1. Which one is best array list or set

The set will give much better performance (O(n) vs O(n^2) for the list), and that's normal because avoiding duplicates is the *very purpose* of a set.

Contains for a HashSet is O(1) compared to O(n) for a list, therefore you should never use a list if you often need to run contains.

To add an element : ArrayList is faster than Set  
To remove an element Set is faster than ArrayList.   
The reason is that on ArrayList and Vector, the remove() method will result in a System.arraycopy() call if any element except the last element is removed (the last element being the element with index: size – 1). Removing the first element means the entire rest of the array is copied which is an O(n) operation. Since the test removes all the elements in the List, the full test becomes O(n^2) (slow.)  
HashSet remove does not do any such array copies so it’s remove is O(1) or constant time. For the full test it then is O(n) (fast.). If the tests were rewritten to remove the last element from the ArrayList and Vector, you would likely similar performance to the HashSet.

1. Difference between abstract and interface ?

    (Main difference is methods of a Java interface are implicitly abstract and cannot have implementations. A Java abstract class can have instance methods that implements a default behavior. Variables declared in a Java interface is by default final. An abstract class may contain non-final variables.)

1. abstract class is a class while the interface is an interface, means by extending the abstract class you can not extend another class because Java does not support multiple inheritances but you can implement multiple inheritance in Java.
2. *you can not create a non-abstract method in an interface*, every method in an interface is by default abstract, but you can create a non-abstract method in abstract class. Even a class which doesn't contain any abstract method can be made abstract by using the abstract keyword.
3. interface is better suited for Type declaration and abstract class is more suited for code reuse and evolution perspective.
4. abstract class are slightly faster than interface because interface involves a search before calling any overridden method in Java. This is not a significant difference in most of the cases but if you are writing a time critical application then you may not want to leave any stone unturned.
5. when you add a new method in existing interface it breaks all its implementation and you need to provide an implementation in all clients which is not good. By using an abstract class you can provide a default implementation for a new method in the superclass without breaking existing clients.

|  |  |
| --- | --- |
| Abstract class | Interface |
| 1) Abstract class can have abstract and non-abstractmethods. | Interface can have only abstract methods. Since Java 8, it can have default and static methods also. |
| 2) Abstract class doesn't support multiple inheritance. | Interface supports multiple inheritance. |
| 3) Abstract class can have final, non-final, static and non-static variables. | Interface has only static and final variables. |
| 4) Abstract class can provide the implementation of interface. | Interface can't provide the implementation of abstract cla |

1. **Difference between Set and list ?**

1. **What is base class?**
2. **What is the use of abstract class?**
3. **Explain local, instance and static variable?**
4. **What is checked and unchecked exception and give examples**
5. **Explain the Hashmap?**

important points about Java HashMap class are:

* A HashMap contains values based on the key.
* It contains only unique elements.
* It may have one null key and multiple null values.
* It maintains no order.

1. **Tell me the difference between ArrayList and LinkedList? Which one is better?**

|  |  |
| --- | --- |
| **rrayList** | **LinkedList** |
| 1) ArrayList internally uses dynamic array to store the elements. | LinkedList internally uses doubly linked list to store the elements. |
| 2) Manipulation with ArrayList is slow because it internally uses array. If any element is removed from the array, all the bits are shifted in memory. | Manipulation with LinkedList is faster than ArrayList because it uses doubly linked list so no bit shifting is required in memory. |
| 3) ArrayList class can act as a list only because it implements List only. | LinkedList class can act as a list and queue both because it implements List and Deque interfaces. |
| 4) ArrayList is better for storing and accessing data. | LinkedList is better for manipulating data. |

1. **Can static methods be  overloaded and overridden**

No, you can't override the static method because they are the part of class not object.

Yes., we can overload static methods.

1. **Write code to remove duplicates and sort in an integer array**
2. **What is a linked list?**

**i**mportant points about Java LinkedList are:

* Java LinkedList class can contain duplicate elements.
* Java LinkedList class maintains insertion order.
* Java LinkedList class is non synchronized.
* In Java LinkedList class, manipulation is fast because no shifting needs to be occurred.
* Java LinkedList class can be used as list, stack or queue.

1. **Stringbuilder and stringbuffer**

|  |  |  |
| --- | --- | --- |
| No. | StringBuffer | StringBuilder |
| 1) | StringBuffer is *synchronized* i.e. thread safe. It means two threads can't call the methods of StringBuffer simultaneously. | StringBuilder is *non-synchronized* i.e. not thread safe. It means two threads can call the methods of StringBuilder simultaneously. |
| 2) | StringBuffer is *less efficient* than StringBuilder. | StringBuilder is *more efficient* than StringBuffer. |

1. **difference b/w exception and wrapper class**
2. **Multithreading**

synchronization is one byone

Multithreading in java is a process of executing multiple threads simultaneously.

Thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

But we use multithreading than multiprocessing because threads share a common memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.

Java Multithreading is mostly used in games, animation etc.

Thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

) It doesn't block the user because threads are independent and you can perform multiple operations at same time.

2) You can perform many operations together so it saves time.

3) Threads are independent so it doesn't affect other threads if exception occur in a single thread.

Advantages of Java Multithreading

1) It doesn't block the user because threads are independent and you can perform multiple operations at same time.

2) You can perform many operations together so it saves time.

3) Threads are independent so it doesn't affect other threads if exception occur in a single thread

Threads can be created by using two mechanisms :

1. Extending the Thread class

2. Implementing the Runnable Interface

say it

Synchronized blocks in Java are marked with the synchronized keyword. Asynchronized block in Java is synchronized on some object. All synchronizedblocks synchronized on the same object can only have one thread executing inside them at a time. ... This synchronization is implemented in Java with a concept called monitors.

1. **Exception**

Java - Exceptions. An exception (or exceptional event) is a problem that arises during the execution of a program. When an Exception occurs the normal flow of the program is disrupted and the program/Application terminates abnormally, which is not recommended, therefore, these exceptions are to be handled.

Java exception handling is managed via five keywords: try, catch, throw, throws, and finally. Briefly, here is how they work. Program statements that you think can raise exceptions are contained within a try block. If an exception occurs within the try block, it is thrown. Your code can catch this exception (using catch block) and handle it in some rational manner. System-generated exceptions are automatically thrown by the Java run-time system. To manually throw an exception, use the keyword throw. Any exception that is thrown out of a method must be specified as such by a throws clause. Any code that absolutely must be executed after a try block completes is put in a finally block.

try-catch – We use try-catch block for exceptionhandling in our code. try is the start of the block andcatch is at the end of try block to handle theexceptions. We can have multiple catch blocks witha try and try-catch block can be nested also. catchblock requires a parameter that should be of typeException

In a method, there can be more than one statements that might throw exception, So put all these statements within its own try block and provide separate exception handler within own catch block for each of them.

If an exception occurs within the try block, that exception is handled by the exception handler associated with it. To associate exception handler, we must put catch block after it. There can be more than one exception handlers. Each catch block is a exception handler that handles the exception of the type indicated by its argument. The argument, ExceptionType declares the type of the exception that it can handle and must be the name of the class that inherits from Throwableclass.

For each try block there can be zero or more catch blocks, but only one finally block.

The finally block is optional.It always gets executed whether an exception occurred in try block or not . If exception occurs, then it will be executed after try and catch blocks. And if exception does not occur then it will be executed after the try block. The finally block in java is used to put important codes such as clean up code e.g. closing the file or closing the connection.

Advantage of exception handling

Exception handling ensures that the flow of the program doesn’t break when an exception occurs.

For example, if a program has bunch of statements and an exception occurs mid way after executing certain statements then the statements after the exception will not execute and the program will terminate abruptly.

exception can also be thrown to the calling method using throws keyword

1. Difference between Java 7 and Java 8.
2. What is new in Java9

Coding:-

1. Remove duplicate from Array

public class RemoveduplicateArray {

           // Function to remove duplicate elements

           // This function returns new size of modified

           // array.

           static int removeDuplicates(int arr[], int n)

           {

               // Return, if array is empty

               // or contains a single element

               if (n==0 || n==1)

                   return n;

               int[] temp = new int[n];

               // Start traversing elements

               int j = 0;

               for (int i=0; i<n-1; i++)

                   // If current element is not equal

                   // to next element then store that

                   // current element

                   if (arr[i] != arr[i+1])

                       temp[j++] = arr[i];

               // Store the last element as whether

               // it is unique or repeated, it hasn't

               // stored previously

               temp[j++] = arr[n-1];

               // Modify original array

               for (int i=0; i<j; i++)

                   arr[i] = temp[i];

               return j;

           }

           public static void main (String[] args)

           {

               int arr[] = {1, 2, 2, 3, 4, 4, 4, 5, 5};

               int n = arr.length;

               n = removeDuplicates(arr, n);

               // Print updated array

               for (int i=0; i<n; i++)

                  System.out.print(arr[i]+" ");

           }

    }

**2)**

for(int i=0;i<=10;i++) {

        System.out.print(++i);

    }

**Ans:- 1357911**

**3)**

for(int i=0;i<=10;i++) {

    System.out.print(i++);

}

**Ans:- 0246810**

-------------------------Gomathy -wipro/visa---10/27----------------------------------

**Garbage Collector Que:---**

**Does java support GC?**

yes

**Does it do call automatically or need to call?**

It does automatically but if you force to call you can call using system.gc(). It will wait for gc thread in JVM  at that time only  it will run. Suddenly you cannot.

**Can you force GC?**

Yes using system.gc() it have less priority compare to all thread.

**When you force GC Does it Swipe immediately?**

No it will wail gc thread in JVM.

**Garbage Collection In Java :**Whenever you run a java program, JVM creates three threads. 1) main thread   2) Thread Scheduler   3) Garbage Collector Thread. In these three threads, main thread is a user thread and remaining two are daemon threads which run in background.  
  
The task of main thread is to execute the main() method. The task of thread scheduler is to schedule the threads. The task of garbage collector thread is to sweep out abandoned objects from the heap memory. Abandoned objects or dead objects are those objects which does not have live references. Garbage collector thread before sweeping out an abandoned object, it calls finalize() method of that object. After finalize() method is executed, object is destroyed from the memory. That means clean up operations which you have kept in the finalize() method are executed before an object is destroyed from the memory.  
  
Garbage collector thread does not come to heap memory whenever an object becomes abandoned. It comes once in a while to the heap memory and at that time if it sees any abandoned objects, it sweeps out those objects after calling finalize() method on them. Garbage collector thread calls finalize() method only once for one object.

**Where you used inheritance.?(multiple and multilevel)**

**How many parent and child class in java?**

Java only supports Multi-level inheritance. A Java class can have only one base class but multiple parent using interfaces.

**What is Multiple inheritance and Hybrid Inheritance?**

**Is annotation name casesensitive?why?**

**What is annotation?**

**Difference between hashmap and hashtable?**

1) HashMap is non-synchronized. It is not-thread safe and can't be shared between many threads without proper synchronization code.

→ Hashtable is synchronized. It is thread-safe and can be shared with many threads.  
2) HashMap allows one null key and multiple null values.

→     Hashtable doesn't allow any null key or value.  
3) HashMap is a new class introduced in JDK 1.2.

→ Hashtable is a legacy class.  
4) HashMap is fast.

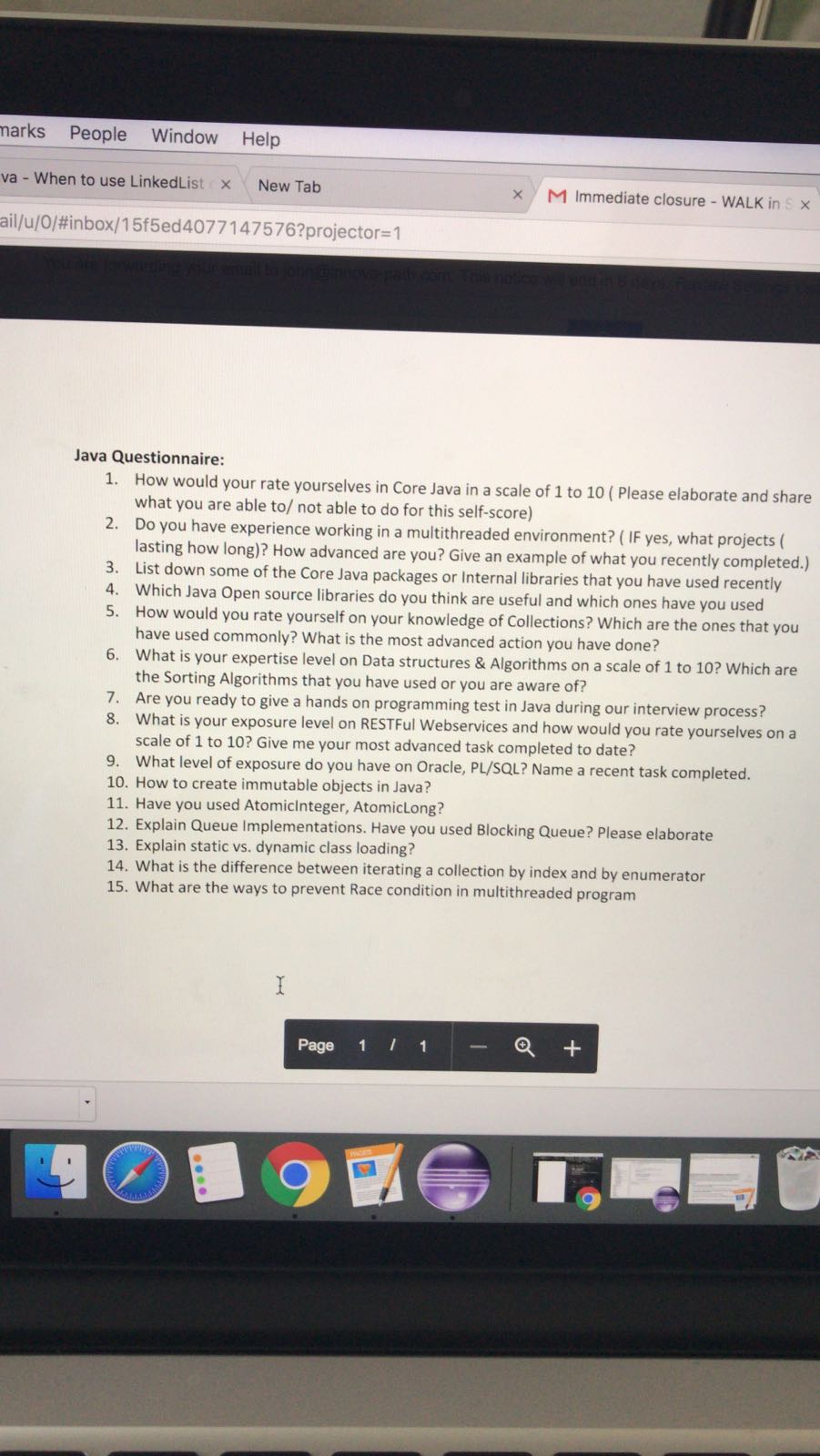
→ Hashtable is slow.  
5) We can make the HashMap as synchronized by calling this code  
Map m = Collections.synchronizedMap(hashMap);

→ Hashtable is internally synchronized and can't be unsynchronized.  
6) HashMap is traversed by Iterator.-->

Hashtable is traversed by Enumerator and Iterator.  
7) Iterator in HashMap is fail-fast.

→     Enumerator in Hashtable is not fail-fast.  
8) HashMap inherits AbstractMap class.

→ Hashtable inherits Dictionary class.



**How would your rate yourselves in Java in a scale of 1 to 10 ( Please elaborate and share what you are able to/ not able to do for this self-score)**

**A)** 6

**Do you have experience working in a multithreaded environment? ( IF yes, what projects ( lasting how long)? How advanced are you? Give an example of what you recently completed.**)

A)I know the concept but didn’t get a chance to work in any of my projects…

**List down some of the Java packages or Internal libraries that you have used recently**

Apache POI, log4J, json

**Which Java Open source libraries do you think are useful and which ones have you used \**

Testng, ApacheHTTPClient, junit, Apache POI, log4J, json —> Used all of them

**How would you rate yourself on your knowledge of Collections? Which are the ones that you have used commonly? What is the most advanced action you have done?**

I would rate myself 8.  The most commonly used ones are List(ArrayList), Set and Map .

Used ArrayList to store list of data from the webelements. Set is used for unique elements. Most used action was List.sort();

**What is your expertise level on Data structures & Algorithms on a scale of 1 to 10? Which are the Sorting Algorithms that you have used or you are aware of?**

7. I am aware of merge sort, bubble sort and insert sort. In my opinion merge sort is the best. Insert sort can be used in Linked List. Most of the times I used Java methods like Arrays.sort() and List.sort() for collections

**Are you ready to give a hands on programming test in Java during our interview process**?

Yes, sure I would like to do.

**What is your exposure level on RESTFul Webservices and how would you rate yourselves on a scale of 1 to 10? Give me your most advanced task completed to date?**

8. In my recent project, I used POSTMAN to test RESTful webservices manually and Used HTTP Client and Java to automate.

**What level of exposure do you have on Oracle, PL/SQL? Name a recent task completed.**

Used SQL for data integrity testing for BI events, captured the SQL statements from the application execution and manually checked the results.

**How to create immutable objects in Java?**

To create immutable objects in Java, it has to be final and private. No getter, setter methods to be used. Don't allow subclasses to override methods. The simplest way to do this is to declare the class as final.

**Have you used AtomicInteger, AtomicLong?**

No. but I do have knowledge about that. An AtomicInteger can be used as a drop-in replacement that provides a way to access counter variable in synchronized blocks/methods.

**Explain Queue Implementations. Have you used Blocking Queue? Please elaborate**

BlockingQueue is a unique collection type which not only store elements but also supports flow control by introducing blocking if either BlockingQueue is full or empty. take() method of BlockingQueue will block if Queue is empty and put() method of BlockingQueue will block if Queue is full. This property makes BlockingQueue an ideal choice for implementing Producer consumer design pattern where one thread insert element into BlockingQueue and other thread consumes it.

**Explain static vs. dynamic class loading?**

Dynamic Class Loading allows the loading of java code that is not known about before a program starts. It is done when the name of the class is not known at compile time.

Static class loading, classes are statically loaded with Java’s "new" operator. In this case, the retrieval of class definition and instantiation of the object is done at compile time.

**What is the difference between iterating a collection by index and by enumerator**

Enumeration is a legacy class. Vector supports enumerator

Iterator can be used with ArrayList, Set and other collection classes.

**What are the ways to prevent Race condition in multithreaded program**

A race condition occurs when two or more threads can access shared data and they try to change it at the same time.

Inorder to avoid the race condition we can synchronize the write and access methods on the shared variables to lock these variables to other threads

-----------------------------------------------F2F Madhavi -Apple-------------------------

1. Write a program on Fibonacci series
2. What is immutability?
3. what is the diff b/w serialization and externalization?
4. how do you make code thread safe? Can we do without synchronization block?

-------------------------------------F2F Manasa-Apple-------------------------------------

1.abstract vs interface

2.Why string is immutable

3. collections - which collections do u often use

4. explain get fn in maps frameworks used?

5. diff btwn 1 way and 2 way encryption  
-------------------------------------F2F Puja Jha-Apple-------------------------------------

1.method overloading and overriding