Collection Interview Questions

**What is collections framework ?**

A framework is set of classes and interfaces to build a functionality. Java collections framework provides set of interfaces and classes for storing and manipulating collections. Collection framework contains classes and interfaces in java.util package and java.util.concurrent packages. Advantages or benefits of Collections framework : 1) High performance 2) Using this framework we can create different types of collections 3) We can create our own collection and we can extend a collection. 4) Reduces programming effort. 5) Increases speed and quality : Collections framework provides high performance, implementations of useful data structures and algorithms.

**What is collection ?**

A collection is a container which holds group of objects. Collection provides a way to manage objects easily. Collections manages group of objects as single unit. Examples include list of strings, integers etc. Here are few basic operations we do on collections : 1) Adding objects to collection. 2) Removing or deleting objects from collection. 3) Retrieving object from collection. 4) Iterating collection. 200) Difference between collection, Collection and Collections in java? collection : represent group of objects where objects are stored. Collection : This is one of the core interface which provides basic functionality for collection. Collections : Collections contains some utility static methods that operate on collections.

**Explain about Collection interface in java ?**

Collection is the fundamental and root interface in Collections framework. Collection extends Iterable interface and inherits iterator method which returns Iterator object. Signature : public interface Collection extends Iterable { } Methods in Collection interface : boolean add(E e); Adds an element to the collection. Returns true if element is added. boolean remove(Object o); Removes an object from collection if that object is present in collection. Return true if matching object is removed from collection. boolean addAll(Collection c); Adds all the elements specified in the collection to this collection.Returns true if all elements are added. boolean removeAll(Collection c); Removes all the elements from this collection that are specified in other collection.Returns true if all the elements are removed. int size(); Returns number of elements in collection. boolean isEmpty(); Checks whether collection contains elements or not. If no elements are present it returns false. boolean contains(Object o); Checks whether specified object is in collection or not. Return true if object is in collection. Iterator iterator(); Used to iterator over collection. No guarantee on order of elements iterated. boolean retainAll(Collection c); Removes all the elements which are not in specified collection. Returns only elements specified in collection removing other elements. Object[] toArray(); Returns an array of elements in collection.

**List the interfaces which extends collection interface ?**

1) List 2) Set 3) Queue 4) Deque ( From Java 6) 203) Explain List interface ? List interface extends collection interface used to store sequence of elements in collection. We can even store duplicate elements in list. We can insert or access elements in list by using index as we do in arrays. List is an ordered collection. The main difference between List and non list interface are methods based on position. 38 Some of the operations we can perform on List : 1) Adding an element at specified index. 2) Removing an element at specified index. 3) To get the index of element List contains some specific methods apart from Collection interface methods.

**Explain methods specific to List interface ?**

boolean addAll(int index, Collection c); This method inserts all the elements in specified collection to the list at specified position. E get(int index); This method returns an element at specified position in the list. E set(int index, E element); This method replaces the element at specified position in the list with the specified element. void add(int index, E element); This method inserts the specified element with the index specified. E remove(int index); This method removes the element at specified index and returns the element removed. int indexOf(Object o); indexOf() method returns the index of last occurrence of specified element. If there is no element in the list it removes the element. ListIterator listIterator(); Returns a list iterator of elements in list. List subList(int fromIndex, int toIndex); This method returns list of elements between indexes specified.

**List implementations of List Interface ?**

1) ArrayList 2) Vector 3) LinkedList

**Explain about ArrayList ?**

ArrayList is an ordered collection which extends AbstractList and implements List interface. We use ArrayList mainly when we need faster access and fast iteration of elements in list. We can insert nulls in to arraylist. Arraylist is nothing but a growable array. public class ArrayList extends AbstractList implements List, RandomAccess, Cloneable, java.io.Serializable{} From java 1.4 ArrayList implements RandomAccess interface which is a marker interface which supports fast and random access. Advantages : 1) Faster and easier access. 2) Used for Random access of elements. Drawbacks : 1) We cannot insert or delete elements from middle of list.

**Difference between Array and ArrayList ?**

Arrays are used to store primitives or objects of same type or variables that are subclasses of same type. ArrayList : It is an ordered collection which grows dynamically. In list we can insert nulls values and list allows duplicate elements. ARRAY ARRAY LIST 1) While creating array we have to know the size. 1) But it is not required to know size while creating ArrayList, because arraylist grows dynamically. 2) To put an element in to array we use the following syntax :String array[] = newString[5];array[1] = “java”;We must know specific location to insert an element in to 2) We can add element to arraylist with following syntax :List stringList = new ArrayList();stringList.add(“java”); 39 array. If we try to put element in index which is out of range we get ArrayIndexOutOfBounds Exception 3) Arrays are static 3) ArrayList is dynamic 4) We can store objects and primitives 4) We can store only primitives prior to 1.5 . From 1.5 we can store even objects also. 5) We have to manually write logic for inserting and removing elements. 5) Just a method call would add or remove elements from list. 6) Arrays are faster 6) Arraylist is slower. 7) Arraylist is implemented using arrays.

**What is vector?**

Vector is similar to arraylist used for random access. Vector is a dynamic array like arraylist. vector size increases or decreases when elements are added and removed . Vector is synchronized . vector and Hashtable are the only collections since 1.0. Rest of the collections are added from 2.0. public class Vectorextends AbstractListimplements List, RandomAccess, Cloneable, java.io.Serializable

**Difference between arraylist and vector ?**

Both ArrayList and vector grows dynamically. The differences between arraylist and vector are : 1) Arraylist is not synchronized and vector is synchronized. 2) Vector is legacy collection introduced in 1.0 and Arraylist introduced in java 2.0. Performance wise it is recommended to use arraylist rather than vector because by default vector is synchronized which reduces performance if only one thread accesses it.

**In which order the Iterator iterates over collection?**

The order in which Iterator will iterate the collection depends on the traversal order of collection. For example : for list traversal order will be sequential, and for set the order cannot be determined, and for sorted sorted set will sort the elements in sorted order. So it all depends on the collection in which order iterator iterates.

**Explain ListIterator and methods in ListIterator?**

List Iterator is similar to Iterator but ListIterator is bidirectional. We can traverse through the collection in either forward or backward direction. List Iterator extends Iterator and all the methods in Iterator will be there in ListIterator too with some additional methods . List Iterator doesn’t have current element .Position of List Iterator lies between two elements i.e previous element and next element. Features of ListIterator : 1) Traversal of List in either direction.

2) Modification of its elements.

3) Access to elements position.

Signature : public interface ListIterator extends Iterator { } ListIterator methods : Method Description Void add(E obj) Inserts element in to the list infront of the element returned by call to next() and after the element returned by call to next(). boolean hasNext(); Returns true if there are more elements in the list instead of throwing exception if there are no elements. E next(); Returns the next element . NoSuchElementException is thrown if there is no next element. boolean hasPrevious(); Returns true if there are elements when iterating list in reverse direction. E previous(); Returns the previous element in the list. int nextIndex(); Returns the index of the element returned by next() method. If there are no elements it returns the size of the list. int previousIndex(); Returns the index of the element returned by previous() method. If there are no elements it returns the size of the list. Returns -1 if the iterator is at beginning of list. void remove(); Removes the element that was returned by calling next() or previous(). An Illegal state Exception will be thrown if remove() is called before next() or previous(). void set(E e); This method replaces an element in the list with the specified element.

**Explain about Sets ?**

A set is a collection which does not allow duplicates. Set internally implements equals() method which doesn’t allow duplicates.Adding an duplicate element to a set would be ignored .Set interface is implemented in java.util.set package.Set interface does not have any additional methods . It has only collection methods. A set can contain atmost one null value. ArrayList is an ordered collection.In arraylists order remains same in which they are inserted. But coming to set it is an unordered collection. public interface Set extends Collection { } Important operations that can be performed on set : 1) Adding an element to set. 2) Removing an element from set. 3) Check if an element exist in set. 4) Iterating through set.

**Implementations of Set interface ?**

1) HashSet 2) Linked HashSet 3) TreeSet

**Explain HashSet and its features ?**

Hashset implements set interface and extends AbstractSet. Features of Hashset are : 1) It does not allow duplicates. 2) It does not gurantee ordering of elements. 3) It is unsorted and unordered set. 4) Performance wise it is recommended to use hashset when compared to other sets because it internally uses hashing mechanism. 5) Allows insertion of nulls. Note : For efficiency whenever objects are added to HashSet it need to implement the hashCode() method. public class HashSet extends AbstractSet implements Set, Cloneable, java.io.Serializable { }

**Explain Tree Set and its features?**

TreeSet implements navigableSet interface and extends Abstract set.It creates collection that uses tree for storage. Features of Treeset are : 1) It does not allow duplicates. 2) When we retrieve the elements in treeset we will get elements in sorted order. 42 public class TreeSet extends AbstractSet implements NavigableSet, Cloneable, java.io.Serializable { 217) When do we use HashSet over TreeSet? If we want to search for an element in collection and does not want any sorting order we go for HashSet. 82) When do we use TreeSet over HashSet? TreeSet is preferred 1) if elements are to be maintained in sorting order. 2) Fast insertion and retrieval of elements.

**What is Linked HashSet and its features?**

LinkedHashSet extends HashSet and implements Set interface. public class LinkedHashSet extends HashSet implements Set, Cloneable, java.io.Serializable { } Linked HashSet is similar to HashSet but in linked HashSet we maintain order but in HashSet we don’t maintain order. Maintaining order means elements will be retrieved in order which they are inserted.

**Explain about Map interface in java?**

A map is an association of key-value pairs. Both keys and values in map are objects. Features of map : 1) Maps cannot have duplicate keys but can have duplicate value objects.

**What is linked hashmap and its features?**

LinkedHashMap extends HashMap and implements Map.lLinked hashmap gurantees order of elements . Elements are retrieved in same order they are inserted.Linked HashMap uses internally double linked lists to keep insertion order. The differences between Hashmap and linked hashmap is 1) LinkedHashMap maintains the insertion order while HashMap doesnot maintain order. 2) HashMap if faster for insertion and deletion of elements when compared to linked hashmap. Linked hashmap is preferred only for faster iteration of elements. public class LinkedHashMap extends HashMap implements Map { }

**What is SortedMap interface?**

SortedMap extends Map interface.Sorted Map maintains sorted order of keys in a map. By default sorted map maintains natural ordering if we want custom order we can specify using comparator. public interface SortedMap extends Map { } 222) What is Hashtable and explain features of Hashtable? Hashtable was available before collection framework. When collection framework was started Hashtable extends Dictionary class and Map interface. Hashtable offers a convenient way of storing key/ value pairs. Hashtable does not allow nulls either keys or values. Hashtable is synchronized.

**Difference between HashMap and Hashtable?**

Difference HashMap Hashtable Synronization HashMap is not synchronized. Hashtable is synchronized. Nulls HashMap allows atmost one null key and any number of null values. Hashtable does not allow null values. Performance Since HashMap is not synchronized its performance is faster than Hashtable. Performance is slower when compared to HashMap. Introduction HashMap introduced starting from Hashtable is even before collection 43 collection framework. framework.

**Difference between arraylist and linkedlist**?

Difference Arraylist LinkedList Access Implements RandomAccess interface we can search randomly all the elements in the list. It extends Abstract sequential List interface which provides sequential access to elements. Searching and retrieval of elements Searching and retrieval of elements is fast since arraylist provides random access. Searching and retrieval of elements is slow because of sequential access to elements. Addition and removal of elements Adding and removal of elements in random positions is slow.For example if we want to add element to middle of the list we have to move the elements in the list and then we need to insert the element. Similarly for removing the element we need to follow the same thing. Adding and removal of elements in random positions is fast because there is no need of resizing the array just by updating the node structures with new addresses.

**Difference between Comparator and Comparable in java?** Sno Comparator Comparable 1. Defined in java.util package Defined in java.lang package. 2. Comparator interface is used when we want to compare two different instances Comparable is used to compare itself with other instance. 3. Comparator is used when we want custom sorting.Ex : If we take employee class sorting by employeeId is natural sorting. Comparable is used for natural sorting of objects.Ex : If we take employee class sorting by ename and age we can say as custom sorting. 4. Should override int compare(T o1, T o2) method which takes two instances. Should override public int compareTo(T o) method which takes one instance. 5. For sorting objects we use collections.sort(list,new Comparator); For sorting objects we use collections.sort(list);

**What is concurrent hashmap and its features ?**

Concurrent HashMap is implemented in java.util.concurrent package. Concurrent HashMap extends Abstract Map and implements concurrent Map. Concurrent HashMap is used in multi threaded environment. ]It is similar to Hashtable and synchronized version of hashmap but with minor differences. Concurrent HashMap does not allow null keys and values.

**Difference between Concurrent HashMap and Hashtable and collections.synchronizedHashMap?**

Locking Mechansim :ConcurrentHashMap uses completely different hashing mechanism called lock striping which offers better concurrency and scalability. The main advantage of this mechanism is better concurrency instead of synchronizing every method by using common lock which allows only one thread to access at a time, it allows better concurrency by allowing multiple threads to access. ConcurrentModificationException :ConcurrentHashMap provides iterators which doesnot throw concurrent modification exception which allows only one thread to access iterator, while synchronized map may throw concurrent modification exception. 228) Explain copyOnWriteArrayList and when do we use copyOnWriteArrayList? copyOnWriteArrayList is used in multithreaded environment. If we want to iterate over arraylist ,but the arraylist is updated by other threads to prevent concurrent modification exception we have two solutions : 1) First one is we need to synchronize the list manually by using collections.synchronized(list) and iterate over the list in synchronized block to avoid concurrent modification exception. 2) The second one is to use copyOnWriteArrayList which takes care of concurrency. The advantage of using copyOnWriteArrayList is no need to synchronize list explicitly. So when we use copyOnWriteArrayList when a thread modifies the list while the other thread was iterating it does not modify original list but creates a copy of list with modified contents so that the iterator won’t know the modifications made to original list.

**Explain about fail fast iterators in java?**

When iterator iterates over collection, collection should not be modified except by that iterator. Modification means collection cannot be modified by thread when other thread is iterating, if such modification happens a concurrent modification exception will be thrown.Such kind of iterators are fail fast iterators. Ex : ArrayList,HashSet,HashMap. Almost all the iterators implemented in collections framework are fail fast.

**Explain about fail safe iterators in java?**

Fail safe iterators are iterators which does not throw concurrent modification exception, when one thread modifies collection and other thread in the process of iterating the collection. It does not throw concurrent modification exception because when other thread was iterating it does not modify original list but creates a copy of list with modified contents so that the iterator won’t know the modifications made to original list. Ex : copyOnWriteArrayList Core java Serialization interview questions

**What is serialization in java?**

Serialization is the process of converting an object in to bytes, so that it can be transmitted over the network,or stored in a flat file and can be recreated later. Serialized object is an object represented as sequence of bytes that includes objects data, object type, and the types of data stored in the object.

**What is the main purpose of serialization in java?**

The main uses of serialization are : 1) Persistence: We can write data to a file or database and can be used later by deserializing it. 2) Communication : To pass an object over network by making remote procedure call. 3) Copying : We can create duplicates of original object by using byte array. 4) To distribute objects across different JVMs.

**What are alternatives to java serialization?**

XML based data transfer JSON based data transfer. XML based data transfer : We can use JIBX or JAXB where we can marshall our object’s data to xml and transfer data and then unmarshall and convert to object. JSON based transfer : We can use json to transfer data. 234) Explain about serializable interface in java? To implement serialization in java there is an interface defined in java.io package called serializable interface. Java.io.Serializable interface is an marker interface which doesnot contain any any methods. A class implements Serializable lets the JVM know that the instances of the class can be serialized. Syntax: public interface Serializable { }

**How to make object serializable in java?**

1) Our class must implement serializable interface.If our object contains other objects those class must also implement serializable interface. 2) We use ObjectOutputStream which extends OutputStream used to write objects to a stream. 3) We use ObjectInputStream which extends InputStream used to read objects from stream

**What is serial version UID and its importance in java?**

Serial version unique identifier is a 64 bit long value .This 64 bit long value is a hash code of the class name,super interfaces and member. Suid is a unique id no two classes will have same suid. Whenever an object is serialized suid value will also serialize with it. When an object is read using ObjectInputStream, the suid is also read. If the loaded class suid does not match with suid read from object stream, readObject throws an InvalidClassException.

**What happens if we don’t define serial version UID ?**

If we don’t define serial version UID JVM will create one suid for us. But it is recommended to have suid rather than JVM creating because at run time JVM has to compute the hashcode of all the properties of class. This process makes serialization low. We can’t serialize static fields one exception to this is suid where suid gets serialized along with the object. Ex :private static final long serialVersionUID = -5885568094444284875L;

**Can we serialize static variables in java?**

We can’t serialize static variables in java. The reason being static variable are class variables that belongs to a class not to object, but serialization mechanism saves only the object state not the class state.

**When we serialize an object does the serialization mechanism saves its references too?**

When we serialize an object even the object it refers must implement serializable then the reference objects also get serialized. If we don’t make reference objects serializable then we get NotSerializableException.

**If we don’t want some of the fields not to serialize How to do that?**

If we don’t want to serialize some fields during serialization we declare those variables as transient. During deserialization transient variables are initialized with default values for primitives and null for object references