CAREERHUNT

Careerhunt is a broad-based career highway for IT professionals. This is where people meet peers, mentors, recruiters and trainers who are difficult to access otherwise.

Few Interview Questions on JAVA

by admin | category Java | No Comments

Java is the versatile languages for programming. That can be used for machine learning but the problem with these languages is that they make implementing machine learning very tough because of very less inbuilt functions and classes to implement the mathematical functions.

Here we have listed the questions can be asked on Java in interview.

1. What is Abstract Class?
2. What is Interface?
3. What is the difference between Interface and Abstract Class?
4. What is the main difference between arraylist and vector?
5. Difference between hashmap and hashtable ?
6. Difference between string and string buffer classes ?
7. Explain how hashmap works?
8. What is the size of empty class in java ?
9. What is difference between shallow comparision and deep comparision ?
10. What is difference between final,finally,finalize ?
11. What do you know about java garbage collector?
12. When an Object becomes Eligible for Garbage Collection?

Q1.What is Abstract Class?

An *abstract class* is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

An *abstract method* is a method that is declared without an implementation (without braces, and followed by a semicolon), like this:

abstract void moveTo(double deltaX, double deltaY);

If a class includes abstract methods, then the class itself *must* be declared abstract, as in:

```
public abstract class GraphicObject {

  // declare fields

  // declare nonabstract methods

  abstract void draw();

}
```

When an abstract class is subclassed, the subclass usually provides implementations for all of the abstract methods in its parent class. However, if it does not, then the subclass must also be declared abstract.

Q2.What is Interface?

An **interface in java** is a blueprint of a class. It has static constants and abstract methods.

The interface in java is **a mechanism to achieve abstraction**. There can be only abstract methods in the java interface not method body. It is used to achieve abstraction and multiple inheritance in Java.

Java Interface also **represents IS-A relationship**.

Recent Posts

It cannot be instantiated just like abstract class.

Why use Java interface?

There are mainly three reasons to use interface. They are given below.

- It is used to achieve abstraction.
- By interface, we can support the functionality of multiple inheritance.
- It can be used to achieve loose coupling.

Q3.What is the difference between Interface and Abstract Class?

1. **Type of methods:** Interface can have only abstract methods. Abstract class can have abstract and non-abstract methods. From Java 8, it can have default and static methods also.
2. **Final Variables:** Variables declared in a Java interface are by default final. An abstract class may contain non-final variables.
3. **Type of variables:** Abstract class can have final, non-final, static and non-static variables. Interface has only static and final variables.
4. **Implementation:** Abstract class can provide the implementation of interface. Interface can't provide the implementation of abstract class.
5. **Inheritance v/s Abstraction:** A Java interface can be implemented using keyword "implements" and abstract class can be extended using keyword "extends".
6. **Multiple implementations:** An interface can extend another Java interface only; an abstract class can extend another Java class and implement multiple Java interfaces.
7. **Accessibility of Data Members:** Members of a Java interface are public by default. A Java abstract class can have class members like private, protected, etc.

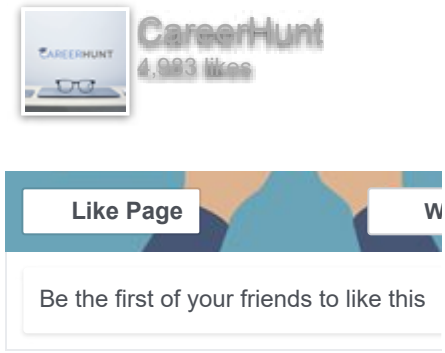
Q4.What is the main difference between arraylist and vector?

- Different JOB Roles for Computer Science Students in Software industries
- What are the different Job roles of a Python Developer
- Cyber Security – Career Opportunities and Certifications

Archives

- April 2018
- March 2018
- February 2018
- January 2018
- December 2017
- April 2017

Find us on Facebook



ArrayList and Vector both use Array as a data structure internally. However there are few differences in the way they store and process the data.

ArrayList V/S Vector:

1) Synchronization:

ArrayList is non-synchronized which means multiple threads can work on ArrayList at the same time. while Vector is synchronized. This means if one thread is working on Vector, no other thread can get a hold of it. Unlike ArrayList, only one thread can perform an operation on vector at a time.

2) Resize:

Both ArrayList and Vector can grow and shrink dynamically to maintain the optimal use of storage; however the way they resized is different. ArrayList grow by half of its size when resized while Vector doubles the size of itself by default when grows.

3) Performance:

Vector is slow as it is thread safe . In comparison ArrayList is fast as it is non-synchronized.

4) Set Increment Size

ArrayList does not define the increment size . Vector defines the increment size .

You can find the following method in Vector Class

public synchronized void setSize(int i) { //some code }

There is no setSize() method or any other method in ArrayList which can manually set the increment size.

5) Enumerator

Other than Hashtable ,Vector is the only other class which uses both Enumeration and Iterator .While

ArrayList can only use Iterator for traversing an ArrayList
.

Q5.Difference between hashmap and hashtable ?

Both HashMap and Hashtable class implements Map interface in java but there are many differences between HashMap and Hashtable in java.There are few differences which are listed below.

1) One of the major differences between HashMap and Hashtable is that HashMap is non-synchronized whereas Hashtable is synchronized, which means Hashtable is thread-safe and can be shared between multiple threads but HashMap cannot be shared between multiple threads without proper synchronization.

2)The HashMap class is roughly equivalent to Hashtable, except that it permits nulls. HashMap allows null values as key and value whereas Hashtable doesn't allow null.

3)The third significant difference between HashMap vs Hashtable is that Iterator in the HashMap is a fail-fast iterator while the enumerator for the Hashtable is not and throw ConcurrentModificationException if any other Thread modifies the map structurally by adding or removing any element except Iterator's own remove() method. But this is not a guaranteed behaviour and will be done by JVM on best effort. This is also an important difference between Enumeration and Iterator in Java.

4)Hashtable is synchronized so it is slow. HashMap is not synchronized so it is fast. HashMap out performs Hashtable in single threaded environment. And ConcurrentHashMap out performs Hashtable in Multi-Threaded environment.

5) Hashtable uses Separate chaining(with linked lists) as collision handling strategy bounding search time of

O(n).As of JAVA 8 HashMap uses Balanced Tree as collision resolution strategy which has bounding search time of O(log n).

Q6.Difference between string and string buffer classes ?

Java provides the StringBuffer and String classes, and the String class is used to manipulate character strings that cannot be changed. Simply stated, objects of type String are read only and immutable.
The StringBuffer class is used to represent characters that can be modified.

The significant performance difference between these two classes is that StringBuffer is faster than String when performing simple concatenations.
In String manipulation code, character strings are routinely concatenated. Using the String class, concatenations are typically performed as follows:

```
String str = new String ("Stanford  ");

str += "Lost!!";
```

If you were to use StringBuffer to perform the same concatenation, you would need code that looks like this:

```
StringBuffer str = new StringBuffer ("Stanford ");

str.append("Lost!!");
```

Developers usually assume that the first example above is more efficient because they think that the second example, which uses the append method for concatenation, is more costly than the first example, which uses the + operator to concatenate two String objects.

The + operator appears innocent, but the code generated produces some surprises. Using a StringBuffer for

concatenation can in fact produce code that is significantly faster than using a String

If the functionality of the String class is desired, consider using a StringBuffer for concatenation and then performing one conversion to String.

Q7.Explain how hashmap works?

HashMap in Java works on hashing principle. It is a data structure which allows us to store object and retrieve it in constant time O(1) provided we know the key. In hashing, hash functions are used to link key and value in HashMap. Objects are stored by calling put(key, value) method of HashMap and retrieved by calling get(key) method.

When we call put method, hashcode() method of the key object is called so that hash function of the map can find a bucket location to store value object, which is actually an index of the internal array, known as the table. HashMap internally stores mapping in the form of Map.

Entry object which contains both key and value object. When you want to retrieve the object, you call the get() method and again pass the key object. This time again key object generates same hash code and we end up at same bucket location.

Q8.What is the size of empty class in java ?

Creating the object means creating the memory, therefore even though there are no members in the class also, it will create a **physical memory location of 1 byte** and gets the address(object is reality) .The size of empty class is one even though no variables are declared in it. As when programmers take address of an object, it should not return invalid address. To avoid these concerns, compiler allocates dummy memory of implementation defined size to class with no data members and member function.

Q9.What is difference between shallow comparision and deep comparision ?

There are two types of object comparison in java – shallow and deep.

when you use '==' operator, you are comparing the references for equality. It means you are comparing memory addresses of the objects. It is called shallow comparison.

When you use .equals() you are comparing the object values themselves for equality. It means you are checking 'do both objects have same value for corresponding instance variables or not '.

Let's have an example to understand it better:

String person1="James"; // memory address 500

String person2="James"; //memory address 500 as the value is same

// Shallow comparison

if(person1== person2){

// It will return true, because it is comparing the memory addresses which are same for both the objects

System.out.println("Success");

}

// Deep comparison

if(person1.equals(person2)){

//It will return true, because it is comparing the object values which are again same for both

System.out.println("Success");

```
}
```

Q10.What is difference between final,finally,finalize ?

final – final keyword can be used with a class, variable or a method. A variable declared as final acts as constant, which means one a variable is declared and assigned , the value cannot be changed.

An object can also be final, which means that the once the object is created it cannot be assigned a different object, although the properties or fields of the object can be changed.

A final class is immutable, which means that no other class can extend from it. E.g String, Integer.

A final method in a class cannot be overridden in the child class.

finally – finally keyword is used with try-catch block for handling exception handling. The finally block is optional in try-catch block. The finally code block is always executed after try or catch block is completed. The general use case for finally block is to close the resources or clean up objects used in try block. For e.g. Closing a FileStream, I/O stream objects, Database connections, HTTP connections are generally closed in a finally block.

finalize() – It is a method (function) which is executed before deleting any object in java. In java we don't need to explicitly declare and define constructor and destructor for a class, they are running in background with help of thread. This thread is of least priority and is executed every time when the purpose of objects made is finished. So if it happens that some important information is linked with the object which is going to be deleted by the destructor, the programmer can write the relevant operation to be done with that object to save the data linked with it before deletion of this object. Code for

such operation is written in **finalize** and this method is called before deleting any object from memory.

Q11.What do you know about java garbage collector?

Garbage collection is a mechanism provided by Java Virtual Machine to reclaim heap space from objects which are eligible for Garbage collection.

Garbage Collection in Java is carried by a daemon thread called Garbage Collector.

Before removing an object from memory garbage collection thread invokes finalize() method of that object and gives an opportunity to perform any sort of clean up required.

You as Java programmer cannot force garbage collection in Java; it will only trigger if JVM thinks it needs a garbage collection based on Java heap size.

There are methods like System.gc() and Runtime.gc() which is used to send request of Garbage collection to JVM but it's not guaranteed that garbage collection will happen.

Q12.When an Object becomes Eligible for Garbage Collection?

An object becomes eligible for Garbage collection or GC if it's not reachable from any live threads or by any static references. In other words, you can say that an object becomes eligible for garbage collection if it's all references are null. Cyclic dependencies are not counted as the reference so if object A has a reference to object B and object B has a reference to Object A and they don't have any other live reference then both Objects A and B will be eligible for Garbage collection.

Generally, an object becomes eligible for garbage collection in Java on following cases:

1) All references to that object explicitly set to null e.g. object = null

2) The object is created inside a block and reference goes out scope once control exit that block.

3) Parent object set to null if an object holds the reference to another object and when you set container object's reference null, child or contained object automatically becomes eligible for garbage collection.

4) If an object has only lived weak references via WeakHashMap it will be eligible for garbage collection.

Related Posts

12 Best Resources to Learn Java
by admin

Scripting Languages on JVM
by admin

About Author

admin