So when you try to put employeeTwo in [HashSet](https://java2blog.com/how-hashset-works-in-java/) when it will check for equals method, it will always return true, so employeeTwo won’t be added to HashSet

**More explanation:**  
[hashcode and equals method in java](https://www.java2blog.com/2014/02/hashcode-and-equals-method-in-java.html)

**Question 3:**

|  |  |
| --- | --- |
| 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  33  34  35  36  37 | **package** com.org.arpit.java2blog;    import java.util.HashSet;    public **class** Employee {  public **String** name;    @Override  public **boolean** equals(**Object** obj) {  **if** (**this** == obj)  **return** **true**;  **if** (obj == **null**)  **return** **false**;  **if** (getClass() != obj.getClass())  **return** **false**;    Employee other = (Employee) obj;  **if** (name == **null**) {  **if** (other.name != **null**)  **return** **false**;    } **else** **if** (!name.equals(other.name))  **return** **false**;  **return** **true**;  }    public static **void** main(**String** args[]) {    Employee employeeOne = **new** Employee();    Employee employeeTwo = **new** Employee();    employeeOne.name = "John";    employeeTwo.name = "John";    HashSet employeeSet = **new** HashSet();    employeeSet.add(employeeOne);    employeeSet.add(employeeTwo);    System.out.println(employeeSet.size());  }  } |

What will be output of above program:  
A. Compilation fails  
B. 1  
C. 2  
D. An Exception is thrown at run time.

**Answer:** C.2

Skip Ad

**Explanation :**  
As we did not implement hashcode method, each object will have different hashcode(memory address) by default, so employeeTwo will also be added to employeeSet and size of HashSet will be 2  
 **More explanation:**  
[hashcode and equals method in java](https://www.java2blog.com/2014/02/hashcode-and-equals-method-in-java.html)

**Question 4:**

|  |  |
| --- | --- |
| 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 | **package** com.org.arpit.java2blog;    import java.util.Iterator;  import java.util.TreeSet;    public **class** Employee **implements** Comparable {  public **String** name;    public **int** compareTo(**Object** o) {  **return** 0;  }    public static **void** main(**String** args[]) {    Employee employeeOne = **new** Employee();    Employee employeeTwo = **new** Employee();    employeeOne.name = "John";    employeeTwo.name = "Martin";    TreeSet employeeSet = **new** TreeSet();    employeeSet.add(employeeOne);    employeeSet.add(employeeTwo);    Iterator empIt = employeeSet.iterator();  **while** (empIt.hasNext()) {     System.out.println(empIt.next().name);    }  }  } |

What will be output of above program:  
A. Martin  
B. John  
C. John  
Martin  
D. Compilation fails.  
E. The code runs with no output.  
F. An exception is thrown at runtime.  
**Answer:**B. John

**Explanation :**  
As you can see we have overridden compareTo method in Employee class and always return 0.  
Following steps will take place:

* First element with “John” will be added to employeeSet.
* When we will added second element with martin, compareTo method will get called with employeeOne.compareTo(employeeTwo) and it will return 0.
* As compareTo method returns 0, employeeOne is equals to employeeTwo, so employeeTwo will not be added to treeSet.
* So output of above program is “John”

**More explanation:**  
[TreeSet in java](https://www.java2blog.com/2015/09/treeset-in-java.html)

**Question 5:**

How to sort a collection of custom Objects in Java?</span  
**Answer :**  
We need to implement comparable interface to custom object class(Lets 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)

**Question 6:**

What will be output of below code :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | **package** org.arpit.java2blog;    import java.util.LinkedList;  import java.util.Queue;    public **class** CollQuestion {  public static **void** main(**String**... args) {    Queue q = **new** LinkedList();    q.add("Delhi");    q.add("Mumbai");    q.add("Pune");    show(q);  }    public static **void** show(Queue q) {    q.add(**new** **Integer**(100));  **while** (!q.isEmpty())     System.out.print(q.poll() + " ");  }  } |

options are:  
A)Compile error : Integer can’t be added to the queue  
B)Delhi Mumbai Pune 100  
C)Delhi Mumbai Pune  
D)Delhi Mumbai  
Correct Answer: B. Delhi Mumbai Pune 100  
Explanation:  
As show method done not have any [generic](https://java2blog.com/java-generics-tutorial/) attached to it, integer can be also added to queue in show method.

**Question 7:  
Lets say you have class Employee as below:**

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40 | **package** com.org.arpit.java2blog;    import java.util.ArrayList;  import java.util.Collections;  import java.util.Comparator;  import java.util.List;    public **class** Employee {  public **String** name;  public **int** age;    public Employee(**String** name, **int** age) {  **super**();  **this**.name = name;  **this**.age = age;  }    public static **void** main(**String** args[]) {      List employees= **new** ArrayList();    Employee emp1= **new** Employee("John",26);    Employee emp2= **new** Employee("Martin",23);    Employee emp3= **new** Employee("John",20);    Employee emp4= **new** Employee("Martin",19);    Employee emp5= **new** Employee("Arpit",27);      employees.add(emp1);    employees.add(emp2);    employees.add(emp3);    employees.add(emp4);    employees.add(emp5);    System.out.println("List before sorting : ");  **for**(Employee e: employees)    {     System.out.println(e.name+" - "+e.age);    }  }  } |

You have list of employees, now you need to sort them on the basis of name then age. It means if names are equals, then sort it by age.  
**Answer:**

Write a anonymous comparator to sort it by name then age.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | Collections.sort(employees,**new** Comparator() {       @Override     public **int** compare(Employee o1, Employee o2) {  *// TODO Auto-generated method stub*  **if**(o1.name.compareTo(o2.name)==0)      {  **return** o1.age - o2.age;      }  **else**      {  **return** o1.name.compareTo(o2.name);      }     }    }); |

So complete program will be :

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62 | **package** com.org.arpit.java2blog;    import java.util.ArrayList;  import java.util.Collections;  import java.util.Comparator;  import java.util.List;    public **class** Employee {  public **String** name;  public **int** age;    public Employee(**String** name, **int** age) {  **super**();  **this**.name = name;  **this**.age = age;  }    public static **void** main(**String** args[]) {      List employees= **new** ArrayList();    Employee emp1= **new** Employee("John",26);    Employee emp2= **new** Employee("Martin",23);    Employee emp3= **new** Employee("John",20);    Employee emp4= **new** Employee("Martin",19);    Employee emp5= **new** Employee("Arpit",27);      employees.add(emp1);    employees.add(emp2);    employees.add(emp3);    employees.add(emp4);    employees.add(emp5);    System.out.println("List before sorting : ");  **for**(Employee e: employees)    {     System.out.println(e.name+" - "+e.age);    }      Collections.sort(employees,**new** Comparator() {       @Override     public **int** compare(Employee o1, Employee o2) {  *// TODO Auto-generated method stub*  **if**(o1.name.compareTo(o2.name)==0)      {  **return** o1.age - o2.age;      }  **else**      {  **return** o1.name.compareTo(o2.name);      }     }    });    System.out.println("--------------------------");    System.out.println("List after sorting : ");  **for**(Employee e: employees)    {     System.out.println(e.name+" - "+e.age);    }  }  } |

When you run able program, you will get below output:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | List before sorting :  John - 26  Martin - 23  John - 20  Martin - 19  Arpit - 27  --------------------------  List after sorting :  Arpit - 27  John - 20  John - 26  Martin - 19  Martin - 23 |

**Question 8:  
How HashMap works in java**

I will recommend you to go to [How HashMap works in java](https://www.java2blog.com/2014/02/how-hashmap-works-in-java.html) to understand it better.This is one of the most asked java collections interview questions.

**Question 9:**

|  |  |
| --- | --- |
| 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 | **package** com.org.arpit.java2blog;    import java.util.HashSet;    public **class** Employee {  public **String** name;    @Override  public **int** hashCode() {  **return** 31;  }    public static **void** main(**String** args[]) {    Employee employeeOne = **new** Employee();    Employee employeeTwo = **new** Employee();    employeeOne.name = "John";    employeeTwo.name = "John";    HashSet employeeSet = **new** HashSet();    employeeSet.add(employeeOne);    employeeSet.add(employeeTwo);    System.out.println(employeeSet.size());  }  } |

What will be output of above program:  
A. Compilation fails  
B. 1  
C. 2  
D. An Exception is thrown at run time.

**Answer:** C. 2

**Explanation :**  
As we did not override equals method here, it will have default implementation which return true if objects point to same reference (==) but here, as we have created two different object as employeeOne and employeeTwo, size of [HashSet](https://java2blog.com/how-hashset-works-in-java/) will be 2

**More explanation:**  
[hashcode and equals method in java](https://www.java2blog.com/2014/02/hashcode-and-equals-method-in-java.html)

**Question 10:  
Difference between Comparator and Comparable in java?**

Answer: This is also one of the most asked java collections interview questions. Please go to [difference between comparator and comparable](https://www.java2blog.com/2013/02/difference-between-comparator-and.html) for differences.

Please go through [java interview programs](https://www.java2blog.com/2015/08/java-interview-programs.html)for more such programs and [core java interview questions](https://www.java2blog.com/2016/07/core-java-interview-questions-and-answers.html) for more interview questions.

# Java Multithreading and Concurrency Interview Questions

In this tutorial, we are going to see Multithreading interview questions with answers.  
Here is list of Java Multithreading interview questions.

Table of Contents [[hide](https://java2blog.com/java-multithreading-interview-questions-and-answers/)]

* [Java multithreading interview questions](https://java2blog.com/java-multithreading-interview-questions-and-answers/#Java_multithreading_interview_questions)
  + [1. What is thread in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#1_What_is_thread_in_java)
  + [2. What is Multithreading?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#2_What_is_Multithreading)
  + [3. What are ways to create a thread in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#3_What_are_ways_to_create_a_thread_in_java)
  + [4. Thread vs Runnable which is better approach to create a thread?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#4_Thread_vs_Runnable_which_is_better_approach_to_create_a_thread)
  + [5. What are differences between thread and process?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#5_What_are_differences_between_thread_and_process)
  + [6. What are differences between Sleep and wait in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#6_What_are_differences_between_Sleep_and_wait_in_java)
  + [7. Why wait(), notify() And notifyAll() methods are in Object Class?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#7_Why_wait_notify_And_notifyAll_methods_are_in_Object_Class)
  + [8. Why wait and notify method are called from synchronized block?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#8_Why_wait_and_notify_method_are_called_from_synchronized_block)
  + [9. Why sleep() and yield() are static methods in Thread class?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#9_Why_sleep_and_yield_are_static_methods_in_Thread_class)
  + [10. Define states of thread in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#10_Define_states_of_thread_in_java)
  + [11. Can we call run method directly to start a thread?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#11_Can_we_call_run_method_directly_to_start_a_thread)
  + [12. Can we start a thread twice in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#12_Can_we_start_a_thread_twice_in_java)
  + [13. How to make a main thread wait until all other threads finished execution?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#13_How_to_make_a_main_thread_wait_until_all_other_threads_finished_execution)
  + [14. What are daemon threads?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#14_What_are_daemon_threads)
  + [15. How can you change user thread to daemon thread?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#15_How_can_you_change_user_thread_to_daemon_thread)
  + [16. What is Synchronization?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#16_What_is_Synchronization)
  + [17. What is need of Synchronization?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#17_What_is_need_of_Synchronization)
  + [18. Can you explain about Object level locking and class level locking?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#18_Can_you_explain_about_Object_level_locking_and_class_level_locking)
  + [19. Can two threads execute static and non static methods concurrently?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#19_Can_two_threads_execute_static_andnon_staticmethods_concurrently)
  + [20. If one method of class is synchronized and other method of same class is not synchronized? Can they be executed concurrently by two threads?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#20_If_one_method_ofclassis_synchronized_andothermethod_ofsameclass_is_not_synchronized_Can_they_be_executed_concurrently_by_twothreads)
  + [21. Is it safe to call a synchronized method from another synchronized method?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#21_Is_it_safe_to_call_a_synchronized_method_from_another_synchronized_method)
  + [22. What is deadlock?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#22_What_is_deadlock)
  + [23. What are differences between notify and notifyall?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#23_What_are_differences_between_notify_and_notifyall)
  + [24. What is use of volatile keyword in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#24_What_is_use_of_volatile_keyword_in_java)
  + [25. You have started 3 threads and you ought to be certain that main thread finish last. How will you get it done?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#25_You_have_started_3_threads_and_you_ought_to_be_certain_that_main_thread_finish_last_How_will_you_get_it_done)
  + [26.How can you print even odd number using 2 threads?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#26How_can_you_print_even_odd_number_using_2_threads)
  + [27. What is ThreadLocal in java?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#27_What_is_ThreadLocal_in_java)
  + [28. What is Thread dump?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#28_What_is_Thread_dump)
  + [29. What happens if you don’t override run() method of Thread class?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#29_What_happens_if_you_don8217t_override_run_method_of_Thread_class)
* [Java Concurrency Interview questions](https://java2blog.com/java-multithreading-interview-questions-and-answers/#Java_Concurrency_Interview_questions)
  + [30. What is executor framework?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#30_What_is_executor_framework)
  + [31. What is Thread pool?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#31_What_is_Thread_pool)
  + [32. Can you list down important factory methods of Executors class?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#32_Can_you_list_down_important_factory_methods_of_Executors_class)
  + [33. Can you write a code to implement newFixedThreadPool?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#33_Can_you_write_a_code_to_implement_newFixedThreadPool)
  + [34. What is BlockingQueue](https://java2blog.com/java-multithreading-interview-questions-and-answers/#34_What_is_BlockingQueue)
  + [35. What are difference between Runnable and Callable interface in java](https://java2blog.com/java-multithreading-interview-questions-and-answers/#35_What_are_difference_between_Runnable_and_Callable_interface_in_java)
  + [36. What is Lock interface? What are its advantages over synchronized block?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#36_What_is_Lock_interface_What_are_its_advantages_over_synchronized_block)
  + [37. What is Condition interface?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#37_What_is_Condition_interface)
  + [38. Implement custom BlockingQueue using Lock and Condition?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#38_Implement_custom_BlockingQueue_using_Lock_and_Condition)
  + [39. What is CountDownLatch?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#39_What_is_CountDownLatch)
  + [40. What is CyclicBarrier?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#40_What_is_CyclicBarrier)
  + [41. What is Semaphore?](https://java2blog.com/java-multithreading-interview-questions-and-answers/#41_What_is_Semaphore)

## Java multithreading interview questions

### 1. What is thread in java?

**Answer:**  
Thread can be called as light weight process. It can be referred as smallest part of process which can be executed concurrently with other parts(threads) of process.

### 2. What is Multithreading?

**Answer:**  
Multithreading is execution of multiple threads concurrently. Java supports multithreading , so it allows your application to perform two or more task concurrently. Multithreading can be of advantage specially when now a days, machine has multiple CPUs, so multiple tasks can be executed concurrently.

### 3. What are ways to create a thread in java?

**Answer:**  
There are two ways to create a thread in java

* By extending thread class
* By implementing Runnable interface.

You can read about more at [Java thread example](https://www.java2blog.com/?p=346).

## Further reading:

##### [Top 100+ Java coding interview questions](https://java2blog.com/java-coding-interview-questions/)

[Read more](https://java2blog.com/java-coding-interview-questions/) →

##### [Java Collections interview questions](https://java2blog.com/java-collections-interview-questions/)

[Read more](https://java2blog.com/java-collections-interview-questions/) →

### 4. Thread vs Runnable which is better approach to create a thread?

**Answer:**  
Implementing Runnable interface is considered to be better approach than Extending Thread due to following reasons.

* Java does not support multiple inheritance so if you extend Thread class and you can not extend any other class which is needed in most of the cases.
* Runnable interface represents a task and this can be executed with help of Thread class or Executors.
* When you use inheritance, it is because you want to extend some properties of parent, modify or improve class behavior. But if you are extending thread class just to create thread, so it may not be recommended behavior for Object Oriented Programming.

### 5. What are differences between thread and process?

**Answer:**  
You can go through [difference between process and thread](https://www.java2blog.com/?p=12) to see the differences.

### 6. What are differences between Sleep and wait in java?

|  |  |  |
| --- | --- | --- |
| **Parameter** | **wait** | **sleep** |
| Synchronized | 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/?p=360) for more details.

### 7. Why wait(), notify() And notifyAll() methods are in Object Class?

**Answer:**  
Thread waits for lock associated with the object and notify other threads which are waiting for same lock.

If wait(), notify() and notifyAll() will be in thread class, then each thread has to be aware of status of another thread and that does not make sense as each thread runs independent of other thread and has no specific knowledge about other thread.

Please refer for more details:  
[Why wait(), notify() And notifyAll() methods are in Object Class](https://java2blog.com/why-wait-notify-notifyall-methods-object-class/)

### 8. Why wait and notify method are called from synchronized block?

**Answer:**  
wait() is called, so that thread can wait on some condition. When condition is met, then thread has to give up the lock.

To give up the lock, thread has to own it first. Thread can acquire lock by enter into synchronized context.

If wait method is called outside of synchronized context, then it will throw IllegalMonitorStateException.

### 9. Why sleep() and yield() are static methods in Thread class?

**Answer:**  
You can call sleep() and yield() method on current executing thread. If there is in wait state, you can not call these methods.

To avoid the confusion for programmers, there methods are made static.

### 10. Define states of thread in java?

**Answer:**  
There are 5 states of thread in java

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

**Runnable:**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.

### 11. Can we call run method directly to start a thread?

**Answer:**  
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/?p=206) for more details

### 12. Can we start a thread twice in java?

**Answer:**  
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/?p=207) for more details

### 13. How to make a main thread wait until all other threads finished execution?

**Answer:**  
You can make use of join method to achieve above scenario.  
You can read more about [join method](https://www.java2blog.com/?p=344).

### 14. What are daemon threads?

**Answer:**

Daemon threads are low-priority background threads which provide services to user threads. Its life depends on user threads. If no user thread is running then JVM can exit even if daemon threads are running. [JVM]([JVM](https://java2blog.com/java-virtual-machine-architecture/) “JVM”) does not wait for daemon threads to finish.

### 15. How can you change user thread to daemon thread?

**Answer:**setDaemon() method can be used to mark thread as user thread. If you put setDaemon(true), it makes thread as daemon.

### 16. What is Synchronization?

**Answer:**

Synchronization is ability to restrict access to shared resource to only one thread. When two or more threads need access to shared resource, there has to be some mechanism such that shared resource will be used by only one thread. The process by which we can achieve it is called Synchronization.

### 17. What is need of Synchronization?

Let’s understand this with the help of an example.

Let’s say you want to count number of request you got for a particular URL. If you get two requests at the same time, then count may be inconsistent.

**Without Synchronization:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | **package** org.arpit.java2blog;    public **class** RequestCounter {    private **int** count;    public **int** incrementCount()  {  count++;  **return** count;  }  } |

**For example:**  
Thread T1 sees count as 20 and increments it to 21. At the same time, thread t2 also sees count as 20 and increment it to 21. This shows that count became inconsistent.

**With Synchronization:**

You can achieve Synchronization using two ways.

* synchronized method
* synchronized block

You can not use synchronized with  instance or class variables.

**synchronized method**

You can make whole incrementCount() method synchronized so no two thread can access it parallelly.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | **package** org.arpit.java2blog;    public **class** RequestCounter {    private **int** count;    public synchronized **int** incrementCount()  {    count++;  **return** count;  }  } |

**For example:**  
Thread T1 sees count as 20 and increments it to 21. At the same time, Thread t2 will now see count as 21 and increment it to 22.  
**synchronized block**

You can make use block to synchronize critical section in  incrementCount() method so no two thread can access block concurrently.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | **package** org.arpit.java2blog;    public **class** RequestCounter {    private **int** count;    public **int** incrementCount() {    synchronized (**this**) {     count++;  **return** count;    }  }  } |

**For example:**  
Thread T1 sees count as 20 and increment it to 21. At the same time, thread t2 will now see count as 21 and increment it to 22.

### 18. Can you explain about Object level locking and class level locking?

There are two types of locking in java.

* Object level locking
* Class level locking

You can refer [Object level locking and class level locking](https://www.java2blog.com/2017/01/object-level-locking-vs-class-level-locking-java.html) for more details.

### 19. Can two threads execute static and non static methods concurrently?

**Answer:**  
Yes, Since two threads will acquire lock on different objects, they can be executed concurrently without any issues.

### 20. If one method of class is synchronized and other method of same class is not synchronized? Can they be executed concurrently by two threads?

**Answer:**  
Yes, because one thread will require lock to get into synchronized block but second thread which will execute non synchronized method that won’t require any lock, so it can be executed concurrently.

### 21. Is it safe to call a synchronized method from another synchronized method?

**Answer:**  
Yes, it is safe to call a synchronized method from another synchronized method because when you call synchronized method, you will get lock on **this** object and when you call another synchronized method of same class, it is safe to execute as it already has lock on **this** object.  
**For example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | public synchronized **void** method1() {    method2();  *// some code*  }    public synchronized **void** method2() {  *// some code*  } |

You are actually doing this.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | public **void** method1() {    synchronized (**this**) {     method2();  *// some code*    }    }    public **void** method2() {    synchronized (**this**) {  *// some code*    }  } |

Here if any thread calls method2 from method1, it will already have lock on **this** object hence It is safe to execute.

### 22. What is deadlock?

**Answer:**

**Answer:**  
Deadlock is a situation where two or more threads are waiting for each other to release the resource.

For example:

Thread 1 have lock over object 1 and waiting to get lock on object 2. Thread 2 have lock over object 2 and waiting to get lock on object 1. In this scenario, both threads will wait for each other indefinitely.

### 23. What are differences between notify and notifyall?

**Answer:**  
You can go through [difference between notify() and notifyall()](https://www.java2blog.com/?p=11)to see the differences.

### 24. What is use of volatile keyword in java?

**Answer:**  
If you make any variable volatile, then this [variable](https://java2blog.com/variables-java/) will be read from main memory rather then CPU cache, so that each thread will get updated value of the variable.

### 25. You have started 3 threads and you ought to be certain that main thread finish last. How will you get it done?

**Answer:**  
You can use thread’s join() method to attain this scenario.

Let’s see with the help of example:  
**Without join() method:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | **package** org.arpit.java2blog;    **class** MyRunnable **implements** Runnable{        @Override      public **void** run() {          System.out.println("Current executing thread: "+Thread.currentThread().getName());      }      public static **void** main(**String**[] args) {          MyRunnable runnable=**new** MyRunnable();          Thread thread1=**new** Thread(runnable,"Thread1");          Thread thread2=**new** Thread(runnable,"Thread2");          Thread thread3=**new** Thread(runnable,"Thread3");            thread1.start();          thread2.start();          thread3.start();            System.out.println("Exiting main thread");      }  } |

**Output:**

Exiting main thread  
Current executing thread: Thread2  
Current executing thread: Thread1  
Current executing thread: Thread3

As you can see, main thread exited first here.  
**With join() method:**

|  |  |
| --- | --- |
| 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;    **class** MyRunnable **implements** Runnable{        @Override      public **void** run() {          System.out.println("Current executing thread: "+Thread.currentThread().getName());      }      public static **void** main(**String**[] args) {          MyRunnable runnable=**new** MyRunnable();          Thread thread1=**new** Thread(runnable,"Thread1");          Thread thread2=**new** Thread(runnable,"Thread2");          Thread thread3=**new** Thread(runnable,"Thread3");            thread1.start();          thread2.start();          thread3.start();    **try** {              thread1.join();              thread2.join();              thread3.join();          } **catch** (InterruptedException e) {              e.printStackTrace();          }            System.out.println("Exiting main thread");        }  } |

**Output:**

Current executing thread: Thread3  
Current executing thread: Thread2  
Current executing thread: Thread1  
Exiting main thread

As you can see, main thread exited last after calling join() method on all three threads i.e. thread1, thread2 and thread3.

### 26.How can you print even odd number using 2 threads?

5.6M

Print, Install Software and Email - Linux Tutorial 7

You have been given 2 threads i.e. T1 and T2. You need to print odd numbers using one thread and even numbers using another thread.  
For example:  
Let’s say you have to [Print even odd number using 2 threads](https://java2blog.com/print-even-odd-numbers-threads-java/) upto 10. Here is the sample expected output.

1 T1  
2 T2  
3 T1  
4 T2  
5 T1  
6 T2  
7 T1  
8 T2  
9 T1  
10 T2

**Answer:**  
Here is the solution using [wait and notify](https://java2blog.com/wait-notify-and-notifyall-method-in/) in java.  
Here is the approach you can use.

* If num%2==1, then print using T1 and increment it else T1 will go to wait state.
* If num%2==0, then print using T2 and increment it else T2 will go to wait state.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48 | **package** org.arpit.java2blog;    **class** PrintOddEven **implements** Runnable{        public **int** MAX\_NUMBER =10;      static **int**  number=1;  **int** rem;      static **Object** lock=**new** **Object**();        PrintOddEven(**int** remainder)      {  **this**.rem =remainder;      }        @Override      public **void** run() {  **while** (number < MAX\_NUMBER) {              synchronized (lock) {  **while** (number % 2 != rem) { *// wait*  **try** {                          lock.wait();                      } **catch** (InterruptedException e) {                          e.printStackTrace();                      }                  }                  System.out.println(Thread.currentThread().getName() + " " + number);                  number++;                  lock.notifyAll();              }          }      }  }  public **class** OddEvenMain {      public static **void** main(**String**[] args) {            PrintOddEven oddRunnable=**new** PrintOddEven(1);          PrintOddEven evenRunnable=**new** PrintOddEven(0);            Thread t1=**new** Thread(oddRunnable,"T1");          Thread t2=**new** Thread(evenRunnable,"T2");            t1.start();          t2.start();        }  } |

**Output:**

T1 1  
T2 2  
T1 3  
T2 4  
T1 5  
T2 6  
T1 7  
T2 8  
T1 9  
T2 10

### 27. What is ThreadLocal in java?

**Answer:**  
TheadLocal helps you to create [variable](https://java2blog.com/variables-java/) that can be read and write only by that thread. Two threads can not see each other’s ThreadLocal variable, so even if they are executing same code, there won’t be any race condition and code will be thread safe.  
Here is an example:  
Create a ThreadLocalRunnable which will have ThreadLocal variable.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | **package** org.arpit.java2blog;    public **class** ThreadLocalRunnable **implements** Runnable {    *// ThreadLocal of Integer type*      private ThreadLocal<**Integer**> tl = **new** ThreadLocal<**Integer**>();        @Override      public **void** run() {          tl.set( (**int**) (Math.random() \* 10) );    **try** {              Thread.sleep(1000);          } **catch** (InterruptedException e) {          }            System.out.println(Thread.currentThread().getName()+":"+tl.get());      }  } |

Create main class named ThreadLocalMain.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | **package** org.arpit.java2blog;    public **class** ThreadLocalMain {        public static **void** main(**String**[] args) throws InterruptedException {          ThreadLocalRunnable tl = **new** ThreadLocalRunnable();            Thread t1 = **new** Thread(tl,"Thread1");          Thread t2 = **new** Thread(tl,"Thread2");            t1.start();          t2.start();            t1.join();          t2.join();      }  } |

**Output:**

Thread2:8  
Thread1:7

As you can see, Thread1 and Thread2 shares same instance of ThreadLocalRunnable and both set different values for ThreadLocal variable. If we have used synchronized rather than ThreadLocal, later executing thread would have overridden value set by first thread.

### 28. What is Thread dump?

**Answer:**  
A thread dump is snapshot of all the active threads that are part of process. It contains lot of information about thread and its current state.

A thread dump is very useful when you want to analyze any issue related to deadlock.

You can use jdk tools such as jvisualVM,jstack and Java Mission control. These tools are part of JDK installation and very handy to use.

### 29. What happens if you don’t override run() method of Thread class?

**Answer:**  
If you don’t override run() method thread class, then nothing will happen.

When you call `run() method of thread class, it calls target.run() method. Here target is instance of Runnable. When you directly create Thread using default constructor, target is set to null.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | @Override      public **void** run() {  **if** (target != **null**) {              target.run();          }      } |

Let’s see with the help of example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | **package** org.arpit.java2blog;    public **class** MyThread **extends** Thread {    *// no run method*        public static **void** main(**String** args[]){          Thread thread1 = **new** MyThread();          System.out.println("Before thread execution");          thread1.start();          System.out.println("After thread execution");      }  } |

**Output:**

Before thread execution  
After thread execution

As you can see, nothing happened when we called thread1.start().

## Java Concurrency Interview questions

### 30. What is executor framework?

**Answer:**  
Java 5 has introduced new framework named executor framework for managing threads.

If you need to create thousands of the threads, your application performance may suffer and also, maintenance of each thread is also overhead. Executor framework solves this problem by limiting number of threads and reusing the same threads once threads had completed its current task.

**Read more :**[Executor framework](https://java2blog.com/java-executor-framework-tutorial-example/)

### 31. What is Thread pool?

**Answer:**  
Thread pool represents set of worker threads that are waiting for the jobs and these worker threads can be reused many times.

Whenever a task needs to be performance, service provider will pull out a thread out of thread pool to perform the task. If no worker threads are available, then task has to wait for execution.

java.util.concurrent.Executors class provides factory methods to create the thread pools.

### 32. Can you list down important factory methods of Executors class?

**Answer:**  
Here are the few important methods of Executors class.

[**newFixedThreadPool**](https://java2blog.com/java-newfixedthreadpool-example/)**:** This method returns a thread pool whose maximum size is fixed. If all threads are busy in execution of tasks and additional tasks are submitted, then they have to wait in queue until any thread is available for executing these tasks.

[**newCachedThreadPool**](https://java2blog.com/java-newcachedthreadpool-example/)**:** This method returns unbounded thread pool. If threads are not used for certain defined time (keepAliveTime), then it will kill extra threads.

[**newSingleThreadedExecutor**](https://java2blog.com/java-newsinglethreadexecutor-example/)**:** This method returns an Executor with single thread.

[**newScheduledThreadPool**](https://java2blog.com/java-scheduledthreadpoolexecutor-example/)**:** This method returns thread pool with fixed size, that can schedule task periodically or with given delay.

### 33. Can you write a code to implement newFixedThreadPool?

**Answer:**  
You need to use Executors.newFixedThreadPool(numberOfThreads) to create newFixedThreadPool.

|  |  |
| --- | --- |
| 1  2  3 | ExecutorService executorService=Executors.newFixedThreadPool(noOfThreads); |

Let’s create very simple example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | **package** org.arpit.java2blog;    public **class** RunnableTask **implements** Runnable {        private **String** taskName;        public RunnableTask(**String** taskName) {  **super**();  **this**.taskName = taskName;      }        @Override      public **void** run() {            System.out.println("Starting "+ taskName);  **for** (**int** i = 1; i <= 3; i++) {              System.out.println("Executing "+ taskName +" with "+Thread.currentThread().getName()+"====>"+i);          }          System.out.println("Finishing "+ taskName);      }  } |

Create main class FixedThreadPoolMain.java.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | **package** org.arpit.java2blog;    public **class** ThreadLocalRunnable **implements** Runnable {    *// ThreadLocal of Integer type*      private ThreadLocal<**Integer**> tl = **new** ThreadLocal<**Integer**>();        @Override      public **void** run() {          tl.set( (**int**) (Math.random() \* 10) );    **try** {              Thread.sleep(1000);          } **catch** (InterruptedException e) {          }            System.out.println(Thread.currentThread().getName()+":"+tl.get());      }  } |

**Output:**

Starting RunnableTask 1  
Starting RunnableTask 2  
Starting RunnableTask 3  
Executing RunnableTask 2 with pool-1-thread-2====>1  
Executing RunnableTask 1 with pool-1-thread-1====>1  
Executing RunnableTask 3 with pool-1-thread-3====>1  
Executing RunnableTask 1 with pool-1-thread-1====>2  
Executing RunnableTask 2 with pool-1-thread-2====>2  
Executing RunnableTask 1 with pool-1-thread-1====>3  
Executing RunnableTask 3 with pool-1-thread-3====>2  
Executing RunnableTask 2 with pool-1-thread-2====>3  
Executing RunnableTask 3 with pool-1-thread-3====>3  
Finishing RunnableTask 1  
Finishing RunnableTask 2  
Finishing RunnableTask 3  
Starting RunnableTask 4  
Executing RunnableTask 4 with pool-1-thread-3====>1  
Executing RunnableTask 4 with pool-1-thread-3====>2  
Executing RunnableTask 4 with pool-1-thread-3====>3  
Finishing RunnableTask 4  
Starting RunnableTask 5  
Executing RunnableTask 5 with pool-1-thread-3====>1  
Executing RunnableTask 5 with pool-1-thread-3====>2  
Executing RunnableTask 5 with pool-1-thread-3====>3  
Finishing RunnableTask 5

As you can notice in the preceding output, first 3 tasks started executed immediately as we have maximum 3 threads in newFixedThreadPool. Once these 3 tasks are finished, then RunnableTask 4 and RunnableTask 5 got executed.

### 34. What is BlockingQueue

**Answer:**  
[BlockingQueue](https://java2blog.com/blockingqueue-in-java/) is special type of queue which is used when producer threads produce object and consumer threads consume object.

Producer thread keeps inserting objects in the queue unless it is full and once it is full, it will be blocked unless consumer threads start consuming.

Similarly, consumer thread keep consuming objects until it is empty. Once it is empty, it will be blocked unless producer threads start producing.

Here are two important methods of BlockingQueue.  
put(): This method to put objects in queue until queue is full and waits for consumer thread to take object after that.

take(): This method is used to take objects from the queue until it becomes empty. Once it is empty, it waits for producer threads to start producing the objects and put into the queue.

Let’s see simple example of BlockingQueue.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74 | **package** org.arpit.java2blog;    import java.util.concurrent.ArrayBlockingQueue;  import java.util.concurrent.BlockingQueue;    public **class** BlockingQueuePCExample {        public static **void** main(**String**[] args) {            BlockingQueue<**String**> queue=**new** ArrayBlockingQueue<>(5);          Producer producer=**new** Producer(queue);          Consumer consumer=**new** Consumer(queue);          Thread producerThread = **new** Thread(producer);          Thread consumerThread = **new** Thread(consumer);            producerThread.start();          consumerThread.start();        }        static **class** Producer **implements** Runnable {            BlockingQueue<**String**> queue=**null**;            public Producer(BlockingQueue queue) {  **super**();  **this**.queue = queue;          }            @Override          public **void** run() {    **try** {                      System.out.println("Producing element 1");                      queue.put("Element 1");                      Thread.sleep(1000);                      System.out.println("Producing element 2");                      queue.put("Element 2");                      Thread.sleep(1000);                      System.out.println("Producing element 3");                      queue.put("Element 3");                  } **catch** (InterruptedException e) {                        e.printStackTrace();                  }          }      }        static **class** Consumer **implements** Runnable {            BlockingQueue<**String**> queue=**null**;            public Consumer(BlockingQueue queue) {  **super**();  **this**.queue = queue;          }            @Override          public **void** run() {    **while**(**true**)              {  **try** {                      System.out.println("Consumed "+queue.take());                  } **catch** (InterruptedException e) {                      e.printStackTrace();                  }              }          }        }  } |

**Output:**

Producing element 1  
Consumed Element 1  
Producing element 2  
Consumed Element 2  
Producing element 3  
Consumed Element 3

### 35. What are difference between Runnable and Callable interface in java

Refer: [Difference between Runnable and Callable in java](https://java2blog.com/difference-runnable-callable-java/)

### 36. What is Lock interface? What are its advantages over synchronized block?

java.util.concurrent.lock.Lock is introduced in java 1.5 and it provides important operation for blocking.

It is more flexible and convenient synchronization aid than standard synchronized block.

Advantages of Locks are:

* Lock supports fairness which can not be achieved in case of synchronized block. We can specify fairness property to make sure longest waiting thread get fair chance for execution.
* We can use Lock interface’s lock() and unlock() operation in different methods.
* We can use tryLock() method if we don’t want to block the thread. tryLock() method attempts to Lock immediately and return true if the locking succeeds.
* We can use lockInterruptibly() method to interrupt the thread which is waiting for the lock.

### 37. What is Condition interface?

A Condition instance is intrinsically bound to a lock. You can use newCondtion() method of Lock interface to obtain Condition instance.

Condition is useful when you want to create multiple wait-sets per lock.

**For example:**  
In producer consumer problem, if buffer is full, then producer can wait on one Condition instance notEmpty and if buffer is empty, then consumer can wait on another Condition instance notFull unless elements are available.

In case, spaces become available in buffer, consumer can signal producer using Condition instance notEmpty. Similarly, when producer starts adding elements to buffer, it can signal consumer on Condition instance notFull.

### 38. Implement custom BlockingQueue using Lock and Condition?

**Answer:**  
Here is the implementation of [custom BlockingQueue](https://java2blog.com/custom-blockingqueue-implementation-java/) using Lock and Condition.

* Use array to store elements in BlockingQueue. Size of array will define maximum capacity of the Blocking Queue.
* Create one lock and two conditions objects: notFull and notEmpty.
* If Queue is full, then producer has to wait on notFull condition object
* If Queue is empty, then consumer has to wait on notEmpty condition object
* Create two threads producer and consumer which will share same CustomBlockingQueue object

Create a class named CustomBlockingQueue.java

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68 | **package** org.arpit.java2blog.entry;    import java.util.concurrent.locks.Condition;  import java.util.concurrent.locks.Lock;  import java.util.concurrent.locks.ReentrantLock;    public **class** CustomBlockingQueue {        final Lock lock = **new** ReentrantLock();    *// Conditions*      final Condition notFull = lock.newCondition();      final Condition notEmpty = lock.newCondition();    *// Array to store element for CustomBlockingQueue*      final **Object**[] arr = **new** **Object**[3];  **int** putIndex, takeIndex;  **int** count;        public **void** put(**Object** x) throws InterruptedException {            lock.lock();  **try** {  **while** (count == arr.length){  *// Queue is full, producers need to wait*                  notFull.await();              }                arr[putIndex] = x;              System.out.println("Putting in Queue - " + x);              putIndex++;  **if** (putIndex == arr.length){                  putIndex = 0;              }  *// Increment the count for the array*              ++count;  *// Send signal to consumer*              notEmpty.signal();          } **finally** {              lock.unlock();          }      }        public **Object** take() throws InterruptedException {          lock.lock();  **try** {  **while** (count == 0){  *// Queue is empty, consumers need to wait*                  notEmpty.await();              }  **Object** x = arr[takeIndex];              System.out.println("Taking from queue - " + x);              takeIndex++;  **if** (takeIndex == arr.length){                  takeIndex = 0;              }  *// reduce the count for the array*              --count;  *// send signal producer*              notFull.signal();  **return** x;          } **finally** {              lock.unlock();          }      }  } |

Create main class CustomBlockingQueueMain.java.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | **package** org.arpit.java2blog.entry;    public **class** CustomBlockingQueueMain {        public static **void** main(**String**[] args) {          CustomBlockingQueue customBlockingQueue = **new** CustomBlockingQueue();  *// Creating producer and consumer threads*          Thread producer = **new** Thread(**new** Producer(customBlockingQueue));          Thread consumer = **new** Thread(**new** Consumer(customBlockingQueue));            producer.start();          consumer.start();      }  }    **class** Producer **implements** Runnable {        private CustomBlockingQueue cbq;        public Producer(CustomBlockingQueue cbq){  **this**.cbq = cbq;      }      @Override      public **void** run() {  **for** (**int** i = 1; i <= 5; i++) {  **try** {                  cbq.put(i);              } **catch** (InterruptedException e) {                  e.printStackTrace();              }          }      }    }    **class** Consumer **implements** Runnable {      private CustomBlockingQueue cbq;        public Consumer(CustomBlockingQueue cbq){  **this**.cbq = cbq;      }      @Override      public **void** run() {  **for** (**int** i = 1; i <= 5; i++) {  **try** {                  cbq.take();              } **catch** (InterruptedException e) {                  e.printStackTrace();              }          }      }    } |

**Output:**

Putting in Queue – 1  
Putting in Queue – 2  
Putting in Queue – 3  
Taking from queue – 1  
Taking from queue – 2  
Taking from queue – 3  
Putting in Queue – 4  
Putting in Queue – 5  
Taking from queue – 4  
Taking from queue – 5

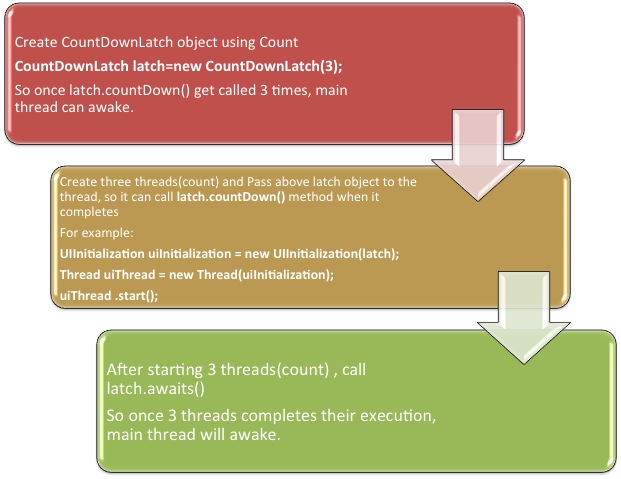
As you can see in the output, Producer thread got blocked after putting 3 elements in CustomBlockingQueue as array size was given as 3.

### 39. What is CountDownLatch?

[CountDownLatch](https://java2blog.com/countdownlatch-in-java/) is synchronization aid that allows one or more threads to wait until specified operations in other threads complete.

CountDownLatch is initialized by count. Whenever a thread calls latch.awaits() where latch is instance of CountDownLatch, so it will wait until count becomes zero or thread is interrupted by another thread.

So when other threads call latch.countDown(), count is reduced by 1. Once count reaches 0, thread which called latch.await() will be resumed.



**Read more :**[CountDownLatch in java](https://java2blog.com/countdownlatch-in-java/)

### 40. What is CyclicBarrier?

[CyclicBarrier](https://java2blog.com/java-cyclicbarrier-example/) is similar to CountDownLatch, but you can reuse once count reaches to 0.

You can also trigger common event once count reaches to 0 in case of CyclicBarrier.

**Read more :**[CyclicBarrier in java](https://java2blog.com/java-cyclicbarrier-example/)

### 41. What is Semaphore?

When we want to limit number of concurrent threads accessing any resources, we can use Semaphore.

[Semaphore](https://java2blog.com/java-semaphore-example/) maintains set of permits, if permits are not available, thread has to wait.

Semaphores can be used to implement resource pools or bounded collection.

Let’s implement binary semaphore.

|  |  |
| --- | --- |
| 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  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70 | **package** org.arpit.java2blog.entry;    import java.util.concurrent.Semaphore;  public **class** SemaphoreExampleMain  {  *// Creating semaphore with 1 permit*      static Semaphore semaphore = **new** Semaphore(1);      static **class** MyThread **extends** Thread      {  **String** name;  *//constructor of the MyThread class*          MyThread(**String** name)          {  **this**.name = name;          }          public **void** run()          {  **try**              {                  System.out.println(name+" : Getting lock");                  System.out.println(name + " : Available Semaphore permits is: "+ semaphore.availablePermits());                    semaphore.acquire();                  System.out.println(name + " : acquired the lock");  **try**                  {  **for** (**int** i = 1; i <= 5; i++)                      {                          System.out.println(name + " : is performing task " + i+ ", available Semaphore permits : "+ semaphore.availablePermits());  *//sleep 2 second*                          Thread.sleep(2000);                      }                  }  **finally**                  {                      System.out.println(name + " : releasing lock...");  *// release the permit using release() method*                      semaphore.release();                  }              }  **catch** (InterruptedException e)              {                  e.printStackTrace();              }          }      }  *//main method*      public static **void** main(**String**[] args)      {            System.out.println("Total available Semaphore permits is: "+ semaphore.availablePermits());  *//creating 2 threads namely Thread 1 and Thread 2*          MyThread thread1 = **new** MyThread("Thread 1");  *//staring thread 1*          thread1.start();          MyThread thread2 = **new** MyThread("Thread 2");  *//staring thread 2*          thread2.start();    **try** {              thread1.join();              thread2.join();          } **catch** (InterruptedException e) {              e.printStackTrace();          }          System.out.println("Total available Semaphore permits is: "+ semaphore.availablePermits());      }  } |

**Output:**

Total available Semaphore permits is: 1  
Thread 2 : Getting lock  
Thread 1 : Getting lock  
Thread 2 : Available Semaphore permits is: 1  
Thread 1 : Available Semaphore permits is: 1  
Thread 1 : acquired the lock  
Thread 1 : is performing task 1, available Semaphore permits : 0  
Thread 1 : is performing task 2, available Semaphore permits : 0  
Thread 1 : is performing task 3, available Semaphore permits : 0  
Thread 1 : is performing task 4, available Semaphore permits : 0  
Thread 1 : is performing task 5, available Semaphore permits : 0  
Thread 1 : releasing lock…  
Thread 2 : acquired the lock  
Thread 2 : is performing task 1, available Semaphore permits : 0  
Thread 2 : is performing task 2, available Semaphore permits : 0  
Thread 2 : is performing task 3, available Semaphore permits : 0  
Thread 2 : is performing task 4, available Semaphore permits : 0  
Thread 2 : is performing task 5, available Semaphore permits : 0  
Thread 2 : releasing lock…  
Total available Semaphore permits is: 1

# Java String Interview questions and answers

Table of Contents [[hide](https://java2blog.com/java-string-interview-questions-and-answers/)]

* [1. Why String is declared final or immutable in java?](https://java2blog.com/java-string-interview-questions-and-answers/#1_Why_String_is_declared_final_or_immutable_in_java)
* [2. How to reverse a String in java? Can you write a program without using any java inbuilt methods?](https://java2blog.com/java-string-interview-questions-and-answers/#2_How_to_reverse_a_String_in_java_Can_you_write_a_program_without_using_any_java_inbuilt_methods)
* [3. How to check if two Strings are anagram in java?](https://java2blog.com/java-string-interview-questions-and-answers/#3_How_to_check_if_two_Strings_are_anagram_in_java)
* [4. How to check if String has all unique characters in java?](https://java2blog.com/java-string-interview-questions-and-answers/#4_How_to_check_if_String_has_all_unique_characters_in_java)
* [5. How to check if one String is rotation of another String in java?](https://java2blog.com/java-string-interview-questions-and-answers/#5_How_to_check_if_one_String_is_rotation_of_another_String_in_java)
* [6. Write a java program to find duplicate characters in String in java?](https://java2blog.com/java-string-interview-questions-and-answers/#6_Write_a_java_program_to_find_duplicate_characters_in_String_in_java)
* [7. Find first non repeated character in String in java?](https://java2blog.com/java-string-interview-questions-and-answers/#7_Find_first_non_repeated_character_in_String_in_java)
* [8. Find all substrings of String in java?](https://java2blog.com/java-string-interview-questions-and-answers/#8_Find_all_substrings_of_String_in_java)
* [9. Find length of String without using any inbuilt method in java?](https://java2blog.com/java-string-interview-questions-and-answers/#9_Find_length_of_String_without_using_any_inbuilt_method_in_java)
* [10. Write a java program to print all permutations of String in java?](https://java2blog.com/java-string-interview-questions-and-answers/#10_Write_a_java_program_to_print_all_permutations_of_String_in_java)
* [11. What is the difference between creating String as new() and literal?](https://java2blog.com/java-string-interview-questions-and-answers/#11_What_is_the_difference_between_creating_String_as_new_and_literal)
* [12. How many objects will be created in below code?](https://java2blog.com/java-string-interview-questions-and-answers/#12_How_many_objects_will_be_created_in_below_code)
* [13. How do you convert String to char array in java?](https://java2blog.com/java-string-interview-questions-and-answers/#13_How_do_you_convert_String_to_char_array_in_java)
* [14. What is difference between StringBuffer and StringBuilder in java?](https://java2blog.com/java-string-interview-questions-and-answers/#14_What_is_difference_between_StringBuffer_and_StringBuilder_in_java)
* [15. How many objects will be created in below code?](https://java2blog.com/java-string-interview-questions-and-answers/#15_How_many_objects_will_be_created_in_below_code)

In this post, we will see interview questions on java String. String is most important data type which we use in our programs very often.

## 1. Why String is declared final or immutable in java?

There are various reasons to make String immutable.

* 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.

## 2. How to reverse a String in java? Can you write a program without using any java inbuilt methods?

There are many ways to do it, some of them are:

* Using for loop
* Using recursion
* Using StringBuffer

Please refer to the solution at [reverse a String in java](https://www.java2blog.com/2015/08/java-program-to-reverse-string.html)

## 3. How to check if two Strings are anagram in java?

4.3M

Linux File Permissions Commands with Examples - Linux Tutorial

Anagrams means if two String have same characters but in different order. For example: Angel and Angel are anagrams  
There are many ways to check if Strings are anagrams. Some of them are:

1. Using String methods
2. Using array.sort

Please refer to solution at [check if two Strings are anagram in java.](https://www.java2blog.com/2016/05/check-if-two-strings-are-anagrams-in-java-example-program.html)

## 4. How to check if String has all unique characters in java?

There are multiple ways to find if String has all unique characters or not.

* By using [HashSet](https://java2blog.com/how-hashset-works-in-java/ “HashSet”)
* Using indexOf and [lastIndexOf](https://java2blog.com/java-string-lastindexof-example/ “lastIndexOf”) methods of String
* By Using ascii value of characters.

Please refer to complete solution at [check if String has all unique characters](https://www.java2blog.com/2016/06/check-if-string-has-all-unique-characters-in-java.html).

## 5. How to check if one String is rotation of another String in java?

Lets say you need to check whether str1 and str2 is rotation of one another or not.

1. Create a new String with str3= str1 + str1
2. Check if str3 [contains](https://www.java2blog.com/2016/05/java-string-contains-example.html) str2 or not.
3. if str3 [contains](https://www.java2blog.com/2016/05/java-string-contains-example.html) str2 then str2 is rotation of str1 else it is not

You can find complete solution at [check if one String is rotation of another in java](https://www.java2blog.com/2016/05/check-if-one-string-is-rotation-of-another-java.html).

## 6. Write a java program to find duplicate characters in String in java?

1. Create a [HashMap](https://www.java2blog.com/2016/04/hashmap-in-java-with-examples.html) and character of String will be inserted as key and its count as value.
2. If [Hashamap](https://www.java2blog.com/2016/04/hashmap-in-java-with-examples.html) already contains char,increase its count by 1, else put char in HashMap.
3. If value of Char is more than 1, that means it is duplicate character in that String.

Please refer to solution at[program to find duplicate characters in a String](https://www.java2blog.com/2016/05/find-duplicate-characters-in-string-java.html).

## 7. Find first non repeated character in String in java?

There are may ways to find it.  
Some of them are:

* Using [LinkedHashMap](https://java2blog.com/linkedhashmap-in-java-with-example/ “LinkedHashMap”)
* Using indexOf and lastIndexOf methods.

Please find complete solution at [find first non repeated character in  a String](https://www.java2blog.com/2015/08/find-first-non-repeated-character-in.html).

## 8. Find all substrings of String in java?

Java program to find all substrings of a String.  
For example: If input is “abb”  then output should be “a”, “b”,”b”, “ab”, “bb”, “abb”

We will use String class’s subString method to find all subString.  
Please refer to complete solution at [find all subStrings of String.](https://www.java2blog.com/2015/08/find-all-substrings-of-string-in-java.html)

## 9. Find length of String without using any inbuilt method in java?

You can use try catch block for catching StringIndexOutOfBoundException and when this exception aries, you can simply return i(Index at which you will get the exception)  
Please refer to complete solution at [find length of String without inbuilt methods](https://www.java2blog.com/2015/08/find-length-of-string-without-using.html).

## 10. Write a java program to print all permutations of String in java?

Take out first character of String and insert into different places of permutations of remaining String recursively. Please find complete solution at [how to find all permutations of String in java](https://www.java2blog.com/2016/07/find-all-permutations-of-string-in-java.html).

## 11. 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**("Java2Blog");  **String** str2=**new** **String**("Java2Blog");          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="Java2Blog";  **String** str2="Java2Blog";  System.out.println(str1==str2); |

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

## 12. How many objects will be created in below code?

|  |  |
| --- | --- |
| 1  2  3  4 | **String** str1= "java2blog";  **String** str2= "java2blog"; |

Only one object will be created and will be stored in String constant pool.

## 13. How do you convert String to char array in java?

You can use string’s [toCharArray()](https://www.java2blog.com/2016/05/how-to-convert-string-to-char-array-in.html) method to convert String to char array.

## 14. 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. |

## 15. How many objects will be created in below code?

|  |  |
| --- | --- |
| 1  2  3  4 | **String** str1= **new** **String**("java2blog");  **String** str2= **new** **String**("java2blog"); |

Three objects will be created here, two in heap memory and one in String constant pool.



<=a;a--){const b=g[a].name;0===b.indexOf("data-jcp-")&&(f[b.substring(9)]=g[a].value)} (function(a,b,h){var d=window;a&&b&&h&&d.ReportingObserver&&d.fetch&&(new d.ReportingObserver((e,k)=>{e=e[0];"HeavyAdIntervention"===e?.body?.id&&(d.fetch(`${a}&label=${0<(e.body.message?.indexOf("network")||0)?h:b}`,{keepalive:!0,method:"get",mode:"no-cors"}),k.disconnect())},{types:["intervention"],buffered:!0})).observe()})(f.base\_url,f.cpu\_label,f.net\_label);}).call(this);

# Java Serialization interview questions and answers

Serialization is one of most important concept in java. If you are going to face core java interview, then you might be asked some questions from Serialization.

## Java Serialization Tutorial:

* [Serialization in java](https://www.java2blog.com/?p=393)
* [Java Serialization interview questions and answers](https://www.java2blog.com/?p=14)
* [serialversionuid in java serialization](https://www.java2blog.com/?p=392)
* [externalizable in java](https://www.java2blog.com/?p=375)
* [Transient keyword in java](https://www.java2blog.com/?p=201)
* [Difference between Serializable and Externalizable in Java](https://www.java2blog.com/?p=15)

Here are list of questions that may be asked on Serialization.

#### Question 1: What is Serialization?

**Answer:**  
Java provides mechanism called serialization to persists java objects in a form of ordered or sequence of bytes that includes the object’s data as well as information about the object’s type and the types of data stored in the object.So if we need to serialize any object then it can be read and deserialize it using object’s type and other information so we can retrieve original object.Classes ObjectInputStream and ObjectOutputStream are high-level streams that contain the methods for serializing and deserializing an object.  
ObjectOutputStream has many method for serializing object but commonly used method is

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | private **void** writeObject(ObjectOutputStream os) throws IOException  {    } |

Similarly ObjectInputStream has

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | private **void** readObject(ObjectInputStream **is**) throws IOException, ClassNotFoundException      {        } |

#### Question 2: What is need of Serialization?

**Answer:**  
Serialization is usually used when there is need to send your data over network or to store in files. By data I mean objects and not text.

Now the problem is your Network infrastructure and your Hard disk is hardware components that understand bits and bytes but not Java objects.

Serialization is the translation of Java object’s values/states to bytes to send it over network or to save it.On the other hand, Deserialization is conversion of byte code to corresponding java objects.

#### Question 3: Can you explain about Concept of serialVersionUID?

**Answer:**  
serialVersionUID is used to ensure that same class(That was used during Serialization) is loaded during Deserialization.serialVersionUID is used for version control of object.You can read more at [serialVersionUID in java serialization](https://www.java2blog.com/2013/03/serialversionuid-in-java-serialization.html)

#### Question 4: Is it necessary to implement Serializable interface if you want to serialize any object?

2.4M

How to Upgrade to Linux Mint 20 from 19.3 [Step by Step]

**Answer:**  
Yes, it is necessary to implement Serializable interface if you want to serialize any object. Serializable is marker interface.Marker interface in Java is interfaces with no field or methods or in simple word empty interface in java is called marker interface.

#### Question 5: Can you Serialize static variables?

**Answer :**  
No,you can’t.As you know static variable are at class level not at object level and you serialize a object so you can’t serialize static variables.

#### Question 6: How can you customize serialization process?

**Answer :**  
You can customize Serialization process by defining writeObject and readObject method.Java serialization provides a mechanism such that if you have private methods with particular signature then they will get called during serialization and deserialization, so in this way, we can customize Serialization process.  
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 | private **void** writeObject(ObjectOutputStream os) throws IOException, ClassNotFoundException  {  **try** {     os.defaultWriteObject();     os.writeInt(address.getHomeNo());     os.writeObject(address.getStreet());     os.writeObject(address.getCity());    }  **catch** (Exception e)    { e.printStackTrace(); }  }    private **void** readObject(ObjectInputStream **is**) throws IOException, ClassNotFoundException  {  **try** {  **is**.defaultReadObject();  **int** homeNo=**is**.readInt();  **String** street=(**String**) **is**.readObject();  **String** city=(**String**) **is**.readObject();     address=**new** Address(homeNo,street,city);      } |

One thing should be kept in mind that ObjectInputStream should read data in same sequence in which we have written data to ObjectOutputStream.

#### Question 7: How can you avoid certain member variable of class to be serialized?

**Answer:**  
You can mark that variable as either static or transient. Let’s see a simple example using transient variable.  
**Transient variable** is the variable whose value is not serialized during [serialization](https://www.java2blog.com/?p=393). You will get default value for these variable when you deserialize it.

Let’s say you have Country class and you don’t want to Serialize population attribute as it will change with time, so you can declare population attribute as transient and it won’t serialized any more.

**Transient keyword example:**

Create a classed called **Country.java** as below:

|  |  |
| --- | --- |
| 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  33 | **package** org.arpit.java2blog;    import java.io.Serializable;    public **class** Country **implements** Serializable {    **String** name;  transient **long** population;    public Country() {  **super**();  }  public Country(**String** name, **long** population) {  **super**();  **this**.name = name;  **this**.population = population;  }  public **String** getName() {  **return** name;  }  public **void** setName(**String** name) {  **this**.name = name;  }  public **long** getPopulation() {  **return** population;  }  public **void** setPopulation(**long** population) {  **this**.population = population;  }    } |

Create **serializationMain.java** as below:

|  |  |
| --- | --- |
| 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;  import java.io.FileOutputStream;  import java.io.IOException;  import java.io.ObjectOutputStream;    public **class** SerializeMain {    */\*\**  *\* @author Arpit Mandliya*  *\*/*  public static **void** main(**String**[] args) {      Country india = **new** Country();  india.setName("India");  india.setPopulation(100000);  **try**    {     FileOutputStream fileOut = **new** FileOutputStream("country.ser");     ObjectOutputStream outStream = **new** ObjectOutputStream(fileOut);     outStream.writeObject(india);     outStream.close();     fileOut.close();    }**catch**(IOException i)    {     i.printStackTrace();    }      System.out.println("serialized");  }  } |

When you run above program, you will get below output:

|  |  |
| --- | --- |
| 1  2  3 | serialized |

Now Create a classed called **DeserializeMain.java**as below:

|  |  |
| --- | --- |
| 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  33  34  35  36  37 | **package** org.arpit.java2blog;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.ObjectInputStream;    public **class** DeserializeMain {  */\*\**  *\* @author Arpit Mandliya*  *\*/*  public static **void** main(**String**[] args) {      Country india = **null**;  **try**         {            FileInputStream fileIn =**new** FileInputStream("country.ser");            ObjectInputStream **in** = **new** ObjectInputStream(fileIn);            india = (Country) **in**.readObject();  **in**.close();            fileIn.close();         }**catch**(IOException i)         {            i.printStackTrace();  **return**;         }**catch**(ClassNotFoundException c)         {            System.out.println("Country class not found");            c.printStackTrace();  **return**;         }         System.out.println("Deserialized Country...");         System.out.println("Country Name : " + india.getName());         System.out.println("Population : " + india.getPopulation());    }  } |

When you run above program, you will get below output:

|  |  |
| --- | --- |
| 1  2  3  4  5 | Deserialized Country...  Country Name : India  Population : 0 |

As you can see in above example, we have declared population as transient, so after deserialization, its value became 0 (Default value for long)

#### Question 8: What  if superclass is Serializable?  Does that mean child class is automatically Serializable?

**Answer :** Yes

#### Question 9: What if superclass is Serializable but you don’t want subclass to be Serializable?

**Answer :**If you don’t want subclass to serializable then you need to implement writeObject() and readObject() method and need to throw NotSerializableException from this methods.

#### Question 10 :What is externalizable interface?

**Answer:** As name suggest it is externalilizing your serialization.If you want to customize your serialization mechanism then you can use it.It uses custom written mechanism to perform marshalling and unmarshalling of objects.[Externalizable](https://java2blog.com/externalizable-in-java/ “Externalizable”) interface extends Serializable interface. If you implement this interface then you need to override following methods.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | @Override  public **void** readExternal(ObjectInput arg0) throws IOException,ClassNotFoundException {    }    @Override  public **void** writeExternal(ObjectOutput arg0) throws IOException {    } |

#### Question 11 : What are differences between Serializable and Externalizable in Java?

**Answer :**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Serializable** | **Externalizable** |
| Marker interface | It is marker interface. You don’t have to provide implementation of any method. | Externalizable is not marker interface, you have to override writeExternal and readExternal method. |
| Control | Serializable interface has less control over serialization process and it is optional to override readObject and writeObject. | Externalizable interface has more control over serialization process and it is mandatory to override writeExternal and readExternal. |
| Performance | JVM uses reflection to perform serialization in the case of Serializable interface which is quite slow. | Programmer have to implement readExternal and writeExternal methods but it relatively results in better performance |
| Supersedes | NA | If you implement Externalizable interface and provide implementation of readExternal and writeExternal then it supersedes readObject and writeObject methods in that class. It is due to the fact that Externalizable extends Serializable interface. |
| Constructor called during Deserialization | Default constructor is not called during Deserialization process. |  |

# Method overloading and overriding interview questions in java

In this tutorial, we are going to see [Method overloading](https://www.java2blog.com/2014/06/method-overloading-in-java.html) and [overriding](https://www.java2blog.com/2014/06/method-overriding-in-java.html) interview questions.

Table of Contents [[hide](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/)]

* [1. What is method overloading?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#1_What_is_method_overloading)
* [2. What are rules of method overloading?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#2_What_are_rules_of_method_overloading)
* [3. Can we overload static methods in java?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#3_Can_we_overload_static_methods_in_java)
* [4. Can you overload main method?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#4_Can_you_overload_main_method)
* [5. Can we change only return type while method overloading?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#5_Can_we_change_only_return_type_while_method_overloading)
* [6. What is method overriding?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#6_What_is_method_overriding)
* [7. What are rules of method overriding?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#7_What_are_rules_of_method_overriding)
* [8.  Can you override static methods in java?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#8_Can_you_override_static_methods_in_java)
* [9. Can you override private methods in java?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#9_Can_you_override_private_methods_in_java)
* [10. Can you override final methods?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#10_Can_you_override_final_methods)
* [11. What is static binding?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#11_What_is_static_binding)
* [12. What is dynamic binding?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#12_What_is_dynamic_binding)
* [13. What are Covariant return type in java?](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#13_WhatareCovariant_return_type_in_java)
* [14. Predict output of below program:](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#14_Predict_output_of_below_program)
* [15. Predict output of below program:](https://java2blog.com/method-overloading-and-overriding-interview-questions-in-java/#15_Predict_output_of_below_program)

### 1. What is method overloading?

**Answer:**  
If two or more methods have same name, but different argument then it is called method overloading.  
**For example:**  
Array’s sort method have many overloaded versions. You can sort array of double, int, String etc.

### 2. What are rules of method overloading?

80.5K

Barbie joins the Metaverse

**Next**

**Stay**

Rules of Method overloading:

|  |  |
| --- | --- |
| **Number of Arguments** | Overloaded method can have different number of arguments |
| **Date type** | Overload method can have different data type for argument |
| **Return type** | Return type can be changed but either number of argument or data type of argument should also be changed. |
| **Order of arguments** | If you change sequence of arguments then it is also valid method overloading provided you have different data types arguments. |
| **Constructor** | Can be overloaded |

### 3. Can we overload [static](https://www.java2blog.com/2013/12/static-keyword-in-java.html) methods in java?

**Answer:**

Yes, we can overload static methods in java but we can not override them.

### 4. Can you overload main method?

**Answer:**  
Yes, you can [overload main method in java](https://www.java2blog.com/method-overloading-in-java/) but only method with signature [public static void main(String[] args)](https://java2blog.com/public-static-void-main-string-args-java-main-method/) will be used when your class is invoked by JVM.

### 5. Can we change only return type while method overloading?

**Answer:**  
You can not.If we change only return type, it will become ambiguous for compiler to figure out which method to call.That is why you can not change only return type.

### 6. What is method overriding?

**Answer:**  
If subclass is having same method as base class then it is known as method overriding Or in another words, If subclass provides specific implementation to any method which is present in its one of parents classes then it is known as method overriding.

### 7. What are rules of method overriding?

**Rules for method overriding:**

|  |  |
| --- | --- |
| **Arguments** | Must not change |
| **Return type** | Can’t change except for covariant (subtype) returns |
| **Access Modifier** | Must not be more restrictive. Can be less restrictive. |
| **Exceptions** | Can reduce or eliminate but must not throw new/broader checked exceptions |
| **Contructor** | Can not be overridden |
| **Static method** | Can not be overridden |
| **final method** | Can not be overridden |

### 8.  Can you override [static](https://www.java2blog.com/2013/12/static-keyword-in-java.html) methods in java?

**Answer:**  
No, you can not override static methods in java. Static methods belongs to class level not at object level.You can create static method with same name in child class and it won’t give you compilation error but it is called method hiding. You won’t get overriding behaviour with it.

### 9. Can you override private methods in java?

**Answer:**  
No, you can not override private methods in java. Private methods are not visible to child class, hence you can not override it , you can only hide it.

### 10. Can you override [final](https://www.java2blog.com/2017/03/final-keyword-java-example.html) methods?

**Answer:**  
Because final methods are meant to be not overridden.You declare a method final because you don’t want it to be overridden in subclass.

### 11. What is static binding?

**Answer:**  
When you compile Java program. During compilation process, compiler bind method call to actual method. This is called static binding and method overloading binding happens at compile time.

### 12. What is dynamic binding?

**Answer:**  
Binding of overridden methods happen at runtime is known as dynamic binding.

### 13. What are 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.

### 14. Predict output of below program:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | public **class** MethodOverloadingExample {    public **void** methodTest(**Object** **object**)  {    System.out.println("Calling object method");  }    public **void** methodTest(**String** **object**)  {    System.out.println("Calling String method");  }    public static **void** main(**String** args[])  {    MethodOverloadingExample moe=**new** MethodOverloadingExample();    moe.methodTest(**null**);  }  } |

**Output:**

|  |  |
| --- | --- |
| 1  2  3 | Calling **String** method |

**Explanation:**  
When we have two overloaded version of same method, JVM will always call most specific method.

### 15. Predict output of below program:

|  |  |
| --- | --- |
| 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 | import java.io.IOException;    public **class** MethodOverrdingTestMain {  public static **void** main(**String**[] args) {    B b=**new** B();  **try** {     b.method();    } **catch** (Exception e) {     e.printStackTrace();    }  }    }  **class** A{    public **void** method() throws IOException  {    }  }  **class** B **extends** A{    public **void** method() throws Exception  {    }  } |

**Output:**

|  |  |
| --- | --- |
| 1  2  3 | compile time error |