# Accolite

* **1. Write own implementation of Linked Hash Map**
* **2. Merge two sorted linked List**
* Node MergeLists(Node list1, Node list2) {
* if (list1 == null) return list2;
* if (list2 == null) return list1;
* if (list1.data < list2.data) {
* list1.next = MergeLists(list1.next, list2);
* return list1;
* } else {
* list2.next = MergeLists(list2.next, list1);
* return list2;
* }
* }

**Find common elements in Set**

// union

Set<String> union = new HashSet<String>(h1.keySet());

union.addAll(h2.keySet());

// h1 - h2

Set<String> diff1 = new HashSet<String>(h1.keySet());

diff1.removeAll(h2.keySet());

// h2 - h1

Set<String> diff2 = new HashSet<String>(h2.keySet());

diff1.removeAll(h1.keySet());

// intersection

Set<String> intersect = new HashSet<String>(h1.keySet());

intersect.retainAll(h2.keySet());

Map<String, String> m0 = new HashMap<String, String>();

m0.put("a", "a");

m0.put("b", "b");

Map<String, String> m1 = new HashMap<String, String>();

m1.put("c", "c");

m1.put("b", "b");

Set<String> s = new HashSet<String>(m0.keySet());

s.retainAll(m1.keySet());

System.out.println(s);

* 3. How to make class immutable and why we need to make string variables inside immutable class as final even string is already immutable
* Write Thread safe singleton class
* write code for even publisher and subscriber  [Answer Question](https://www.glassdoor.co.in/Interview/write-code-for-even-publisher-and-subscriber-QTN_1732272.htm)
* write a code to write into text file in asynchronous environment  [Answer Question](https://www.glassdoor.co.in/Interview/write-a-code-to-write-into-text-file-in-asynchronous-environment-QTN_1732273.htm)
* suppose that you have many different data sources like .ini, .xml, txt, Oracle, excel. design interface based code with design pattern to read write and insert into different data sources  [1 Answer](https://www.glassdoor.co.in/Interview/suppose-that-you-have-many-different-data-sources-like-ini-xml-txt-Oracle-excel-design-interface-based-code-with-des-QTN_1732274.htm)
* concept of overriding and overloading with subtyping instance based  [Answer Question](https://www.glassdoor.co.in/Interview/concept-of-overriding-and-overloading-with-subtyping-instance-based-QTN_1732275.htm)
* thread synchronization with monitor, lock, read writer lock  [Answer Question](https://www.glassdoor.co.in/Interview/thread-synchronization-with-monitor-lock-read-writer-lock-QTN_1732276.htm)
* concept of deadlock  [Answer Question](https://www.glassdoor.co.in/Interview/concept-of-deadlock-QTN_1732277.htm)
* design patterns like factory, strategy, singleton  [Answer Question](https://www.glassdoor.co.in/Interview/design-patterns-like-factory-strategy-singleton-QTN_1732279.htm)
* database table design and some queries  [Answer Question](https://www.glassdoor.co.in/Interview/database-table-design-and-some-queries-QTN_1732280.htm)
* concept of dispatcher and background worker  [Answer Question](https://www.glassdoor.co.in/Interview/concept-of-dispatcher-and-background-worker-QTN_1732281.htm)
* indexing in database table  [Answer Question](https://www.glassdoor.co.in/Interview/indexing-in-database-table-QTN_1732282.htm)

xor all the array of element with all the elements in array having 1 to n element

in an n\*n matrix, filled with integers, we have to find the minimum number in the shortest time. The matrix is sorted on both column wise and row wise.

shortest path sum of a binary tree :.

If a thread throws an exception, How do you restart a thread again.

As we are already aware that there are two kinds of exceptions in Java. Checked exceptions and Unchecked exceptions. Checked exceptions must be specified in the throws clause of a method or caught inside them. Unchecked exceptions don’t have to be specified or caught. When a checked exception is thrown inside the run() method of a Thread object, we have to catch and treat it accordingly, because the run() method doesn’t accept a throws clause. But when an unchecked exception is thrown inside the run() method of a Threadobject, the default behavior is to write the stack trace in the console (or log it inside error log file) and exit the program.

Fortunately, Java provides us with a mechanism to catch and treat the unchecked exceptions thrown in a Thread object to avoid the program ending. This can be done using UncaughtExceptionHandler.

Let’s take an example of UncaughtExceptionHandler usage. In this example, we have created a thread which tries to parse few strings which are supposed to be integers. We have written the run() method such that it throws a “java.lang.NumberFormatException” during it’s execution. As program does not try to catch this exception, exception floats through JVM level and thread gets killed. This is absolutely normal behavior but it MAY NOT be desired behavior.

Without Using UncaughtExceptionHandler

In real life application, you would like to try more than once to perform a critical task even if it failed couple of times. Our example below demonstrate the usecase, first without use of UncaughtExceptionHandler; which causes the thread to die immediately after failure.

**Task.java**

|  |
| --- |
| class Task implements Runnable  {     @Override     public void run()     {        System.out.println(Integer.parseInt("123"));        System.out.println(Integer.parseInt("234"));        System.out.println(Integer.parseInt("345"));        System.out.println(Integer.parseInt("XYZ")); //This will cause NumberFormatException        System.out.println(Integer.parseInt("456"));     }  } |

**DemoThreadExample.java**

|  |
| --- |
| public class DemoThreadExample  {     public static void main(String[] args)     {        Task task = new Task();        Thread thread = new Thread(task);        thread.start();     }  } |

Below is output we get when we run the thread:

123

234

345

Exception in thread "Thread-0" java.lang.NumberFormatException: For input string: "XYZ"

at java.lang.NumberFormatException.forInputString(Unknown Source)

at java.lang.Integer.parseInt(Unknown Source)

at java.lang.Integer.parseInt(Unknown Source)

at examples.algorithms.sleepingbarber.Task.run(DemoThreadExample.java:24)

at java.lang.Thread.run(Unknown Source)

After Using UncaughtExceptionHandler

Let’s add one UncaughtExceptionHandler implementation to catch any unchecked exception during runtime.

**ExceptionHandler.java**

|  |
| --- |
| class ExceptionHandler implements UncaughtExceptionHandler  {     public void uncaughtException(Thread t, Throwable e)     {        System.out.printf("An exception has been captured\n");        System.out.printf("Thread: %s\n", t.getId());        System.out.printf("Exception: %s: %s\n", e.getClass().getName(), e.getMessage());        System.out.printf("Stack Trace: \n");        e.printStackTrace(System.out);        System.out.printf("Thread status: %s\n", t.getState());        new Thread(new Task()).start();     }  } |

Now add this exception handler to the thread.

|  |
| --- |
| class Task implements Runnable  {     @Override     public void run()     {        Thread.currentThread().setUncaughtExceptionHandler(new ExceptionHandler());        System.out.println(Integer.parseInt("123"));        System.out.println(Integer.parseInt("234"));        System.out.println(Integer.parseInt("345"));        System.out.println(Integer.parseInt("XYZ")); //This will cause NumberFormatException        System.out.println(Integer.parseInt("456"));     }  } |

Now run the above example once again. This will run continuously. In real life, if this task is able to complete it’s task then it will exit without throwing any exception and will complete it’s life cycle.

What is size of empty class?

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)

I understand what you want to know. Even an emply class extends java.lang.Object. Therefore, there is no such thing as empty object. There are several articles on how to do it. They all essentially create lot of object of a type and then run GC, finalizers repeatedly until the heap memory reach to a stable value. You can see more on this here and here. For instace the java.lang.Object takes 8 bytes of memory and 4-5 instructions to create.

* You are given an array with negative and positive unsorted numbers. Find out a pair whose sum = 0. If no such pair exists, find the pair whose sum is the closest to 0.

// Prints the pair with sum cloest to x

    static void printClosest(int arr[], int n, int x)

    {

        int res\_l=0, res\_r=0;  // To store indexes of result pair

          // Initialize left and right indexes and difference between

        // pair sum and x

        int l = 0, r = n-1, diff = Integer.MAX\_VALUE;

          // While there are elements between l and r

        while (r > l)

        {

            // Check if this pair is closer than the closest pair so far

            if (Math.abs(arr[l] + arr[r] - x) < diff)

            {

               res\_l = l;

               res\_r = r;

               diff = Math.abs(arr[l] + arr[r] - x);

            }

              // If this pair has more sum, move to smaller values.

            if (arr[l] + arr[r] > x)

               r--;

            else // Move to larger values

               l++;

        }

public static void main(String args[])

    {

        int[] arr = { 1, 5, 7, -1, 5 };

        int sum = 6;

        getPairsCount(arr, sum);

    }

    // Prints number of pairs in arr[0..n-1] with sum equal

    // to 'sum'

    public static void getPairsCount(int[] arr, int sum)

    {

        int count = 0;// Initialize result

        // Consider all possible pairs and check their sums

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

            for (int j = i + 1; j < arr.length; j++)

                if ((arr[i] + arr[j]) == sum)

                    count++;

        System.out.printf("Count of pairs is %d",count);

    }

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

// Prints the pair with sum cloest to x

    static void printClosest(int arr[], int n, int x)

    {

        int res\_l=0, res\_r=0;  // To store indexes of result pair

        // Initialize left and right indexes and difference between

        // pair sum and x

        int l = 0, r = n-1, diff = Integer.MAX\_VALUE;

        // While there are elements between l and r

        while (r > l)

        {            // Check if this pair is closer than the closest pair so far

            if (Math.abs(arr[l] + arr[r] - x) < diff)

            {

               res\_l = l;

               res\_r = r;

               diff = Math.abs(arr[l] + arr[r] - x);

            }

            // If this pair has more sum, move to smaller values.

            if (arr[l] + arr[r] > x)

               r--;

            else // Move to larger values

               l++;

        }

    System.out.println(" The closest pair is "+arr[res\_l]+" and "+ arr[res\_r]);}

* He then modified the problem and made it to find the pair of elements with largest possible positive product.  [1 Answer](https://www.glassdoor.co.in/Interview/He-then-modified-the-problem-and-made-it-to-find-the-pair-of-elements-with-largest-possible-positive-product-QTN_2220925.htm)
* You are given an array of strings, you need to find out the longest chain such that the last character of the k-th element of the chain and the first character of the k+1th element are the same. Similarly, he last character of the last element should be equal to the first character of the first element. He told me to just explain my approach and not write the whole… Show More
* Given an array of positive integers, how would you break it into two arrays such that the sum of elements in both the arrays is same.

1. Design a tic tac toe game algorithm with all corner cases  
2. Find the maximum sum rectangle in a 2D array ( I gave the brute force and dp solution )  
3. Apply DFS on a graph using adjacency list  
4. Coin Change Problem ( Initially I gave recursive solution which had overlapping subproblems but then I gave him the DP solution and he accepted it.  
5. Print the largest sinusoidal sequence( ZIG ZAG sequence) in an array ( He asked me to code the recursive solution and then write the working code of DP solution)

* row wise column wise sorted matrix. find kth smallest number  [Answer Question](https://www.glassdoor.co.in/Interview/row-wise-column-wise-sorted-matrix-find-kth-smallest-number-QTN_2232145.htm)
* longest palindrome that is possible from a given string  [Answer Question](https://www.glassdoor.co.in/Interview/longest-palindrome-that-is-possible-from-a-given-string-QTN_2232146.htm)
* split an array into two parts such that sum of one part is equal to sum of other part

Nth element from the end of a linked list.  
Majority element in a sorted array.  
Design question: Design a train seat booking system. Mention the data structures and the complexities involved.

What is Atomic Object?  
Why Java follows polymorphism.  
How you will configure Global Transaction in Hibernate.

# Pramati Technologies

What is tree structure?

Tree represents the nodes connected by edges. We will discuss binary tree or binary search tree specifically.

Binary Tree is a special datastructure used for data storage purposes. A binary tree has a special condition that each node can have a maximum of two children. A binary tree has the benefits of both an ordered array and a linked list as search is as quick as in a sorted array and insertion or deletion operation are as fast as in linked list.



Important Terms

Following are the important terms with respect to tree.

Path − Path refers to the sequence of nodes along the edges of a tree.

Root − The node at the top of the tree is called root. There is only one root per tree and one path from the root node to any node.

Parent − Any node except the root node has one edge upward to a node called parent.

Child − The node below a given node connected by its edge downward is called its child node.

Leaf − The node which does not have any child node is called the leaf node.

Subtree − Subtree represents the descendants of a node.

Visiting − Visiting refers to checking the value of a node when control is on the node.

Traversing − Traversing means passing through nodes in a specific order.

Levels − Level of a node represents the generation of a node. If the root node is at level 0, then its next child node is at level 1, its grandchild is at level 2, and so on.

keys − Key represents a value of a node based on which a search operation is to be carried out for a node.

Binary Search Tree Representation

Binary Search tree exhibits a special behavior. A node's left child must have a value less than its parent's value and the node's right child must have a value greater than its parent value.



We're going to implement tree using node object and connecting them through references.

Tree Node

The code to write a tree node would be similar to what is given below. It has a data part and references to its left and right child nodes.

struct node {

int data;

struct node \*leftChild;

struct node \*rightChild;

};

In a tree, all nodes share common construct.

BST Basic Operations

The basic operations that can be performed on a binary search tree data structure, are the following −

Insert − Inserts an element in a tree/create a tree.

Search − Searches an element in a tree.

Preorder Traversal − Traverses a tree in a pre-order manner.

Inorder Traversal − Traverses a tree in an in-order manner.

Postorder Traversal − Traverses a tree in a post-order manner.

We shall learn creating (inserting into) a tree structure and searching a data item in a tree in this chapter. We shall learn about tree traversing methods in the coming chapter.

Insert Operation

The very first insertion creates the tree. Afterwards, whenever an element is to be inserted, first locate its proper location. Start searching from the root node, then if the data is less than the key value, search for the empty location in the left subtree and insert the data. Otherwise, search for the empty location in the right subtree and insert the data.

Algorithm

If root is NULL

then create root node

return

If root exists then

compare the data with node.data

while until insertion position is located

If data is greater than node.data

goto right subtree

else

goto left subtree

endwhile

insert data

end If

Implementation

The implementation of insert function should look like this −

void insert(int data) {

struct node \*tempNode = (struct node\*) malloc(sizeof(struct node));

struct node \*current;

struct node \*parent;

tempNode->data = data;

tempNode->leftChild = NULL;

tempNode->rightChild = NULL;

//if tree is empty, create root node

if(root == NULL) {

root = tempNode;

} else {

current = root;

parent = NULL;

while(1) {

parent = current;

//go to left of the tree

if(data < parent->data) {

current = current->leftChild;

//insert to the left

if(current == NULL) {

parent->leftChild = tempNode;

return;

}

}

//go to right of the tree

else {

current = current->rightChild;

//insert to the right

if(current == NULL) {

parent->rightChild = tempNode;

return;

}

}

}

}

}

Search Operation

Whenever an element is to be searched, start searching from the root node, then if the data is less than the key value, search for the element in the left subtree. Otherwise, search for the element in the right subtree. Follow the same algorithm for each node.

Algorithm

If root.data is equal to search.data

return root

else

while data not found

If data is greater than node.data

goto right subtree

else

goto left subtree

If data found

return node

endwhile

return data not found

end if

The implementation of this algorithm should look like this.

struct node\* search(int data) {

struct node \*current = root;

printf("Visiting elements: ");

while(current->data != data) {

if(current != NULL)

printf("%d ",current->data);

//go to left tree

if(current->data > data) {

current = current->leftChild;

}

//else go to right tree

else {

current = current->rightChild;

}

//not found

if(current == NULL) {

return NULL;

}

return current;

}

}

**Linked list loop, reversal.**

**Immutable and singleton class.  
Design Patterns basic.  
Project discussion in detail.  
Garbage collector and type of heap memory discussion.**

**Java 8 new features  
What is the need of setter and getter methods**

|  |  |
| --- | --- |
| accepted | **There are actually *many good reasons* to consider using accessors** rather than directly exposing fields of a class - beyond just the argument of encapsulation and making future changes easier.  *Here are the some of the reasons I am aware of:*   * Encapsulation of behavior associated with getting or setting the property - this allows additional functionality (like validation) to be added more easily later. * Hiding the internal representation of the property while exposing a property using an alternative representation. * Insulating your public interface from change - allowing the public interface to remain constant while the implementation changes without affecting existing consumers. * Controlling the lifetime and memory management (disposal) semantics of the property - particularly important in non-managed memory environments (like C++ or Objective-C). * Providing a debugging interception point for when a property changes at runtime - debugging when and where a property changed to a particular value can be quite difficult without this in some languages. * Improved interoperability with libraries that are designed to operate against property getter/setters - Mocking, Serialization, and WPF come to mind. * Allowing inheritors to change the semantics of how the property behaves and is exposed by overriding the getter/setter methods. * Allowing the getter/setter to be passed around as lambda expressions rather than values. * Getters and setters can allow different access levels - for example the get may be public, but the set could be protected. |

**Application Server vs Web Server**

1. Application Server supports distributed transaction and EJB. While Web Server only supports Servlets and JSP.

2. Application Server can contain web server in them. most of App server e.g. JBoss or WAS has Servlet and JSP container.

3. Though its not limited to Application Server but they used to provide services like Connection pooling, Transaction management, messaging, clustering, load balancing and persistence. Now Apache tomcat also provides connection pooling.

4. In terms of logical difference between web server and application server. web server is supposed to provide http protocol level service while application server provides support to web service and expose business level service e.g. EJB.

5. Application server are more heavy than web server in terms of resource utilization.

**Why all services are singleton in java**

Hi, We used the singleton pattern in our utility layers which consist of Loggging, Caching, Service host repositories, Load Balancer... If the question is on how we arrived to the design. There was a performance lag on the utility layer eg, Logging, on diagnosing we observed that there are several instance getting created which are not required in my case. So we adopted Singleton pattern

In c# a static class cannot implement an interface. When a single instance class needs to implement an interface for a business contracts or IoC purposes, this is where I use the Singleton pattern without a static class.

Also I think Singleton provides a way to maintain state in stateless scenarios

* 1. When the class is stateless eg: Validator classes instead of using static methods for validation because we cant mock static methods so testing becomes difficult. instead if we have singleton we can mock the instance itself.  
     2. When  we need only one state of class at any given point of time. In this case we have to take care of synchronization. **How to handle java heap memory**

In Java, all objects are stored in the heap. They are allocated using [new](http://www.geeksforgeeks.org/new-operator-vs-newinstance-method-java/) operator. The OutOfMemoryError Exception in Java looks like this:

**Exception in thread "main" java.lang.OutOfMemoryError: Java heap space**

Usually, this error is thrown when the Java Virtual Machine cannot allocate an object because it is out of memory, and no more memory could be made available by the garbage collector.

**OutOfMemoryError** usually means that you’re doing something wrong, either holding onto objects too long, or trying to process too much data at a time. Sometimes, it indicates a problem that’s out of your control, such as a third-party library that caches strings, or an application server that doesn’t clean up after deploys. And sometimes, it has nothing to do with objects on the heap.

The **java.lang.OutOfMemoryError exception** can also be thrown by native library code when a native allocation cannot be satisfied (for example, if swap space is low). Let us understand various cases when the OutOfMemory error might occur.

**Symptom or Root cause?**

To find the cause, the text of the exception includes a detailed message at the end. Lets examine all the errors.

**Error 1 – Java heap space :**This error arises due to the applications that make excessive use of finalizers. If a class has a finalize method, then objects of that type do not have their space reclaimed at [garbage collection](http://www.geeksforgeeks.org/garbage-collection-java/) time. Instead, after garbage collection, the objects are queued for finalization, which occurs at a later time. Implementation:

finalizers are executed by a daemon thread that services the finalization queue.

If the finalizer thread cannot keep up, with the finalization queue, then the Java heap could fill up and this type of OutOfMemoryError exception would be thrown.

The problem can also be as simple as a configuration issue, where the specified heap size (or the default size, if it is not specified) is insufficient for the application.

|  |
| --- |
| // Java program to illustrate  // Heap error  import java.util.\*;    public class Heap {      static List<String> list = new ArrayList<String>();    public static void main(String args[]) throws Exception      {          Integer[] array = new Integer[10000 \* 10000];      }  } |

Run on IDE

When you execute the above code above you might expect it to run forever without any problems. As a result, over time, with the leaking code constantly used, the “cached” results end up consuming a lot of Java heap space and when the leaked memory fills all of the available memory in the heap region and Garbage Collection is not able to clean it, the **java.lang.OutOfMemoryError:Java heap space** is thrown.

**Prevention :**Check how to monitor objects for which finalization is pending in [Monitor the Objects Pending Finalization.](https://docs.oracle.com/javase/8/docs/technotes/guides/troubleshoot/memleaks004.html#CIHCDBJB)

**Error 2 – GC Overhead limit exceeded :** This error indicates that the [garbage collector](http://www.geeksforgeeks.org/garbage-collection-java/) is running all the time and Java program is making very slow progress. After a garbage collection, if the Java process is spending more than approximately 98% of its time doing garbage collection and if it is recovering less than 2% of the heap and has been doing so far the last 5 (compile time constant) consecutive garbage collections, then a **java.lang.OutOfMemoryError** is thrown.  
This exception is typically thrown because the **amount of live data barely fits into the Java heap** having little free space for new allocations.

|  |
| --- |
| // Java program to illustrate  // GC Overhead limit exceeded  import java.util.\*;    public class Wrapper {  public static void main(String args[]) throws Exception      {          Map m = new HashMap();          m = System.getProperties();          Random r = new Random();          while (true) {              m.put(r.nextInt(), "randomValue");          }      }  } |

Run on IDE

If you’ll run this program with **java -Xmx100m -XX:+UseParallelGC Wrapper** then the output will be something like this :

Exception in thread "main" java.lang.OutOfMemoryError: GC overhead limit exceeded

at java.lang.Integer.valueOf(Integer.java:832)

at Wrapper.main(error.java:9)

**Prevention :**Increase the heap size and turn off it with the command line flag **-XX:-UseGCOverheadLimit.**

**Error 3 – Permgen space is thrown :** Java memory is separated into different regions. The size of all those regions, including the permgen area, is set during the JVM launch. If you do not set the sizes yourself, platform-specific defaults will be used.  
The **java.lang.OutOfMemoryError**: PermGen space error indicates that the Permanent Generation’s area in memory is exhausted.

|  |
| --- |
| // Java program to illustrate  // Permgen Space error  import javassist.ClassPool;    public class Permgen {      static ClassPool classPool = ClassPool.getDefault();    public static void main(String args[]) throws Exception      {          for (int i = 0; i < 1000000000; i++) {              Class c = classPool.makeClass(com.saket.demo.Permgen" + i).toClass();              System.out.println(c.getName());          }      }  } |

Run on IDE

In the above sample code, code iterates over a loop and generates classes at run time. Class generation complexity is being taken care of by the [Javassist](http://jboss-javassist.github.io/javassist/) library.  
Running the above code will keep generating new classes and **loading their definitions into Permgen space until the space is fully utilized**and the java.lang.OutOfMemoryError: Permgen space is thrown.  
**Prevention :**When the OutOfMemoryError due to PermGen exhaustion is caused during the application launch, the solution is simple. The application just needs more room to load all the classes to the PermGen area so we just need to increase its size. To do so, alter your application launch configuration and add (or increase if present) the **-XX:MaxPermSize**parameter similar to the following example:

java -XX:MaxPermSize=512m com.saket.demo.Permgen

**Error 4 – Metaspace :** Java class metadata is allocated in native memory. If metaspace for class metadata is exhausted, a **java.lang.OutOfMemoryError** exception with a detail MetaSpace is thrown.  
The amount of metaspace that can be used for class metadata is limited by the parameter MaxMetaSpaceSize, which is specified on the command line. When the amount of native memory needed for a class metadata exceeds MaxMetaSpaceSize, a java.lang.OutOfMemoryError exception with a detail MetaSpace is thrown.

|  |
| --- |
| // Java program to illustrate  // Metaspace error  import java.util.\*;    public class Metaspace {      static javassist.ClassPool cp = javassist.ClassPool.getDefault();    public static void main(String args[]) throws Exception      {          for (int i = 0; i < 100000; i++) {              Class c = cp.makeClass("com.saket.demo.Metaspace" + i).toClass();          }      }  } |

Run on IDE

This code will keep generating new classes and loading their definitions to Metaspace until the space is fully utilized and the java.lang.OutOfMemoryError: Metaspace is thrown. When launched with -XX:MaxMetaspaceSize=64m then on Mac OS X my Java 1.8.0\_05 dies at around 70, 000 classes loaded.

**Prevention :**If **MaxMetaSpaceSize**, has been set on the command-line, increase its value. MetaSpace is allocated from the same address spaces as the Java heap. Reducing the size of the Java heap will make more space available for MetaSpace. This is only a correct trade-off if there is an excess of free space in the Java heap.

**Error 5 – Requested array size exceeds VM limit :** This error indicates that the application attempted to allocate an array that is larger than the heap size. For example, if an application attempts to allocate an array of 1024 MB but the maximum heap size is 512 MB then **OutOfMemoryError** will be thrown with “Requested array size exceeds VM limit”.

|  |
| --- |
| // Java program to illustrate  // Requested array size  // exceeds VM limit error  import java.util.\*;    public class GFG {      static List<String> list = new ArrayList<String>();    public static void main(String args[]) throws Exception      {          Integer[] array = new Integer[10000 \* 10000];      }  } |

Run on IDE

The java.lang.OutOfMemoryError: Requested array size exceeds VM limit can appear as a result of either of the following situations:

Your arrays grow too big and end up having a size between the platform limit and the Integer.MAX\_INT

You deliberately try to allocate arrays larger than 2^31-1 elements to experiment with the limits.

**Error 6 – Request size bytes for reason. Out of swap space? :**This apparent exception occurs when an allocation from the native heap failed and the native heap might be close to exhaustion. The error indicates the size (in bytes) of the request that failed and the reason for the memory request. Usually the reason is the name of the source module reporting the allocation failure, although sometimes it is the actual reason.  
**java.lang.OutOfMemoryError: Out of swap space** error is often caused by operating system level issues, such as:

The operating system is configured with insufficient swap space.

Another process on the system is consuming all memory resources.

**Prevention :**When this error message is thrown, the VM invokes the fatal error handling mechanism (that is, it generates a fatal error log file, which contains useful information about the thread, process, and system at the time of the crash). In the case of native heap exhaustion, the heap memory and memory map information in the log can be useful

**Error 7 : reason stack\_trace\_with\_native\_method :** Whenever this error message(reason stack\_trace\_with\_native\_method) is thrown then a stack trace is printed in which the top frame is a native method, then this is an indication that a native method has encountered an allocation failure. The difference between this and the previous message is that the allocation failure was detected in a Java Native Interface (JNI) or native method rather than in the JVM code.

|  |
| --- |
| // Java program to illustrate  // new native thread error  import java.util.\*;    public class GFG {  public static void main(String args[]) throws Exception      {          while (true) {              new Thread(new Runnable()              {                  public void run()                  {                      try                      {                          Thread.sleep(1000000000);          }          catch (InterruptedException e)          {          }      }              }).start();     }    }  } |

Run on IDE

The exact native thread limit is platform-dependent, for example tests Mac OS X reveal that:

**64-bit Mac OS X 10.9, Java 1.7.0\_45 – JVM dies after #2031 threads have been created**

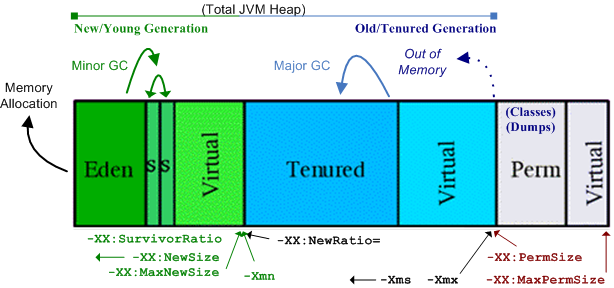
**Prevention :**Use native utilities of the OS to further diagnose the issue. For more information about tools available for various operating systems, see [Native Operating System tools](https://docs.oracle.com/javase/8/docs/technotes/guides/troubleshoot/tooldescr020.html#BABBHHIE).

**Sequential vs data flow diagram**

**, java memory mgmt.**

Garbage Collection:

Garbage collection (GC) is a form of automatic [memory management](http://en.wikipedia.org/wiki/Memory_management). The garbage collector, or just collector, attempts to reclaim [garbage](http://en.wikipedia.org/wiki/Garbage_(computer_science)), or memory occupied by [objects](http://en.wikipedia.org/wiki/Object_(computer_science)) that are no longer in use by the [program](http://en.wikipedia.org/wiki/Application_software). Garbage collection was invented by [John McCarthy](http://en.wikipedia.org/wiki/John_McCarthy_(computer_scientist)) around 1959 to solve problems in [Lisp](http://en.wikipedia.org/wiki/Lisp_(programming_language)).

[](http://3.bp.blogspot.com/-KLUmCgw0yG0/UWn3DoDAkuI/AAAAAAAAAJ8/CroGLX4wlaM/s1600/java_memory.png)Garbage collection is often portrayed as the opposite of [manual memory management](http://en.wikipedia.org/wiki/Manual_memory_management), which requires the programmer to specify which objects to deallocate and return to the memory system. However, many systems use a combination of approaches, including other techniques such as [stack allocation](http://en.wikipedia.org/wiki/Stack-based_memory_allocation) and [region inference](http://en.wikipedia.org/wiki/Region_inference).

Resources other than memory, such as [network sockets](http://en.wikipedia.org/wiki/Network_socket), database [handles](http://en.wikipedia.org/wiki/Handle_(computing)), user interaction windows, and file and device descriptors, are not typically handled by garbage collection. Methods used to manage such resources, particularly [destructors](http://en.wikipedia.org/wiki/Destructor_(computer_science)), may suffice to manage memory as well, leaving no need for GC. Some GC systems allow such other resources to be associated with a region of memory that, when collected, causes the other resource to be reclaimed; this is called [finalization](http://en.wikipedia.org/wiki/Finalizer). Finalization may introduce complications limiting its usability, such as intolerable latency between disuse and reclaim of especially limited resources, or a lack of control over which thread performs the work of reclaiming.

Memory is divided in mainly 3 areas:

 Young\Eden Generation : for newly created objects.

 Tenured Generation : for old objects which are survived after minor gc.

 Perm Space : for class definition, meta data and string pools.

At initialization, a maximum address space is virtually reserved but not allocated to physical memory unless it is needed. The complete address space reserved for object memory can be divided into the young and tenured generations.The young generation consists of eden and two survivor spaces. Most objects are initially allocated in eden. One survivor space is empty at any time, and serves as the destination of any live objects in eden and the other survivor space during the next copying collection. Objects are copied between survivor spaces in this way until they are old enough to be tenured (copied to the tenured generation). A third generation closely related to the tenured generation is the permanent generation which holds data needed by the virtual machine to describe objects that do not have an equivalence at the Java language level.

Java objects are created in Heap and Heap is divided into three parts or generations of garbage collection in Java as mentioned above. New Generation is divided into three parts known as

  Eden space

  Survivor 1

  Survivor 2 space.

When an object first created in heap its gets created in new generation inside Eden space and after subsequent **Minor Garbage collection** if object survives its gets moved to survivor 1 and then Survivor 2 before **Major Garbage collection moved that object to Old or tenured generation.**  
  
Permanent generation of Heap or Perm Area of Heap is special and it is used to stores String pool,  Meta data related to classes and method in JVM.

**Out Of Memory / OOM :**

***If it happens in your application take it is opportunity to display your technical skills.*** In technical term it means the JVM does not have space to put objects in **tenured space** which are survived in minor collection and it stops working or responding due to no memory to do further processing.

These are terms used in context of java memory management:

***Throughput*** is the percentage of total time not spent in garbage collection, considered over long periods of time. Throughput includes time spent in allocation (but tuning for speed of allocation is generally not needed).

***Pauses*** are the times when an application appears unresponsive because garbage collection is occurring.

***Footprint*** is the working set of a process, measured in pages and cache lines. On systems with limited physical memory or many processes, footprint may dictate scalability.

***Promptness*** is the time between when an object becomes dead and when the memory becomes available, an important consideration for distributed systems, including remote method invocation (RMI).

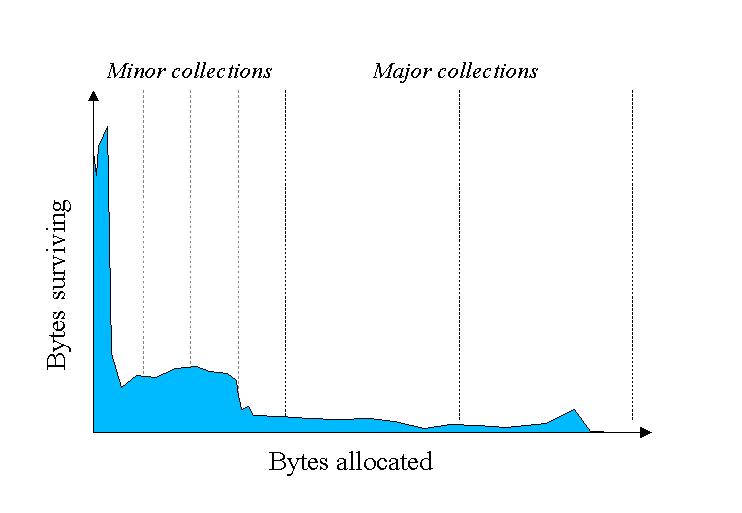
**Heap Dump**is a list of objects in the memory of JVM as well as the content of the memory occupied by those objects. It preserves the value of any attributes of the objects, including references to other objects. In other words, a heap dump gives you a complete picture of the memory.There are multiple tools that allow you to dump heap in a Java process:Some tools like VisualVM and memory profilers allow you to initiate a heap dump from the GUI, but you don’t need any fancy tools here—jmap will do just fine. As it provides the most general case, we'll use jmap in the next example.Before you dump heap, be sure to keep the following issues in mind:Note that the -F option, which will dump non-responsive programs, might be useful on UNIX systems, but is not available on Windows. Note also that JDK 6 includes the option +**XX:+HeapDumpOnOutOfMemoryError** that will dump heap whenever the OutOfMemoryError alert is encountered. This can be a useful option, but keep in mind that it has the potential to consume significant amounts of disk space.

The basic principles of garbage collection are:

Find data objects in a program that cannot be accessed in the future

Reclaim the resources used by those objects

   An object is considered garbage when it can no longer be reached from any pointer in the running program. The most straightforward garbage collection algorithms simply iterate over every reachable object. Any objects left over are then considered garbage. The time this approach takes is proportional to the number of live objects, which is prohibitive for large applications maintaining lots of live data.The blue area in the diagram below is a typical distribution for the lifetimes of objects. The X axis is object lifetimes measured in bytes allocated. The byte count on the Y axis is the total bytes in objects with the corresponding lifetime. The sharp peak at the left represents objects that can be reclaimed (i.e., have "died") shortly after being allocated. Iterator objects, for example, are often alive for the duration of a single loop.

[](http://www.oracle.com/ocom/groups/public/@otn/documents/digitalasset/190245.gif)

Application GC bytes allocation is shown in above picture. Majority of byte survival happens in Minor Collection and majority of objects are cleared in Major collections. Some objects do live longer, and so the distribution stretches out to the the right. For instance, there are typically some objects allocated at initialisation that live until the process exits. Between these two extremes are objects that live for the duration of some intermediate computation, seen here as the lump to the right of the initial peak. Some applications have very different looking distributions, but a surprisingly large number possess this general shape. Efficient collection is made possible by focusing on the fact that a majority of objects "die young".

Collectors

The Java HotSpot VM includes three different collectors, each with different performance characteristics.

The serial collector uses a single thread to perform all garbage collection work, which makes it relatively efficient since there is no communication overhead between threads. It is best-suited to single processor machines, since it cannot take advantage of multiprocessor hardware, although it can be useful on multiprocessors for applications with small data sets (up to approximately 100MB). The serial collector is selected by default on certain hardware and operating system configurations, or can be explicitly enabled with the option -**XX:+UseSerialGC.**

The ***parallel collector*** (also known as the *throughput collector*) performs minor collections in parallel, which can significantly reduce garbage collection overhead. It is intended for applications with medium- to large-sized data sets that are run on multiprocessor or multi-threaded hardware. The parallel collector is selected by default on certain hardware and operating system configurations, or can be explicitly enabled with the option -**XX:+UseParallelGC.**

**New**: *parallel compaction* is a feature introduced in J2SE 5.0 update 6 and enhanced in Java SE 6 that allows the parallel collector to perform major collections in parallel. Without parallel compaction, major collections are performed using a single thread, which can significantly limit scalability. Parallel compaction is enabled by adding the option -**XX:+UseParallelOldGC** to the command line.

The ***concurrent collector*** performs most of its work concurrently (i.e., while the application is still running) to keep garbage collection pauses short. It is designed for applications with medium- to large-sized data sets for which response time is more important than overall throughput, since the techniques used to minimize pauses can reduce application performance. The concurrent collector is enabled with the option -XX:+**UseConcMarkSweepGC**.

VM Options for GC details:

-verbose:gc — prints basic information about GC to the standard output

-XX:+PrintGCTimeStamps — prints the times that GC executes

-XX:+PrintGCDetails — prints statistics about different regions of memory in the JVM

-Xloggc: — logs the results of GC in the given file.

These **Command line options** are available via JMX to monitor VM memory-

**-Dcom.sun.management.jmxremote** — enables JMX monitoring

**-Dcom.sun.management.jmxremote.port** = — controls the port for JMX monitoring

com.sun.management.jmxremote.ssl

com.sun.management.jmxremote.authenticate

If you're using JDK 6, you can use tool called [jmap](http://olex.openlogic.com/packages/jmap) on any platform.

What is Garbage First Collector?

 Recently in one java interview, I was asked about Garbage First collector and Garbage First collector splits the heap up into fixed-size regions and tracks the live data in those regions. It keeps a set of pointers — the "remembered set" — into and out of the region. When a GC is deemed necessary, it collects the regions with less live data first (hence, "garbage first"). Often, this can mean collecting an entire region in one step: if the number of pointers into a region is zero, then it doesn't need to do a mark or sweep of that region.

    For each region, it tracks various metrics that describe how long it will take to collect them. You can give it a soft real-time constraint about pause times, and it then tries to collect as much garbage as it can in that constrained time. 

TECHNICAL DESCRIPTION

The G1 collector achieves high performance and pause time goals through several techniques.

The heap is partitioned into a set of equal-sized heap regions, each a contiguous range of virtual memory. G1 performs a concurrent global marking phase to determine the liveness of objects throughout the heap. After the mark phase completes, G1 knows which regions are mostly empty. It collects in these regions first, which usually yields a large amount of free space. This is why this method of garbage collection is called Garbage-First. As the name suggests, G1 concentrates its collection and compaction activity on the areas of the heap that are likely to be full of reclaimable objects, that is, garbage. G1 uses a pause prediction model to meet a user-defined pause time target and selects the number of regions to collect based on the specified pause time target.

The regions identified by G1 as ripe for reclamation are garbage collected using evacuation. G1 copies objects from one or more regions of the heap to a single region on the heap, and in the process both compacts and frees up memory. This evacuation is performed in parallel on multi-processors, to decrease pause times and increase throughput. Thus, with each garbage collection, G1 continuously works to reduce fragmentation, working within the user defined pause times. This is beyond the capability of both the previous methods. CMS (Concurrent Mark Sweep ) garbage collection does not do compaction. ParallelOld garbage collection performs only whole-heap compaction, which results in considerable pause times.

It is important to note that G1 is not a real-time collector. It meets the set pause time target with high probability but not absolute certainty. Based on data from previous collections, G1 does an estimate of how many regions can be collected within the user specified target time. Thus, the collector has a reasonably accurate model of the cost of collecting the regions, and it uses this model to determine which and how many regions to collect while staying within the pause time target.

RECOMMENDED USE CASES FOR G1

The first focus of G1 is to provide a solution for users running applications that require large heaps with limited GC latency. This means heap sizes of around 6GB or larger, and stable and predictable pause time below 0.5 seconds.

Applications running today with either the CMS or the ParallelOld garbage collector would benefit switching to G1 if the application has one or more of the following traits.  
- More than 50% of the Java heap is occupied with live data.  
- The rate of object allocation rate or promotion varies significantly.

First Round: Two algorithm and data structure questions. First-given huge file, how to find a pattern. Second- given m by n grid, find number of ways to reach bottom right from top left, you go either right or down. Other java string, class loader, how app server deploys ear, spring mvc, web xml etc.  
  
Second Round: Design restful web service for given problem. SOAP vs Restful. What are all java class loaders available and how they work. How HashMap works. Fail safe vs fail fast iterator. How BufferInputReader works. Multi-threading in java, gave variations of a problem and asked to implement using synchronized block/locks/wait and notify. Locks vs re-entrant locks. And other miscellaneous questions.  
  
Third Round: Asked questions about current project. Like technology stack, daily routine, what work I actually do, how I approach to resolve any production issue. Finally, asked about notice period and if I have any questions for him. :)

Hash-map Implementation,Collections,About time-complexity, Own iterator implementaion, finding duplicate count of each char in para etc...

Interview questions were really basic to check your knowledge about Java, JSP/Servlet, JDBC.  
  
Round1 : Telephone Call  
  
1. What is classpath in java ?  
1)The main difference between PATH and CLASSPATH is that  PATH is an environment variable which is used to locate JDK binaries like "java" or "javac" command used to run java program and compile java source file. On the other hand, CLASSPATH, an environment variable is used by System or [Application ClassLoader](http://javarevisited.blogspot.com/2012/12/how-classloader-works-in-java.html) to locate and load compile Java bytecodes stored in the .class file.  
  
2) In order to set PATH in Java, you need to include JDK\_HOME/bin directory in PATH environment variable while in order to set CLASSPATH in Java you need to include all those directories where you have put either your .class file or JAR file which is required by your Java application.  
  
3) Another significant difference between PATH and CLASSPATH is that PATH can not be overridden by any Java settings but CLASSPATH can be overridden by providing command line option -classpath or -cp to both "java" and "javac" commands or by using Class-Path attribute in Manifest file inside [JAR archive](http://java67.blogspot.com/2016/01/how-to-run-jar-file-from-command-prompt.html).  
  
4) PATH environment variable is used by operating system to find any binary or command typed in the shell, this is true for both Windows and Linux environment while CLASSPATH is only used by Java ClassLoaders to load class files.  
  
Read more: <http://www.java67.com/2012/08/what-is-path-and-classpath-in-java-difference.html#ixzz4yWAZWsXp>  
5. How do we run new thread ? Why do we prefer implementing Runnable interface over extending Thread ?  
6. Difference betwee ArrayList and LinkedList in java ?

) How hash map implemented in Java? What is the purpose of hashcode() and equals() method.?  
  
b) Questions about ORM's( iBatis , Hibernate) in Java and their usage.  
c) Quick sort algorithm and implementation logic.  
d) Questions on Spring and Struts Framework.  
e) Two large lists with duplicates were give how to make it as single list by removing duplicates and preserver ordering?

ArrayList a1, a2; // fill with junk  
  
a1.removeAll( a2 ); // removes duplicates  
a1.addAll( a2 ); // combines lists  
  
You might also want to use LinkedLists instead of ArrayLists in order to avoid costly removing-from-an-array operations, and the annoying array-growing operation.  Anyhoo, as long as your chosen collection implements the java.util.Collection interface, you can do the above.

Is using a HashSet the  fastest way to combine two ArrayList(s) and omit duplicates?  
  
ArrayList a1, a2; // fill them with junk  
  
HashSet hash = new HashSet(a1);  
hash.addAll(a2);

  String array1[] = { "Java", "JSP", "Servlet" };  
      String array2[] = { "Spring", "Hibernate" };  
       
      List<String> list3 = **new**ArrayList<String>();  
      list3.addAll(Arrays.asList(array1));  
      list3.addAll(Arrays.asList(array2));  
      String array3[] = list3.toArray(**new**String[list3.size()]);  
f) Java call by reference or by value? Explain with example.  
g) What is static import? Other Core java questions...............  
  
On the same day evening i got a call from HR and scheduled for a Face to Face Interview after 2 days.  
  
2) Second Round : Technical Round.  
  
a) In this round they asked me to explain about Design Patterns and write sample code of implementation.  
I told them about Factory and Singleton Design pattern's and wrote pseudo code.  
  
b) Asked to design Time sheets logging application (Just have to told them what table to be used , how to call them from Java (ORM) and asked me to add authentication to the site .  
  
c) Implementation of Bubble sort (Simple and straight forward !)  
  
d) JDBC and JNDI usage. Application deployment process in Tomcat , Jboss , Weblogic.  
  
and so on,,  
  
3) HR round : It was simplest round , they just asked me about my notice period , ctc and expected ctc. Told me to wait for 10 min and i was selected and got offer letter on the same day evening  
  
Overall interview process was good. If we have a good working knowledge on J2EE and a core java preparation it can be cracked .

ou have two arrays... merge both arrays into one collectoin and avoid duplicates..Which Java Collection will you use ? HASHSET thts all

Again DS &Puzzles,Java indepth (hash code equals methods,implement hash table),Arraylist and linkedlist which one is better in which case,How to implement a dictionary

Design a chessboard

-They asked me to explain the dijkstra's algorithm and about its time complexity.

How would you create a Queue given just one Stack in Java?

# Vitech(Product)

What would you do when the request to a server was un-responsive  
2. What steps do you take to fix a long running query.  
3. What do you do to fix a OutOfmemory issue on the server.

How can you load configuration once in a web app?

Java Code to implement 5 places (code)

# Synechron

There are three methods as below.  
void method(double d, int i){}  
void method(int i, double d){}  
void method(String s1, String s2){}  
Now, if I call method(0,0). Which method will be called?

Ans: ambiguity error  
- Tell me 5 things which I can do to override any method.  
- Why Strings are immutable?  
- Write your own immutable class. Then following questions of Deep cloning.  
- Which are the Wrapper classes in java?  
- Write a program to reverse a string without using any inbuilt direct function.  
- Write a program to check numbers till 10 are even or odd using recursion.  
- Write a program to check given string is Palindrome or not.  
- Can we write custom exception? If yes, how?  
- What is contract between equals() and hashcode(). Show me by writing code if you want to override both methods.  
- I have a class i.e Employee which has two properties id and name. If I want to sort list of employees by their names, what should I do?  
- Difference between synchronized and Lock.  
- Difference between synchronizedMap and ConcurrentHashMap.  
- Producer/Consumer problem using wait and notify  
- Write a program of Producer/Consumer problem using BlockingQueue.  
- Spring IOC. What are different ways to achieve Spring IOC?  
- Difference between BeanFactory and ApplicationContext.  
- Spring Bean Lifecycle.  
- How to create RESTful Webservice in Spring.  
- What to do if I want to get 1st number employee details by hitting URL www.restapi.com/employee/1. Write a controller method for the same.  
- What are the properties to configure Hibernate?  
- What is dialect? Why is it used?  
- How do we write query in Hibernate same as ‘SELECT \* FROM EMPLOYEES’ in SQL?  
- What are alternatives to write queries in Hibernate apart from HQL?  
- Write configuration for one-to-many relationship between two classes in Hibernate.  
- Difference between save() and merge().  
- List<ParentClass> list = new ArrayList<ChildClass>();  
What will happen?  
- I have one method void method(ArrayList<String> list){}.  
I’m calling this method by passing raw-type arraylist i.e. method(new ArrayList()). What will happen?  
- How to convert given String to integer? If I do Integer.parseInt("chintan"); what will happen?  
- What are the classes you have used for Stream io?  
- What is the difference between Aggregation and Composition?

How multiple requests will be handled by single servlet instance

jvm jre and jdk ?  
static method can be override ?  
diff b/w list and set  
diff b/w hashset and treeset  
how sorting is performed in treeset ?  
diff b/w sleep and wait?  
diff b/w noclasdeffound and classnotfound exception  
how u catch exception through jdbc templates  
create your own custom exception class  
design pattern of dispatcherservlet  
modules of spring  
scope of bean  
prototype through singlton and vice versa ?  
same class with diferent bean id in xml file and if yes then how many instances ?  
mappings in hibernate  
diff between hashmap and concurentHashmap  
concurentHashmap in detail  
Bean Cycle  
MVC pattern and its flow  
Callable interface  
Why hibernate and its advantages?  
caching in hibernate  
how to enable second level caching ?  
Indexing- clustered - non-clustered

new String() - how max size of string we can pass using String constructor.  
-What are the design patterns used in JSF?  
-Many to Many relationship in hibernate.  
-How to read a DB table which is very large number of records and insert into other same table?  
-OOPS principle  
-Difference between encapsulation and abstraction?  
-Spring can have multiple SessionFactory with different data source?  
-What is cloning?  
-Why we need serialization?  
-When to use interface and abstract class?  
-Why java.util.concurrency package even though we can implement multi threading?  
-ConcurrentModificationException? How to avoid it.  
-Different type of caching in hibernate?  
-What Future interface returns?  
-ExecutorService?  
-How to communicate using web service by making data securely?  
-Servlet life cycle.  
-JSP scriptlet tag? Where the contents of jsp scriptlet tag will go?  
-URL encoding?  
-How to make a class immutable?

**The maximum size of the java string?**

StringBuffer uses a char[].  
In Java an array can be indexed **only via an integer** which means the highest value the index of the array can be isInteger.MAX\_VALUE - 1 (i.e. 2^31 - 1). Which means that the size of an array in Java can not be larger thanInteger.MAX\_VALUE.

its Integer.MAX\_VALUE or 2^31-1 or about 2 billion. You are more likely to have memory problems before getting to this size. e.g. You need 4 GB for the String and 4 GB to create it.

**Class loaders internal working in java  ?**

Java class loaders are used to load classes at runtime. ClassLoader in Java works on three principle: delegation, visibility and uniqueness. Delegation principle forward request of class loading to parent class loader and only loads the class, if parent is not able to find or load class. Visibility principle allows child class loader to see all the classes loaded by parent ClassLoader, but parent class loader can not see classes loaded by child. Uniqueness principle allows to load a class exactly once, which is basically achieved by delegation and ensures that child ClassLoader doesn't reload the class already loaded by parent. Correct understanding of class loader is must to resolve issues like [NoClassDefFoundError in Java](http://javarevisited.blogspot.sg/2011/06/noclassdeffounderror-exception-in.html) and [java.lang.ClassNotFoundException](http://javarevisited.blogspot.sg/2011/08/classnotfoundexception-in-java-example.html), which are related to class loading. ClassLoader is also an important topic in advanced Java Interviews, where good knowledge of working of Java ClassLoader and [How classpath works in Java](http://javarevisited.blogspot.ca/2011/01/how-classpath-work-in-java.html) is expected from Java programmer. I have always seen questions like, Can one class be loaded by two different ClassLoader in Java on various [Java Interviews](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html).  In this Java programming tutorial, we will learn what is ClassLoader in Java, How ClassLoader works in Java and some specifics about Java ClassLoader.

What is ClassLoader in Java

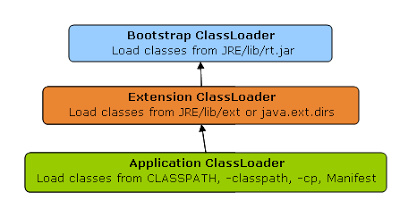
ClassLoader in Java is a class which is used to load [class files in Java](http://javarevisited.blogspot.ca/2012/05/10-points-about-class-file-in-java.html). Java code is compiled into class file by [javac](http://javarevisited.blogspot.sg/2012/12/javac-is-not-recognized-as-internal-or-external-command.html)compiler and [JVM](http://javarevisited.blogspot.sg/2011/12/jre-jvm-jdk-jit-in-java-programming.html)executes Java program, by executing byte codes written in class file. ClassLoader is responsible for loading class files from file system, network or any other source. There are three default class loader used in Java, Bootstrap , Extension and System or Application class loader.   
Every class loader has a predefined location, from where they loads class files. Bootstrap ClassLoader is responsible for loading standard JDK class files from rt.jar and it is parent of all class loaders in Java. Bootstrap class loader don't have any parents, if you call String.class.getClassLoader() it will return null and any code based on that may throw [NullPointerException in Java](http://javarevisited.blogspot.com/2012/06/common-cause-of-javalangnullpointerexce.html). Bootstrap class loader is also known as Primordial ClassLoader in Java.    
  
Extension ClassLoader delegates class loading request to its parent, Bootstrap and if unsuccessful, loads class form jre/lib/ext directory or any other directory pointed by java.ext.dirs system property. Extension ClassLoader in JVM is implemented by  sun.misc.Launcher$ExtClassLoader.   
  
Third default class loader used by JVM to load Java classes is called System or Application class loader and it is responsible for loading application specific classes from [CLASSPATH](http://javarevisited.blogspot.sg/2011/01/how-classpath-work-in-java.html) environment variable, -classpath or -cp command line option, Class-Path attribute of Manifest file inside JAR. Application class loader is a child of Extension ClassLoader and its implemented by sun.misc.Launcher$AppClassLoader class. Also, except Bootstrap class loader, which is implemented in native language mostly in C,  all  Java class loaders are implemented using java.lang.ClassLoader.

In short here is the location from which Bootstrap, Extension and Application ClassLoader load Class files.

1) Bootstrap ClassLoader - JRE/lib/rt.jar

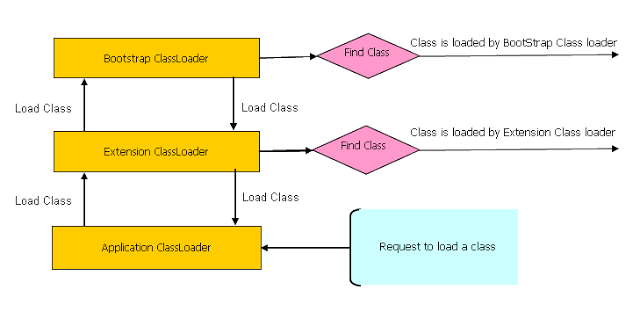
2) Extension ClassLoader - JRE/lib/ext or any directory denoted by java.ext.dirs

3) Application ClassLoader - CLASSPATH environment variable, -classpath or -cp option, Class-Path attribute of Manifest inside [JAR file](http://javarevisited.blogspot.sg/2012/03/how-to-create-and-execute-jar-file-in.html).

[](http://2.bp.blogspot.com/-HCTsr-j_ojw/USTOh1f8JwI/AAAAAAAAAjg/YegPspR5K48/s1600/java_classloader_hierarchy.PNG)

How ClassLoader works in Java

[What is ClassLoader in Java, How classloader works in Java](http://3.bp.blogspot.com/-K6q0DQ1v-tw/TWu8owBtc2I/AAAAAAAAADA/oBoHDBiJ8ag/s1600/17.jpg)As I explained earlier Java ClassLoader works in three principles : delegation, visibility and uniqueness. In this section we will see those rules in detail and understand working of Java ClassLoader with example. By the way here is a diagram which explains How ClassLoader load class in Java using delegation.

[](http://1.bp.blogspot.com/-0gOWex7Pb2E/USTOh2K7zpI/AAAAAAAAAjc/_viQADzxrsk/s1600/Java+classloader+working.PNG)

Delegation principles

As discussed on [when a class is loaded and initialized in Java](http://javarevisited.blogspot.sg/2012/07/when-class-loading-initialization-java-example.html), a class is loaded in Java, when its needed. Suppose you have an application specific class called Abc.class, first request of loading this class will come to Application ClassLoader which will delegate to its parent Extension ClassLoader which further delegates to Primordial or Bootstrap class loader. Primordial will look for that class in rt.jar and since that class is not there, request comes to Extension class loader which looks on jre/lib/ext directory and tries to locate this class there, if class is found there than Extension class loader will load that class and Application class loader will never load that class but if its not loaded by extension class-loader than Application class loader loads it from [Classpath in Java](http://java67.blogspot.sg/2012/08/what-is-path-and-classpath-in-java-difference.html). Remember Classpath is used to load class files while [PATH](http://javarevisited.blogspot.ca/2011/10/how-to-set-path-for-java-unix-linux-and.html) is used to locate executable like javac or java command.

Visibility Principle

According to visibility principle, Child ClassLoader can see class loaded by Parent ClassLoader but vice-versa is not true. Which mean if class Abc is loaded by Application class loader than trying to load class ABC explicitly using extension ClassLoader will throw either [java.lang.ClassNotFoundException](http://javarevisited.blogspot.ca/2011/08/classnotfoundexception-in-java-example.html). as shown in below Example

public class ClassLoaderTest {    
    public static void main(String args[]) {  
        try {            
            //printing ClassLoader of this class  
            System.out.println("ClassLoaderTest.getClass().getClassLoader() : "  
                                 + ClassLoaderTest.class.getClassLoader());  
  
            
            //trying to explicitly load this class again using Extension class loader  
            Class.forName("test.ClassLoaderTest", true   
                            ,  ClassLoaderTest.class.getClassLoader().getParent());  
        } catch (ClassNotFoundException ex) {  
            Logger.getLogger(ClassLoaderTest.class.getName()).log(Level.SEVERE, null, ex);  
        }  
    }  
  
}  
  
Output:  
ClassLoaderTest.getClass().getClassLoader() : sun.misc.Launcher$AppClassLoader@601bb1  
16/08/2012 2:43:48 AM test.ClassLoaderTest main  
SEVERE: null  
java.lang.ClassNotFoundException: test.ClassLoaderTest  
        at

Uniqueness Principle

According to this principle a class loaded by Parent should not be loaded by Child ClassLoader again. Though its completely possible to write class loader which violates Delegation and Uniqueness principles and loads class by itself, its not something which is beneficial. You should follow all  class loader principle while writing your own ClassLoader.

How to load class explicitly in Java

Java provides API to explicitly load a class by Class.forName(classname) and Class.forName(classname, initialized, classloader), remember JDBC code which is used to load JDBC drives we have seen in [Java program to Connect Oracle database](http://javarevisited.blogspot.ca/2012/04/java-program-to-connect-oracle-database.html). As shown in above example you can pass name of ClassLoader which should be used to load that particular class along with binary name of class. Class is loaded by calling loadClass() method ofjava.lang.ClassLoader class which calls findClass() method to locate bytecodes for corresponding class. In this example Extension ClassLoader uses java.net.URLClassLoader which search for class files and resources in [JAR](http://javarevisited.blogspot.ca/2012/10/5-ways-to-add-multiple-jar-to-classpath-java.html) and directories. any search path which is ended using "/" is considered directory. If findClass() does not found the class than it throws [java.lang.ClassNotFoundException](http://javarevisited.blogspot.de/2012/03/jdbc-javalangclassnotfoundexception.html) and if it finds it calls defineClass() to convert bytecodes into a .class instance which is returned to the caller.

Where to use ClassLoader in Java

ClassLoader in Java is a powerful concept and used at many places. One of the *popular example of ClassLoader* is AppletClassLoader which is used to load class by Applet, since Applets are mostly loaded from internet rather than local file system, By using separate ClassLoader you can also loads same class from multiple sources and they will be treated as different class in [JVM](http://javarevisited.blogspot.ca/2011/12/jre-jvm-jdk-jit-in-java-programming.html). J2EE uses multiple class loaders to load class from different location like classes from WAR file will be loaded by Web-app ClassLoader while classes bundled in EJB-JAR is loaded by another class loader. Some web server also supports hot deploy functionality which is implemented using ClassLoader. You can also use ClassLoader to load classes from database or any other persistent store.

That's all about What is ClassLoader in Java and How ClassLoader works in Java. We have seen delegation, visibility and uniqueness principles which is quite important to debug or troubleshoot any ClassLoader related issues in Java. In summary knowledge of How ClassLoader works in Java is must for any Java developer or architect to design Java application and packaging.

# Oracle

Difference between compilers and interpretors. What does Java employ and why

Design a Cache Control in java specifying what would be exposed in an interface and what would contribute to an abstract class

They asked me to design a software for bus ticket booking

A code for finding all anagrams of a word

How do you avoid a deadlock in your program?

1. Using synchronisation properly. 2) Using notify-wait properly.

Scenario given: payroll is running in a pl/sql code block. paralelly all the bonus must be updated without touching the existing session

OOPS Concepts and Java Memory Management

Write a program to eliminate duplicate occurance of a character in a string and display '\*' representing the count of each character as shown below. String: MOHAMMED SHAKIR Output: 'M' 'O' 'H' 'A' 'E' 'D' ' ' 'S' 'K' 'I' 'R' \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

Write a program to display all the possible sequence of numbers from an array which can sum up to a number inputted by the user. Implement using O(n) complexity.

Basics of Complexity Analysis of Codes. (Big-Oh,Big-Theta

 Can a constructor throw an exception

Yes, constructors can throw an exception. However, please note that, if a superclass constructor throws a checked exception, then if any subclass constructor calls that superclass constructor must throw the same exception, or a broader exception.

Sorted 2d array find position of n

**What is RAW datatype?**

RAW datatype is used to store values in binary data format. The maximum size for a raw in a table in 32767 bytes.

RAW datatype is used to store data in binary format.Its length is 32767 bytes in length.

**Difference between varchar and varchar2 data types?**

**Varchar**

 1.varchar can store upto 2000 bytes.

  2. varchar occupies space for the NULL values.

  3.varchar is an ANSI standard

**Varchar2 :**

 1.varchar2 can store upto 4000 bytes.

2. varchar2 is not occupying space for the NULL values.

3.varchar2 is an oracle standard.

**How to get nth max salaries?**

Suppose we have a table, Emp (empId, salary......).

Now for nth max salary,

select min(t.salary) from (select \* from Emp order by salary desc limit n) as t;

select \* from employee emp1 where (N-1)=(select  count(Distinct (emp2.salary)) from employee emp2 where emp2.salary>emp1.salary

select distinct hiredate from emp a where &n =  (select count(distinct sal) from emp b where a.sal >= b.sal);

SELECT MIN(SAL) FROM (SELECT DISTINCT(SAL) FROM EMP ORDER BY SAL)

WHERE ROWNUM<=N;

select max(t.salary) from (select \* from Emp order by salary desc limit n) as t;

reference:hac\_123

**Difference between & and && operator**

& is bitwise. && is logical.

& evaluates both sides of the operation.  
&& evaluates the left side of the operation, if it's true, it continues and evaluates the right side.

& is a bitwise operator whereas && is a logical operator.

The & operator first converts both the operands in binary and then performs the operation.

Example 3&5 results in 1.

For 3= 011

For 5=101.

3&5  =001 which is 1.

(Logical and) && checks both the operands and will be true if both the operands are other than zero.

**Difference between instanceof and isInstance( ) method in java**

Both isInstance() method and instanceof operator (type comparison operator) are used to check if an object is of a particular class or interface type, yet there are differences. Let's see those differences between isInstance()and instanceof. The right-hand-operand of instanceof operator must be evaluated to a type (class or interface), which is known at compile time. There may be chances when you don't know the type name at compile time and you want to pass it as an argument that will be resolved at run time. In those circumstances you won't be able to use instanceof operator because instanceof does not work for types evaluated at run time.

The isInstance() method, which is dynamic equivalent of the Java language instanceof operator does type checking at run time. The public boolean isInstance(Object obj) method determines if the specified Object is assignment-compatible with the object represented by this Class. This method returns true if the specified Object argument is non-null and can be casted to the reference typerepresented by this Class object without raising a ClassCastException. It returns false otherwise.

The term dynamic equivalent is used because isInstance() method is used to check type of a reference at run time. The very important application use of isInstance() method is it can be used in code dealing with type reflection at runtime. Following is a small Java program demonstrates isInstance() and instanceof.

**Example:**

class A {}

class B extends A {}

class C extends A {}

public class InstanceofDemo

{

public static void main([String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) args[])

{

A a = new A();

B b = new B();

C c = new C();

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("a instanceof A: " + (a instanceof A)); //true

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("b instanceof A: " + (b instanceof A)); //true

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("c instanceof A: " + (c instanceof A)); //true

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("a instanceof B: " + (a instanceof B)); //false

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("null instanceof A: " + (null instanceof A)); //false

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("a.getClass().isInstance(a): " + a.getClass().isInstance(a)); //true

[System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("b.getClass().isInstance(c): " + b.getClass().isInstance(c)); //false

}}

 OUTPUT

======

D:\JavaPrograms>javac InstanceofDemo.java

D:\JavaPrograms>java InstanceofDemo

a instanceof A: true

b instanceof A: true

c instanceof A: true

a instanceof B: false

null instanceof A: false

a.getClass().isInstance(a): true

b.getClass().isInstance(c): false

 In above piece of code, you might have noticed that expression a instanceof B returns false. This is because, at run time, the result of the instanceofoperator is true if the value of the left-hand operand is not null and the reference could be cast to the ReferenceType without raising a ClassCastException. Otherwise the result is false.

**Difference between new operator and newInstance( ) method in java ?**

new operator is used for creating a new object when the class name is known at beginning .

whereas  newInstance()  is used to create new object when the class name is not known at run time . in this case we provide the name the name of class at run time through console.

**public static void main(String[] args)   
    {  
        if(true)  
            int x=10;  
    }**

This will give Compile time error .

Because we can write a single statement  inside if block but that should not be declarative statement

Any other statement is valid that is not declarative is valid .

if(true)  
   System.out.println("This is geeks for geeks ");

or **if**(**true**)

{**int** x=10;}

Both of above will work.

**Can we set the Daemon nature of main thread ?**

No , we can not set the Daemon nature of main thread .

Because we can only set the Daemon nature of threads before its start , whereas main thread gets started as soon as program starts .

No we cant set daemon nature to a main thread Instead u can give daemon nature too the user threads with the help of **setDaemon() method**

**Daemon thread** is a service provider thread that provides services to the user thread for background supporting tasks. When all the user threads die, JVM terminates this thread automatically.

It is a low priority thread.

java.lang.Thread provides 2 methods for daemon threads:

**public** **void** setDaemon(**boolean** status)- is used to mark the current thread as Daemon thread.

**public** **boolean** isDaemon()-used to check whether the current thread is daemon or not.

**How can you suspend and resume a thread  ?**

When the **sleep()** method time is over, the thread becomes implicitly active. **sleep()** method is preferable when the inactive time is known earlier. Sometimes, the inactive time or blocked time may not be known to the programmer earlier; to come to the task here comes **suspend()**method. The suspended thread will be in blocked state until **resume()** method is called on it. These methods are deprecated, as when not used with precautions, the thread locks, if held, are kept in inconsistent state or may lead to deadlocks.

**Note:** You must have noticed, in the earlier **sleep()** method, that the thread in blocked state retains all its state. That is, attribute values remains unchanged by the time it comes into runnable state.

**t1.stop( ) ;**

say , we have created thread t1 in our program then it is recommended to stop a thread after its use . we can stop /kill a thread by using  **t1.stop( ) ;** in our program.

**Thread Group**

**What is the difference between object lock and class level lock ?**

Object level locking is mechanism when you want to synchronize a non-static method or non-static code block such that only one thread will be able to execute the code block on given instance of the class.

Class level locking prevents multiple threads to enter in synchronized block in any of all available instances on runtime.

**What is the disadvantage of using synchronized keyword in java ?**

**Advantage of synchronized keyword in java ?**

Process of protecting data from inconsistency

**What do you mean by Green Threads in terms of  multi threading ?**

Threads that are scheduled by a runtime library or virtual machine (VM) instead of natively by the underlying operating system.

**Copy constructor in java?**

According to typical definition of Java , we can not use pointers in Java ,but we can improvise the concept of copy constructor in c++ and we can achieve same output in java .

**Improvisation used :- Pass class type variable as a parameter to next function and we can achieve it .**

see below :-

package copyconstructor;  
public class CopyConstructor   
{  
    public static void main(String[] args)   
    {  
        copy c1=new copy(10,20);  
        c1.show();  
        copy c2=new copy(c1);  
        c2.show();  
    }}  
class copy  
{  
     int x,y;  
     copy(int x,int y)  
    {  
          this.x=x;  
          this.y=y;  
    }  
    copy(copy z)                 // we can pass the class type variable to copy the instance of object .   
    {          this.x=z.x;  
          this.y=z.y;  
    }  
    void show()  
    {

       System.out.println(x);  
       System.out.println(y);

    }}

**Explain difference between multi-threading and multi-tasking?**

Multi-threading:

1) Multiple threads are executing at the same time at same or different part of program

2) CPU switches between multiple threads

3) It is light weight process

4) Example: WEB SERVER

5) It is feature of process

6) Multi-threading is sharing of computing resources among threads of a single process.

Multi-tasking:

1) Several programs are executed concurrently

2) CPU switches between multiple tasks and processes

3) It is heavy weight process

4) Example: WEB BROWSER

5) It is feature of OS

6) Multi-tasking is sharing of computing resources(CPU, memory, devices, etc.) among processes

**What algorithm is used for garbage collection in Java?**

Whenever objects are created they are assigned memory in a heap. So whenever memory is full or overflow condition occurs we need to clear heap . This is done by algorithm called "**Mark and Sweep Algorithm"**

**It is known as Stop & Run Algorithm.**

It has two phases:

1) Mark Phase: to detect unreachable objects

2) Sweep Phase: The second pass traverses all pointers, starting at the accessible roots of a program and for each object traversed it marks the object.The third pass walks the heap linearly again and removes all objects that are not marked.

Benefits:

1) No overhead of pointers manipulations

2) Low space cost

Drawbacks:-

1) Recursive marking leads to space and time issue

2) Cells are reexamined during sweep

**Briefly explain the difference between tree and graph?**

Tree:

1) Tree has only one root node and child have single parent only

2) It is minimally connected graph where there is single path between two vertices.

3) In this we have no loops, no cycles and no circuits.

4) It contains parent child relationship

5) They are less complex with no cycles and no self loops

6) Types of traversals are pre-order,post-order and inorder

7) These are direct acyclic graph

8) Different types are binary, binary search tree, AVL tree, Red Black tree etc.

9) Application: sorting and traversal

10)Number of edges are :- n-1

Graph:

1) No necessarily it will have single root.

2) They can have more than one path

3) They have loops, self loops and circuits

4) No parent child relationship is there

5) Graphs are traversed using DFS and BFS

6) It can be acyclic or cyclic

7)  Types are directed and undirected graphs

8) It can be use for graph coloring, job scheduling

9) No of edges depends on graphs

**What is 8 Queen problem ?**

In this puzzle on a 8\*8 chess board, 8 queens are placed in such a way that no queen can attach each other. In this no queen should share same row, column or diagonal. It can be solved up to 23 queen problem.The problem was proposed by Max  Bezzel including Gauss who solved this problem.

Dijkstra solved it using structured programming. He told about depth first backtracking algorithm.At each recursive step we will place queen in specific column. While placing queen it should be known where other queens are placed.If placement within the column does not lead to solution then queen is moved down.

Pruning: If queen cannot be placed in the i column don't try in i+1 column instead try i-1 column.

Steps:

1) Place the first queen in the left upper corner.

2) Save the attacked position

3) Move to the next queen

4) Search for valid position if you find then go to step 8

5) If there is no valid position for the queen then delete it

6) Move to the previous queen

7) Go to step 4

8) Place it in first valid position

9) Save the attacked position.

10) If queen is proposed in last step otherwise go to step 8

**How undo works internally?**

For implementing undo and redo two stacks are made. So whenever user performs any action then pairs of elements are pushed on to undo stack first one is undo action and next one is its inverse.

Whenever undo is revoked, those pair of values are popped from the stack and placed in the redo stack. Whenever redo is invoked again those value pair are pushed on to undo stack.

**What is heap sort?**

It is the best in place sorting algorithm with no quadratic worst case.

(i) It consists of two steps first creating heap

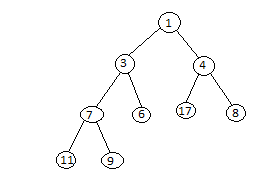
(ii) Then moving the largest element at the root of the heap

Heap should follow following property:-

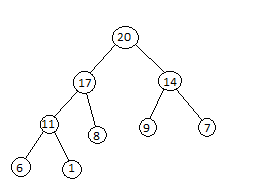
1) It should follow complete binary tree format means no node should be left blank in between

2) If the parent value is larger than its children then it is max heap else it is min heap when value of parent is less than its children.

min heap



max heap



**What is waterfall model?**

Waterfall model was the first model proposed. It is also known as linear sequential model. In this first phase must be completed then we can execute the next phase. No overlapping in phases is allowed in the waterfall model. It is the first model in the software development life cycle.

phases of waterfall are:-

1) Requirement Analysis:All the requirements needed for the software are captured in this phase

2) System Design:-On the basis of requirement gathered model is designed overall system architecture is developed

3) Implementation:-Based on this design small program is developed those small program are known as units

4) Testing:-All the small units are integrated and tested.

5) Deployment:- Once all the testing is done product is deployed in the market.

6) Maintenence: Issues identified by the user are fixed.

Advantages:-

1) This model is easy to implement and maintain

2) Each phase has specific deliverables and review process.

3) It works for small projects.

Disadvantages:-

1) Once process enters testing phase then it cannot be revert back.

2) Poor model for long processes.

3) High amount of risk and uncertainty

**Difference between UNION and UNION ALL?**

UNION:

1) It removes duplicate values in the data

2) As it removes duplicates so it is usually slower than UNION ALL

3) It performs less than UNION ALL

4) It is sorted.

5) SYNTAX:- select \* from table1 UNION select \* from table2;

UNION ALL

1) It does not remove duplicate values from the data

 2) As it does not  removes duplicates so it is usually faster.

3) It is more efficient than UNION.

4) It is unsorted

5) SYNTAX:- select \* from table1 UNION ALL select \* from table2;

**What are Triggers?**

Triggers are stored program or procedure which automatically get executed when some event occurs.They can be executed in response to:-DDL:(update,insert, delete), DML:-(create,alter,drop) and  Database Operation(logon,start up, shut down)

Benefits of using Triggers:-

Prevent invalid transactions

Provides security and authentication

Syntax:CREATE [OR REPLACE ] TRIGGER trigger\_name {INSERT [OR] | UPDATE [OR] | DELETE} [OF col\_name] ON table\_name [REFERENCING OLD AS o NEW AS n] [FOR EACH ROW] WHEN (condition) DECLARE Declaration-statements BEGIN Executable-statements EXCEPTION Exception-handling-statements END;

Each trigger is enclosed in Begin and end tags

A **database trigger** is [procedural code](https://en.wikipedia.org/wiki/Procedural_code) that is automatically executed in response to certain [events](https://en.wikipedia.org/wiki/Event_%28computing%29) on a particular [table](https://en.wikipedia.org/wiki/Table_%28database%29) or [view](https://en.wikipedia.org/wiki/View_%28database%29) in a [database](https://en.wikipedia.org/wiki/Database). The trigger is mostly used for maintaining the [integrity](https://en.wikipedia.org/wiki/Database_integrity) of the information on the database. For example, when a new record (representing a new worker) is added to the employees table, new records should also be created in the tables of the taxes, vacations and salaries.

The four main types of triggers are:

Row Level Trigger: This gets executed before or after any column value of a row changes

Column Level Trigger: This gets executed before or after the specified column changes

For Each Row Type: This trigger gets executed once for each row of the result set affected by an insert/update/delete

For Each Statement Type: This trigger gets executed only once for the entire result set, but also fires each time the statement is executed.

**Why is Java platform independent?**

While you write a Java code and feed it to the compiler a .class file will be generated. Now this .class file/Byte code common for all kind of system whether its Linux or Mac or Windows.  Now to run this .class file, we need JRE (which is actually implementation of JVM). SO Mac will have its own version of JRE, Widows will have different and Linux has different one too. But one thing to notice here is that --- the input for all these , means the .class file is same. This is actually Platform Independence.

**Differentiate between DELETE, DROP and TRUNCATE.**

**DELETE:**It is a DML command that is used to remove the data from a table. It can be used in conjunction with WHERE to delete only the rows meeting the specified criteria.  
**TRUNCATE:**It is a DDL which is used to remove ALL the rows from a table. However, it does not delete the table.  
**DROP:**It is used t remove the data as well as the table from the database.

DELETE command is used to delete rows from an existing table. The rows get deleted on the basis of the conditions mentioned in WHERE clause and if there is no WHERE clause in the query, all the rows are deleted. But the space which was occupied by the table still exists.

DELETE FROM table\_name;

TRUNCATE command deletes all the rows from a table and free the space containing the table.

TRUNCATE TABLE table\_name;

DROP  command deletes the rows as well as the schema of the table. i.e. the complete table gets deleted.

DROP TABLE table\_name;

When the table is dropped, all its relationships, integrity constraints, grant and access privileges also get deleted. And if a table is truncated or deleted, the table structure and schema remains the same, only the rows get deleted.

If table is deleted, the memory of the table still exists but in the truncation the memory also gets freed.

DELETE is a DML command is used to delete rows from a table. We can specify a WHERE clause to delete only specific records. In case if there is no WHERE clause specified all rows will be deleted from the table.  
Syntax:

DELETE FROM tablename where field='value';

DELETE FROM tablename;

TRUNCATE is a DDL command removes all rows from a table whereas table is not deleted.  
Syntax:

TRUNCATE table tablename;

DROP is a DDL command removes the data as well as the table from the database. All the table's rows, indexes, triggers will be dropped.  
Syntax:

DROP table tablename;

**What is Hashing?**

Hashing is a process of indexing and retrieving data in data structure to provide faster way to access element using hashing key.

Hash table is just an array which maps a key (data) into the data structure with the help of hash function such that insertion, deletion and search operations can be performed with constant time complexity (i.e. O(1)).Hash function is a function which takes a piece of data (i.e. key) as input and outputs an integer (i.e. hash value) which maps the data to a particular index in the hash table.

In this element is searched with constant time complexity.

Hashing is process of mapping large amount of table into smaller table with the help of hashing function.

In this process index is associated with each data element which makes the access of data faster.

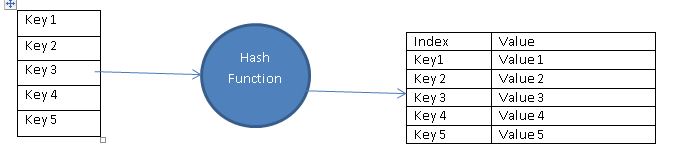
In this array is used for storing data and index associated with it is found using hashing function.

hash function = array\_size % key

Hashing is used in many places. For example in hospitals to find record of patient using patient\_no.

Large keys are converted into small using hash function and these key-value pairs are stored in hash table for access.

Requirements of good hashing function (i) easy to calculate (ii) less collisions(which means two elements with same key)  (iii) It should be uniformly distributed without clustering



Suppose we have key-value pairs as:- with array\_size = 20,  (1,20) (2,70),(42,80),(4,5)

hash\_function = (key% number of elements)

(a) hash\_fn = key%array\_size = 1%20 = 1

(b) hash\_fn = 2%20 = 2

(c) hash\_fn  = 42%20 = 2

(d)hash\_fn = 4%20 = 4

so element should be stored at index 1,2,2,4 respectively but two element cannot be placed in same index ie. 2

Then value associated with is placed in those index

So for this we will perform linear probing that means placing element in next empty index ie. 3

https://practice.geeksforgeeks.org/ckeditor/images/uploads/1491369602_Capture.JPG

Coding structure:-

struct DataItem { int data; int key;};

int hashCode(int key){ return key % SIZE;}

Hash Table :- it a way of storing data in array like format such that access of data becomes faster with index-value stored in the table.Each position is known as slot and mapping between slot and element is done using hash function.

Load factor = (lemda) = number of items/table\_size

Other methods to compute hashing functions are (a) folding method:-dividing item into equal size then adding them all

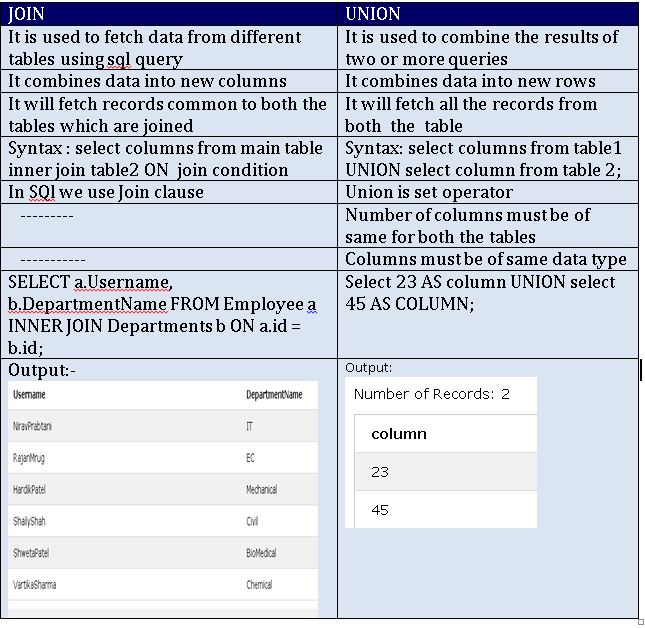
example:- 436-555-4601 so

hash\_function =  43+65+55+46+0143+65+55+46+01 = 210

hash\_fn = 210%11  = 1

(b) mid square method:-in this taking number squaring it and then taking mid and then mod with size

like 44^2 = 1936=(93%11)



**Where is stack/queue used in web browser?**

Stack is used browser back button :

Every web browser has a Back button. As we navigate from web page to web page, those pages are placed on a stack (actually it is the URLs that are going on the stack). The current page that we are viewing is on the top and the first page we looked at is at the base. If we click on the Back button, we begin to move in reverse order through the pages.

Other uses of stacks :

An "undo" mechanism in text editors; this operation is accomplished by keeping all text changes in a stack.

Undo/Redo stacks in Excel or Word.

Queue is used for Browsing history

New pages are added to history.

Old pages removed such as in 30 days.

Stack/Queue is used in back and forward button of web browser.

As we move from one page to another those pages are placed on stack

URL's of website get stored on stack

Current page is placed on the top of the stack

When we press back button that the elements start popping up and display result in reverse order.

In this way stack is used in forward and backward button of web browser

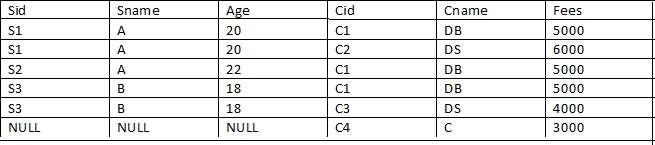
Stack is implemented in form of back button of web browsers : All the pages that we browse at one site are stored in the stack and and we do backward then the top of the stack is popped out and next element is rendered.

Queue : It is used to store the history,all the new pages that we access are stored at the end of the queue and deletion of history takes place in FIFO fashion.  
Stack**-   Move backward to the current position of the web page.**

**-    Undo/Redo**

**Queue  -   Browsing history (to add new pages in history and delete old pages).**

**What are the different issues which may arise due to redundancy in a database relation?**



Insertion Anomalies : In the above relation, due to insertion of some data, some dummy data has to be inserted which creates inconsistency.

For example, suppose we want to add one more course and its details. As no student is enrolled for that course, we need to add dummy details of the student corresponding to that course. This results into inconsistency like if the number of students have to be calculated, the dummy tuple would also be counted resulting into inconsistency.

Deletion Anomalies : Deletion of some data may lead to loss of some other useful data.

For example, Suppose some student details has to be deleted. Now when the required tuple is deleted, the course details corresponding to that student record also gets deleted, which might lead to loss of the details of the course.

Updation Anomalies : If some particular attribute value has to be updated, then it has to be updated in every tuple consisting of that value, which would be too costly operation as all records have to be first transferred to the main memory, and then again transferred to the secondary memory after updation.

Hence, in the design of database, all the above mentioned anomalies should be avoided.

**What all issues should be taken into consideration during the database design ?**

Semantics of Attributes : The relation schema should be designed in such a way that it should have a  clear meaning. Combining attributes from various entity types and relation types into a single relation should be avoided.

Redundancy : Information should be stored only once to minimize the wastage of storage space else the database size would keep on increasing. Redundancy might arise if two or more relations are kept in a single Database Relation.

NULL values: Attributes which have frequent NULL values should be avoided in base relations as NULL values might cause following problems :

Multiple interpretations might corrupt the semantics of relation

The might cause issues in aggregate operations like SUM, COUNT, etc.

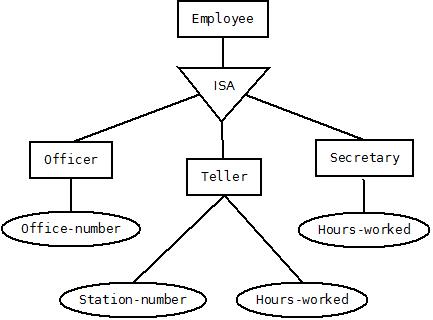
Non Additive join(lossless) property : While joining relations on the basis of the equality conditions on attributes which might be either primary keys or foreign keys , it is guaranteed that no spurious tuples would be generated.

**What is Generalization ?**

Generalization is a bottom up design process. In this process, the higher entities are produced by the union of lower level entities. Generalization is an inverse process of specialization.

Generalization is based on the concept that various entities have some common features, and on the basis of these common features , some entities are grouped together into a generalized higher level entity set. This concept highlights the similarities among lower level entities and hides the differences between them.

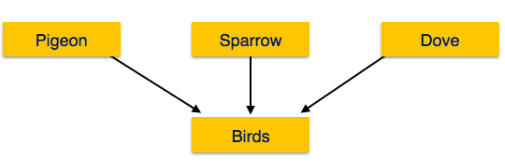
The concept can be explained with the help of an example :



The term Generalization is about bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entity to make further higher level entity.

Eg: if we have a bank account then we have further division which account like type of account so types of account it maybe current or business account also that account current or saving the the bottom up approach lower to high level of entity.

A generalization is an object class, which is a superset of another object class (or classes). Generalization models the “is a” relationship set since members of the specialization class (or classes) are always members of the generalization class. This means that members of the specialization class have all of the same properties of the generalization class including relationships with other objects as well as behaviour.



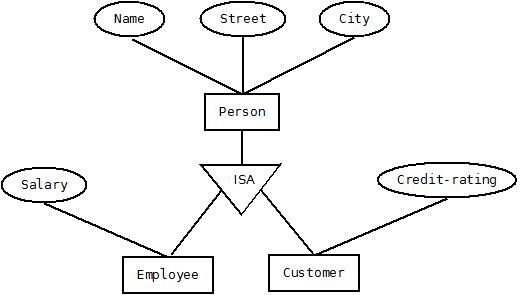
**What is specialization ?**

**Specialization** is a top down design process. Specialization is used to create sub groups within an entity set. The entity sets from which the sub groups are produced are known as higher level entities and the sub groups created are known as lower level entities.

Lower level entities are produced after adding some new attributes to the higher level entities. It can be explained with the help of an example. The entity set person can be specialized to employee and customer.

The lower level entities inherit the characteristics of the higher level entities. Specialization is represented in an ER Diagram with the help of a triangle component with a label**‘ISA’.**

In the below example, the higher level entity ‘person’ had three attributes, name, street and city. The lower level entity ‘employee’ was created by adding new attribute salary and the lower level entity ‘customer’ was created after adding attribute credit-rating. But these lower level entities also inherit the characteristics of person i.e. name street and city.



**Explain Weak Entity Types.**

The entity sets which do not have sufficient attributes to form a primary key are known as **weak entity sets** and the entity sets which have a primary key are known as **strong entity sets**.

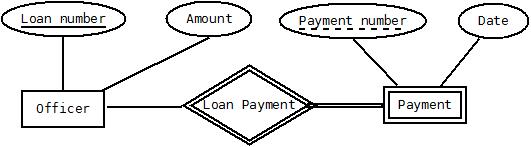
As the weak entities do not have any primary key, they cannot be identified on their own, so they depend on some other entity (known as owner entity). The weak entities have **total participation** **constraint**(existence dependency) in its identifying relationship with owner identity.

For Total Participation Constraint, you may refer https://practice.geeksforgeeks.org/problemSub.php?pid=1400407

Weak entity types have **partial keys**. Partial Keys are set of attributes with the help of which the tuples of the weak entities can be distinguished and identified.

Weak entities are represented with double rectangular box in the ER Diagram and the identifying relationships are represented with double diamond. Partial Key attributes are represented with dotted lines.

In the below ER Diagram , ‘Payment’ is the weak entity. ‘Loan Payment’ is the identifying relationship and ‘Payment Number’ is the partial key. Primary Key of the Loan along with the partial key would be used to identify the records.



The entity sets that do not have primary key of their own. They depend on primary key of another entity set called the strong entity set.

Weak entity is an entity which does not have sufficient attributes to form a primary key.

Weak entity is represented by Double rectangle In an ER Diagram.

Weak entity is always dependent on Strong entities(which have sufficient attribute to form primary key).

**Weak Entity Sets**

They don’t have any primary key of their own so they depend on a primary key from a strong entity set to which they are related.

They must be in a 1:M relation.

A member of a weak entity is called a subordinate entity.

There are two types of weak entities: associative entities and sub-type entitites.

1**. Associative entities :**

An **associative entity** is a term used in relational and entity-relationship theory. A relational database requires the implementation of a base relation (or base table) to resolve many-to-many relationships. This kind of base relation is called an **associative table**.

[https://upload.wikimedia.org/wikipedia/en/1/14/Associate_Entity.png](https://en.wikipedia.org/wiki/File:Associate_Entity.png)

An associative entity (using [Chen](https://en.wikipedia.org/wiki/Peter_Chen) notation)

As mentioned above, associative entities are implemented in a database structure using **associative tables,** which are tables that can contain references to columns from the same or different database tables within the same database.

2. **Sub-type entities  :**

Complete subtype relationship, when all categories are known.

Incomplete subtype relationship, when all categories may not be known.

A classic example of a weak entity without a sub-type relationship would be the "header/detail' records in many real world situations such as claims, orders and invoices, where the header captures information common across all forms and the detail captures information specific to individual items.

You have 5 zeros. Using these 5 zeros and any mathematical functions, you have to get the result of 120.

(0!+0!+0!+0!+0!)!

=5! =120

**Encrypt and Decrypt the password**

we can use SHA-256 or md5 hash functions to encrypt user passwords, and when user tries to login we can compare the hash of password entered and the hash already stored in database to verify the user. This leads to better security as we are not storing passwords as just plaintexts, but rather as hashes.

**Uses of Stack Data structure**

A stack is a container of objects that are inserted and removed according to the last-in first-out (LIFO) principle. In the push down stacks only two operations are allowed: push the item into the stack, and popthe item out of the stack.

MS Word could use the concept of stack in Redo and Undo options (and also in Clipboard), while the browsers might use it in the Back and Forward buttons.

Lets use Stack, Back Stack and Forward Stack.

When stack is empty, disable the button.

When we navigate to new Url, push url on Back Stack. Clear Forward Stack, this is the complicated step in case of Linked List or any other implementation.

When you hit the back button, pop the top Url from Back Stack, push it in Forward Stack.

When you hit the forward button, pop the top Url from Forward Stack and push it on to Back Stack.

This logic can also be used for Undo / Redo functionality. For undo/redo functionality again let us take 2 stacks. One stack for “undoing,” i.e. Going backward to the page we were just at, and one would be for “redoing,” I.e. Going forward.

pseudo code:

    function moveBackward(undo, redo)  
       if undo is not empty then  
           Push current page onto the redo stack  
           Pop the undo stack and go to that page  
    End if  
    End function  
       
    function moveForward (undo, redo)  
         if redo is not empty then  
           Push current page onto the undo stack  
           Pop the redo stack and go to that page  
    End if  
    End function

Another example of stack is if we wished to make some photo editing website and if user wish to add some filter on the photo and there we can see the application of stack. whenever new filter is added it is done on the top of the recent layer so if we wish to remove the edit then it will pop the latest edit and older one will be now on the top. other example can be like online editor where latest edit can be undo.

# GeeksForGeeks

**What are native methods? How do you use them?**

Native methods are methods written in other languages like C, C++, or even assembly language. You can call native methods from Java using JNI. Native methods are used when the implementation of a particular method is present in language other than Java say C, C++. To use the native methods in java we use the keyword native   
public native method\_a(). This native keyword is signal to the java compiler that the implementation of this method is in a language other than java. Native methods are used when we realize that it would take up a lot of rework to write that piece of already existing code in other language to Java.

**How can you minimize the need of garbage collection and make the memory use more effective?**Use object pooling and weak object references.Pooling basically means utilizing the resources efficiently, by limiting access of the objects to only the period the client requires it.  
Increasing utilization through pooling usually increases system performance.  
Object pooling is a way to manage access to a finite set of objects among competing clients. in other words,   
object pooling is nothing but sharing of objects between different clients.  
Since object pooling allows sharing of objects ,the other clients/processes need to re-instantiate the object(which decreases the load time), instead they can use an existing object. After the usage , the objects are returned to the pool.

**What is deep copy?**

**What is shallow copy in JAVA?**

**What is servletrunner?**

**Servletrunner** is a small utility that runs **servlets**. It is included in the JSDK 2.0, while the JSDK 2.1 includes an HTTP server for this purpose.

The **servletrunner** is a small, multithreaded process that handles requests for servlets. Because **servletrunner** is multithreaded, it can be used to run multiple servlets simultaneously or to test one servlet that calls other servlets to satisfy client requests.

**What is the difference between length() and capacity()?**

The current length of a StringBuffer can be found via the method. The total allocated capacity can be found through the capacity() method.

int capacity() Returns the current capacity.

int length() Returns the length (character count).

**What are the uses of this in JAVA?**

It can be used to refer current class instance variable

It can be used to invoke or initiate current class constructor

It can be passed as an argument in the method call

It can be passed as argument in the constructor call

It can be used to return the current class instance

this can be used to refer current class instance variable.

this can be used to invoke current class method (implicitly)

this() can be used to invoke current class constructor.

this can be passed as an argument in the method call.

this can be passed as argument in the constructor call.

this can be used to return the current class instance from the method.

Within the inner class, the keyword **this** holds a reference to the current object but if inner class needs to access the current outer class object then precede the keyword **this** with the outerclass name.

**What are the limitations of static method in JAVA?**

The static method can not use non static data member or call non-static method directly.

this and super cannot be used in static context.

Access only static type data (static type instance variable).

Call only static method ,if non-static then compile time error.

No need the class object to call the static method.

Cant use this and super keyword otherwise compile time error.

If you declare any variable as static, it is known static variable.

The static variable can be used to refer the common property of all objects (that is not unique for each object) e.g. company name of employees,college name of students etc.

The static variable gets memory only once in class area at the time of class loading.

Example:  static String name ="ABC";

Advantage of static variable

It makes your program memory efficient (i.e it saves memory).

**What is the difference between assignment operator and clone() method?**

Assigning an object to another object does not create a duplicate object. It simply assigns a reference of already existing object to a new object.

The **clone()** method when used creates a new object with separate memory space.

For example: **aObj = bObj.clone();**

This statement copies on object **bObj** to new memory location and assign the reference of new object to **aObj.**

**What is finalize in java?**

Finalize is used to perform clean up processing just before object is garbage collected.

Finalize is a method.

It is called by the garbage collector on an object when garbage collection determines that there are no more references to the object.

A subclass overrides the finalize method to dispose of system resources or to perform other cleanup.

This method does not return a value.

It  is a method that is referred as java.lang.Object.finalize()

Java has a automatic garbage collection facility. So, finalize is a method in Java,which is used to free up the memory and  this method is called by garbage collector.

**Difference between transient and volatile keyword?**

VOLATILE

Volatile keyword can also be used in variables to indicate compiler and JVM that always read its value from main memory

It cannot be used along with static keyword

It  guarantees visibility of changes to variables across threads

In volatile: 1) We are only creating instance one time  
2) We are creating instance lazily at the time of the first request comes

You can use Volatile variable if you want to read and write long and double variable atomically. long and double both are 64 bit data type and by default writing of long and double is not atomic and platform dependence.

TRANSIENT:

Transient is used with instance variable to avoid serialization,if a field  is transient its value will not be persisted.

It can be used along with static keyword

Transient variables are initialized with default value during de-serialization and there assignment or restoration of value has to be handled by application code.

 Transient is a variable modifier used in serialization.

If you define any data member as transient, it will not be serialized.

When JVM comes across transient keyword, it ignores original value of the variable and save default value of that variable data type.

It plays an important role to meet security constraints.

Transient keyword is not to serialize the variable whose value can be calculated/derived using other serialized objects or system such as age of a person, current date, etc.

**What is JIT compiler?**

**Just-In-Time(JIT) compiler:**It is used to improve the performance. JIT compiles parts of the byte code that have similar functionality at the same time, and hence reduces the amount of time needed for compilation. Here the term “compiler” refers to a translator from the instruction set of a Java virtual machine (JVM) to the instruction set of a specific CPU.

The just-in-time compiler comes with the virtual machine and is used optionally. It compiles the bytecode into platform-specific executable code that is immediately executed. Sun Microsystems suggests that it's usually faster to select the JIT compiler option, especially if the method executable is repeatedly reused.

The speciality about JIT Compiler is that it keeps a copy of data in cache memory and hence reused it i.e it does not complie the same byte code into machine code again and again  rather uses the saved one.

**Can we define a class inside the interface?**

Yes, definitly we can declare a class inside an interface.  
**Interface I**  
**{**  
**class GfG**  
**{**  
**void m1(){**  
**System.out.println("inside m1");**  
**}**  
**void m2(){**  
**System.out.println("inside m2");**  
**}}}**  
**Class Test{**  
**public static void main(String args[])**  
**I.GfG ia = new I.GfG();**  
**ia.m1();**  
**ia.m2();}**  
In this case, the inner class is becoming as static inner class inside interface where we can access the members of inner class like we do in case of static inner class.

**Why do we need wrapper classes?**

They convert primitive data types into objects. Objects are needed if we wish to modify the arguments passed into a method (because primitive types are passed by value).

The classes in java.util package handles only objects and hence wrapper classes help in this case also.

Data structures in the Collection framework, such as ArrayList and Vector, store only objects (reference types) and not primitive types.

An object is needed to support synchronization in multithreading

public: Public class is visible in other packages, field is visible everywhere (class must be public too)

private : Private variables or methods may be used only by an instance of the same class that declares the variable or method, A private feature may only be accessed by the class that owns the feature.

protected : Is available to all classes in the same package and also available to all subclasses of the class that owns the protected feature. This access is provided even to subclasses that reside in a different package from the class that owns the protected feature.

What you get by default ie, without any access modifier (ie, public private or protected). It means that it is visible to all within a particular package.

# Algorithms(J2B)

Introduction to complexity of algorithm

“How will you calculate complexity of algorithm” is very common question in interview.How will you compare two algorithm? How running time get affected when input size is quite large? So these are some question which is frequently asked in interview.In this post,We will have basic introduction on complexity of algorithm and also to big o notation

What is an algorithm?

An algorithm is step by step instructions to solve given problem.

Lets take a simple example.You want to write an algorithm for listening particular song.

1) Search for song on computer.

2) Is song available?

            i.If Yes,Then listen that song.

           ii.If no,download that song and then listen that song.

So we are solving a problem by step by step procedure.This step by step instructions is called Algorithm.

Why do you need to evaluate an algorithm?

You need to evaluate an algorithm so that you can find most optimize algorithm for solving given problem and also considering various factors and constraints.

For example:

You want to go from city A to City B.Then there are various choices available i.e. by flight,bus or train.So you need to choose among different options depending on your budget and urgency.

Counting number of instructions:

Number of instructions can be different for different programming languages.

Lets count number of instruction for searching a element in a array.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | int n=array.length  for (int i = 0; i < n; i++) {    if(array[i]==elementToBeSearched)  return true;  }  return false; |

Let’s assume our processor takes one instruction for each of the operation:

For assigning a value to a variable

For comparting two values

Multiply or addition

Return statement

In Worst case:

If element which we want to search is last element in sorted array then it will be worst case here.  
So :

|  |  |
| --- | --- |
| 1  2  3  4 | if(array[i]==elementToBeSearched) ,i++ and i<n will be executed n times  int n=array.length,i=0,return true or false will be executed one time. |

Hence f(n)=3n+3

Asymptotic behaviour :

Here We will see how f(n) performs with larger value of n.Now in above function, we have two parts i.e. 3n and 3. Here you can note two points:

As n grows larger, we can ignore constant 3 as it will be always 3 irrespective of value of n. It makes sense as you can consider 3 as initialization constant and different language may take different time for initialization.So other function remains f(n)=3n.

We can ignore constant multiplier as different programming language may compile the code differently. For example array look up may take different number of instructions in different languages. So what we are left with is f(n)=n

How will you compare algorithms?

You can compare algorithms by its rate of growth with input size n

Lets take a example.For solving same problem, you have two functions:  
f(n) =4n^2 +2n+4 and f(n) =4n+4

For f(n) =4n^22 +2n+4  
so here  
f(1)=4+2+4  
f(2)=16+4+4  
f(3)=36+6+4  
f(4)=64+8+4  
….  
As you can see here contribution of n^22 increasing with increasing value of n.So for very large value of n,contribution of n^2 will be 99% of value on f(n).So here we can ignore low order terms as they are relatively insignificant as described above.In this f(n),we can ignore 2n and 4.so  
n^2+2n+4 ——–>n^2For f(n) =4n+4  
so here  
f(1)=4+4  
f(2)=8+4  
f(3)=12+4  
f(4)=16+4  
….  
As you can see here contribution of n increasing with increasing value of n.So for very large value of n,contribution of nwill be 99% of value on f(n).So here we can ignore low order terms as they are relatively insignificant.In this f(n),we can ignore 4 and also 4 as constant multiplier as seen above so  
4n+4 ——–>nSo here n is highest rate of growth.  
Point to be noted :   
We are dropping all the terms which are growing slowly and keep one which grows fastest**.**

Big O Notation:

This notation is used for theoretical measure of  execution  of an algorithm. It gives tight upper bound of a given function. Generally it is represented as f(n)=O(g(n)) and it reads as “f of n is big o of g of n”.

Formal definition:

f(n) = O(g(n)) means there are positive constants c and n0, such that 0 ≤ f(n) ≤ cg(n) for all n ≥ n0. The values of c and n0 must not be depend on n.

When you say O(g(n)) , It means it will never be worst than g(n). Having said that it means O(g(n)) includes smaller or same order of growth as g(n).

So O(n) includes O(n),O(logn) and O(1).

So O(g(n)) is a good way to show complexity of algorithm.

Lets take some example and calculate value for c and n0.  
1. f(n)=4n+3  
Writing in a form of f(n)<=c\*g(n) with f(n)=4n+3 and g(n)=5n

4n+3<=5n for n0=3 and c=5.

or 4n+3<=6n for n0=2 and c=6  
Writing in a form of f(n)<=c\*g(n) with f(n)=4n+3 and g(n)=6n  
so there can be multiple values for n0 and c for which f(n)<=c g(n) will get satisfied.

**2. f(n)=4n^2+2n+4**  
Writing in a form of f(n)<=c\*g(n) with f(n)=4n^2 +2n+4 and g(n)=5n^2  
4n^2 +2n+4<=5n^2 for n0=4 and c=5

Rules of thumb for calculating complexity of algorithm:

Simple programs can be analyzed using counting number of loops or iterations.

**Consecutive statements:**

We need to add time complexity of consecutive statements.

|  |  |
| --- | --- |
| 1  2  3  4 | int m=0; // executed in constant time c1                     m=m+1;  // executed in constant time c2 |

f(n)=c1+c2;  
So O(f(n))=1  
 **Calculating complexity of a simple loop:**

Time complexity of a loop can be determined by running time of statements inside loop multiplied by total number of iterations.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | int m=0; // executed in constant time c1           // executed n times           for (int i = 0; i < n; i++) {               m=m+1;  // executed in constant time c2    } |

f(n)=c2\*n+c1;  
So O(n)=n

**Calculating complexity of a nested loop:**

It is product of iterations of each loop.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | int m=0; executed in constant time c1            // Outer loop will be executed n times     for (int i = 0; i < n; i++) {            // Inner loop will be executed n times    for(int j = 0; j < n; j++)    {     m=m+1; executed in constant time c2    }  } |

f(n)=c2\*n\*n + c1  
So O(f(n))=n^2

**If and else:**

When you have if and else statement, then time complexity is calculated with whichever of them is larger.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | int countOfEven=0;//executed in constant time c1          int countOfOdd=0; //executed in constant time c2             int k=0; //executed in constant time c3  //loop will be executed n times   for (int i = 0; i < n; i++) {            if(i%2==0) //executed in constant time c4              { countOfEven++; //executed in constant time c5                k=k+1; //executed in constant time c6              }            else              countOfOdd++; //executed in constant time c7   } |

f(n)=c1+c2+c3+(c4+c5+c6)\*n  
So o(f(n))=n

Logarithmic complexity

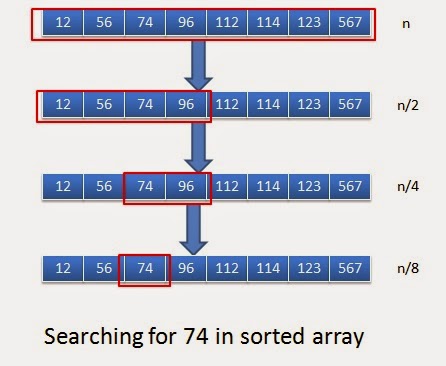
Lets understand logarithmic complexity with the help of example.You might know about binary search.When you want to find a value in sorted array, we use binary search.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | public  int binarySearch(int[] sorted, int first, int last, int elementToBeSearched) {       int iteration=0;       while (first < last) {        iteration++;        System.out.println("i"+iteration);           int mid = (first + last) / 2;  // Compute mid point.           System.out.println(mid);           if (elementToBeSearched < sorted[mid]) { last = mid; // repeat search in first half. } else if (elementToBeSearched > sorted[mid]) {               first = mid + 1;  // Repeat search in last half.           } else {               return mid;     // Found it. return position           }       }       return -1;    // Failed to find element  } |

Now let’s assume our soreted array is:

|  |  |
| --- | --- |
| 1  2  3 | int[] sortedArray={12,56,74,96,112,114,123,567}; |

and we want to search for 74 in above array. Below diagram will explain how binary search will work here.

[](https://www.java2blog.com/wp-content/uploads/2015/06/BinarySearchInJava.jpg)

When you observe closely, in each of the iteration you are cutting scope of array to the half. In every iteration, we are overriding value of first or last depending on soretedArray[mid].  
So for  
0th iteration : n  
1th iteration: n/2  
2nd iteration n/4  
3rd iteration n/8.  
Generalizing above equation:  
For ith iteration : n/2i

So iteration will end , when we have 1 element left i.e. for any i, which will be our last iteration:  
1=n/2i;  
2i=n;  
after taking log  
i= log(n);  
so it concludes that number of iteration requires to do binary search is log(n) so complexity of binary search is log(n)  
It makes sense as in our example, we have n as 8 . It took 3 iterations(8->4->2->1) and 3 is log(8).  
So If we are dividing input size by k in each iteration,then its complexity will be O(logk(n)) that is log(n) base k.

Lets take an example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | int m=0;           // executed log(n) times           for (int i = 0; i < n; i=i\*2) {               m=m+1;    } |

Complexity of above code will be O(log(n)).

Exercise:

Lets do some exercise and find complexity of given code:

**1.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | int m=0;           for (int i = 0; i < n; i++) {               m=m+1;    } |

**Ans:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | int m=0;           // Executed n times           for (int i = 0; i < n; i++) {               m=m+1;    } |

Complexity will be O(n)

**2.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | int m=0;           for (int i = 0; i < n; i++) {               m=m+1;    }     for (int i = 0; i < n; i++) {         for(int j = 0; j < n; j++)               m=m+1;    }  } |

**Ans:**

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | | int m=0;         // Executed n times           for (int i = 0; i < n; i++) {               m=m+1;    }      // outer loop executed n times     for (int i = 0; i < n; i++) {  // inner loop executed n times         for(int j = 0; j < n; j++)               m=m+1;    } |
| 1  2  3 | } | |

Complexity will be :n+n\*n —>O(n^2)

**3.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | int m=0;        // outer loop executed n times     for (int i = 0; i < n; i++) {  // middle loop executed n/2 times         for(int j = n/2; j < n; j++)            for(int k=0;k\*k < n; k++ )               m=m+1;    }          }  } |

**Ans:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | int m=0;        // outer loop executed n times     for (int i = 0; i < n; i++) {  // middle loop executed n/2 times         for(int j = n/2; j < n; j++)        // inner loop executed log(n) times            for(int k=0;k\*k < n; k++ )               m=m+1;    }         }  } |

Complexity will be n\*n/2\*log(n)–> n^2log(n)

**4.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | int m=0;         for (int i = n/2; i < n; i++) {         for(int j = n/2; j < n; j++)            for(int k=0;k < n; k++ )               m=m+1;    } |

Ans:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | int m=0;        // outer loop executed n/2 times     for (int i = n/2; i < n; i++) {  // middle loop executed n/2 times         for(int j = n/2; j < n; j++)        // inner loop executed n times            for(int k=0;k < n; k++ )               m=m+1;    } |

Complexity will be n/2\*n/2\*n –> n^3

# Sr Software Engineer

How a claint program can know the request reached to server?