Collection Interview Notes

1. Difference between array and collection?

Array	Collections
Array is the indexed based collection of homogeneous datatypes. 1. In array we can store only homogeneous data. 2. Arrays are fixed in size. 3. There is no underlying data structure present for arrays so readymade method support is not available for arrays, we have to write our own logic for every operation explicitly. 4. We can store both primitive and object type of data in array. 5. Performance wise arrays performance is faster.	1. Collections is the group of individual object represented as a single entity. 2. We can store both homogeneous and heterogeneous data in a collection. 3. Collections are not fixed in size they are grow-able in nature. 4. Underlying data structure is present for collections so there is predefined method support is available and we don't have to write our own logic to do every operation. 5. We can store only objects in collections. 6. Performance wise collections are slower than arrays.

2. Difference between collection and collection framework?

Collection	Collection Framework
1. Collection is the group of individual object represented as a single entity. 2. Collection is like a container which contains object.	1. It contains several classes and interfaces which can be used to represent a group of individual objects as a single entity. 2. Collection framework is a library which holds different types of methods and interfaces.

3. What are the 9 key interfaces of collection framework?

i. Collection
ii. List
iii. Set
iv. Sorted set
v. Navigable set
vi. Queue
vii. Map
viii. Sorted Map
ix. Navigable Map

4. Which class implements collection interface directly?

- There is no such class which implements collection interface directly.

5. What is the difference between collection and collections?

- Collection is an interface and Collections is a utility class present in java.util package.

- Collection represent a group of individual object as a single entity and Collections is a class which defines several predefined utility methods for collection objects (like sorting, searching etc…).

6. Define List Interface?

- It is the child interface of collection.
- If we want to represent a group of individual object as a single entity where duplicates are allowed and insertion order must be preserved then we should go for list interface.

7. Which are the Implementation classes of list interface?

- Array List , Linked List , Vector , Stack
- Vector and Stack also known as legacy classes because they came in 1.0 version of java and rest implementation classes of List interface came in 1.2 version.

8. Define Set Interface?

- It is the child interface of collection.
- If we want to represent a group of individual objects as a single entity where duplicates are not allowed and insertion order is not required then we should go for set interface.

9. Which are the implementation classes of Set interface?

- HashSet, Linked HashSet.
- Linked HashSet came in 1.4 version of java but HashSet came in 1.2 version of java.

10. When should we use sortedset?

- Sortedset is the child interface of set interface.
- If we want to represent a group of individual objects as a single entity where duplicates are not allowed but the objects should be inserted according to some sorting order then we will use sortedset interface.
- It came in 1.6 version of java.

11. What is a NavigableSet?

- It is the child interface of sortedset and it contains several methods for navigation purposes.

- It came in 1.6 version of java.

12.Differences between List and Set?

List	Set
i. Duplicates are allowed in List. ii. Insertion order is preserved.	i. Duplicates are not allowed. ii. Insertion order is not preserved.

13.What is Queue interface and when should we use it give an example?

- Queue is the child interface of collection interface.
- If we want to represent a group of individual object as a single entity prior to processing then we can go for queue interface.
- Usually queue follows first in first out order but also we can implement our own priority also based on our requirement

Example:

- Suppose we want to develop an email sending software where before sending mail all the email ids we have to store in some data structure. In which order we added the email ids to the data structure in the same order only the emails should be delivered for this requirement Queue is the best choice.
- Queue interface introduced in java 1.5 version.

14. What are the various implementation classes present in Queue interface?

- Priority Queue, Blocking Queue.
- Blocking Queue also have two child classes priority blocking queue and Linked Blocking queue.

15. Define Map interface?

- It is not the child interface of collection interface.
- If we want to represent a group of objects as a key and value pair then we will use the Map interface.
- Both key and values are objects only.
- Duplicate keys are not allowed but value can be duplicate.

16.Which are the implementation classes of Map?

- HashMap , LinkedHashMap , WeakHashMap , IdentityMap , HashTable , Properties.
- LinkedHashMap is the child class of HashMap.

17.Which legacy classes are present in Map?

- Dictionary, HashTable and properties are the legacy classes present in HashMap.
- Dictionary is an abstract class.

18.What is a sorted Map?

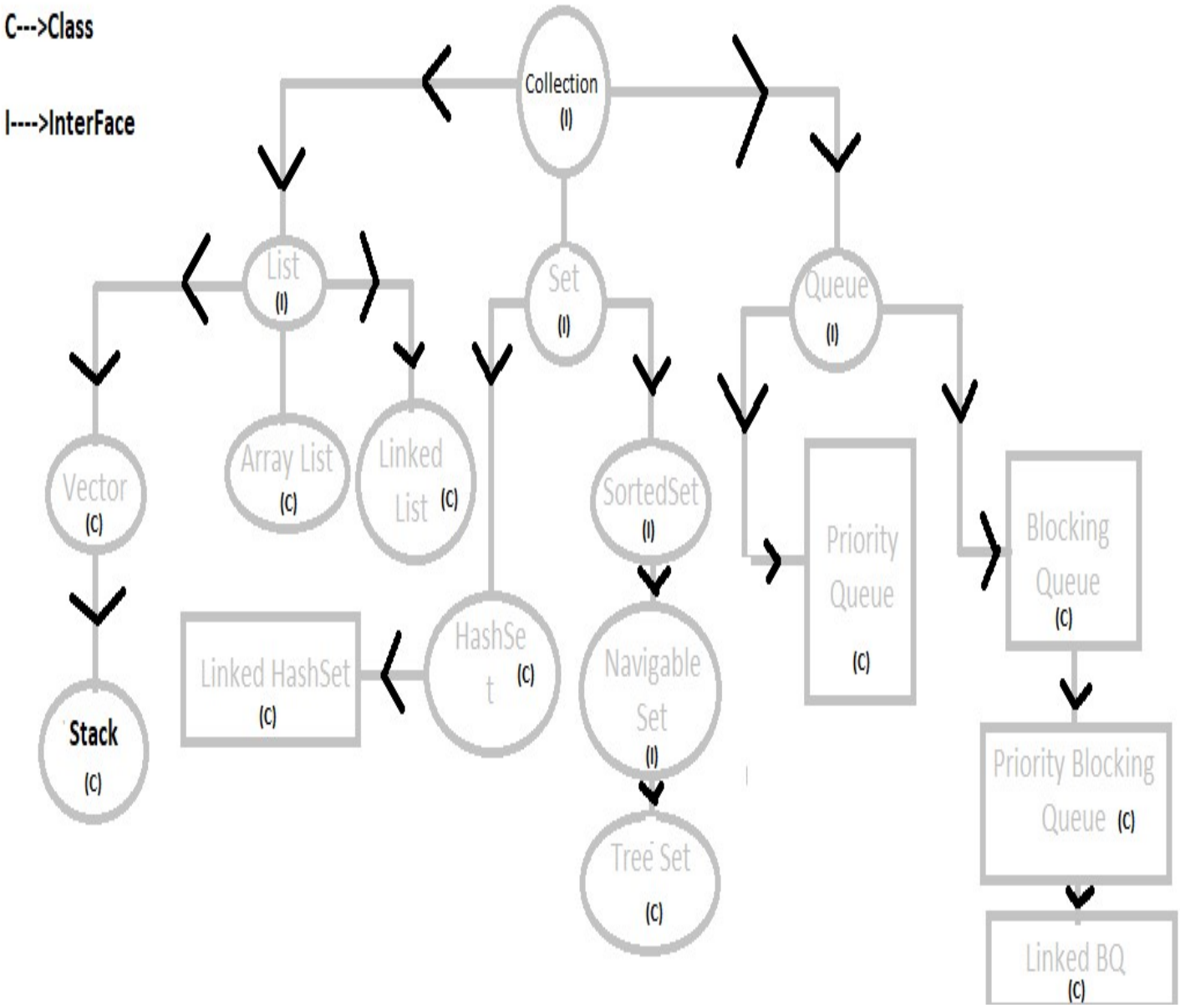
- It is the child interface of Map interface.
- If we want to represent a group of object as a key value pair and keys are inserted according to some sorting order then we will use sortedMap.

- Here sorting is done based on keys but not according to values.

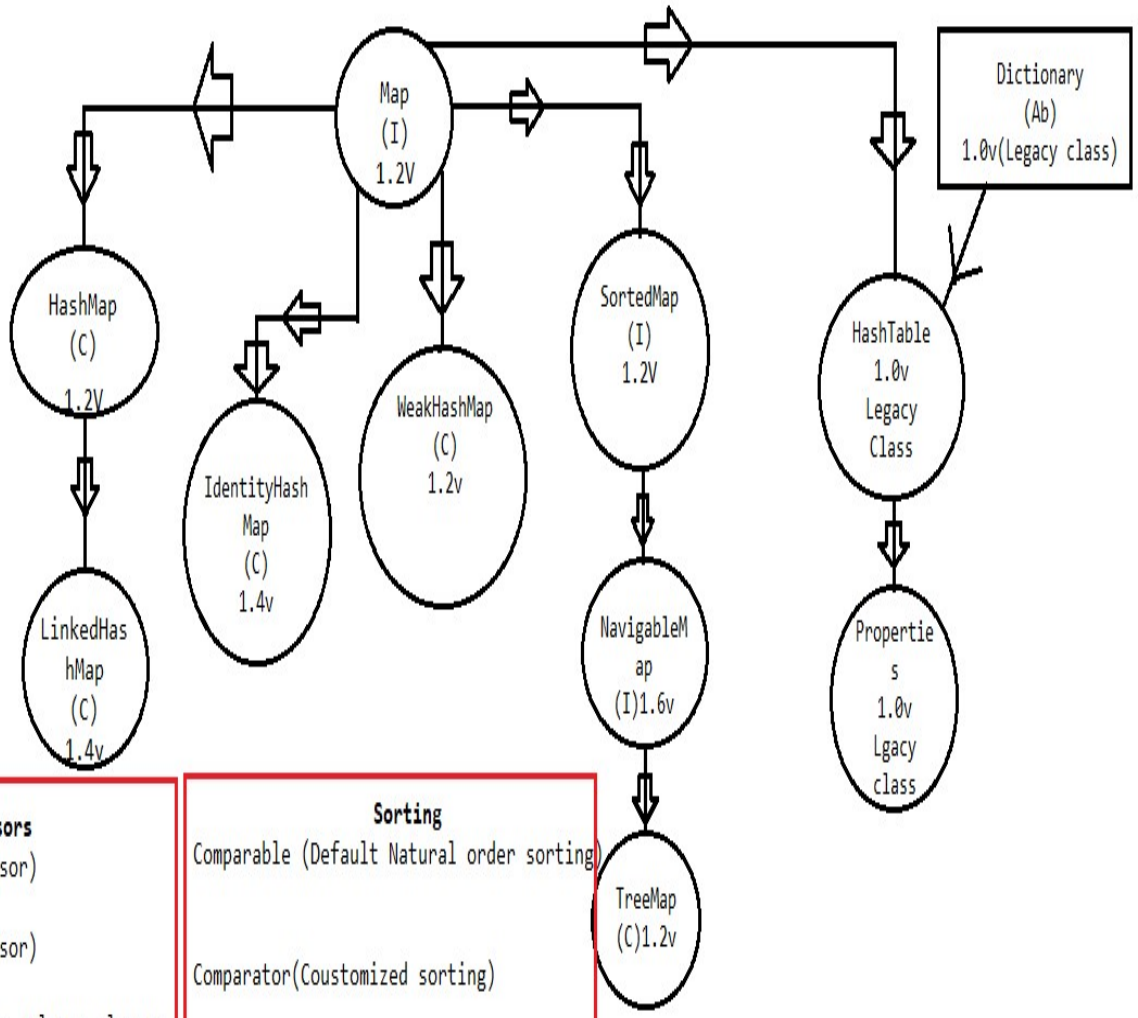
19.What is a Navigable Map?

- It is the child interface of Sorted Map.
- It defines several methods for navigation purposes.

20. Define Collection Hierarchy?



I--->Interface
C--->Class
Ab-->Abstract Class



Cursors

Enumeration(Legacy Cursor)

Iterator(Universal Cursor)

ListIterator(Applicable only on classes present under List interface)

Sorting

Comparable (Default Natural order sorting)

Comparator(Coustomized sorting)

Applicable on TreeSet and TreeMap

21. Differentiate Between ArrayList and Vector?

ArrayList	Vector
1. Every method present in ArrayList is non Synchronized. 2. ArrayList is not thread safe. 3. At a time multiple threads are allowed to operate on its object. 4. ArrayList is not a Legacy class because it is introduced in 1.2 version of java. 5. Performance wise ArrayList is faster because multiple threads are allowed to operate on its object.	1. Every method present in vector is not synchronized. 2. Vector is thread safe. 3. At a time only one thread is allowed to operate on its object. 4. Vector is also known as legacy class because it is introduced in 1.0 version of java. 5. Relatively performance is lower because at a time only only one thread is allowed to operate.

22. How to get Synchronized version of ArrayList?

- By default arrayList is nonSynchronized.
- We can get the synchronized version of ArrayList by using the **SynchronizedList () method of Collections class.**

e.g:

```
ArrayList <String> tusar=new ArrayList<String> ();

List<String> l1=Collections.synchronizedList (tusar);
```

Note:

- Similarly we can get Synchronized version of Map and Set objects by using the following methods of Collections class.

Public static Set SynchronizedSet (Set s);

Public static Map SynchronizedMap (Map m)

23. Differentiate between ArrayList and LinkedList?

ArrayList	LinkedList
1. It follows consecutive memory allocation. 2. Underlying datastructure is Growable Array. 3. Best choice for retrival operation. 4. Worst choice to do insertion and deletion operation in middle because internally several shift operatios are performed. 5. Implements Random Access interface.	1. Don't follow consecutive memory allocation. 2. Underlying datastructure is DoublyLinkedList. 3. Best choice to perform insertion and deletion operation in the middle. 4. Worst choice for retrival operation. 5. It don't implements the Random Access interface.

24. Why we use cursors in java collection?How many cursors are present in collection?

- If we want to iterate over a collection and to get the objects one by one from the collection then we will go for cursors.
- There are 3 types of cursors are present in collection.
  - **i.** Enumeration(Also known as Legacy cursor)
  - **ii.** Iterator
  - **iii.** ListIterator

25.Difference between Iterator and ListIsterato?

Iterator	ListIterator
➢ It is a universal cursor which can be used for all the collection objects. ➢ It is single directional cursor, it can move only in forward direction. ➢ It can only perform retrival and remove operation.	➢ It is applicable only for List interface implementation classes. ➢ It is a bidirectional cursor means it can move both forward and backword direction. ➢ It can perform retrive,remove,replace and insertion operation in the middle.

26.Differentiate between Enumeration and Iterator?

Enumeration	Iterator
➢ It is only applicable for vector class and stack class. ➢ It is also known as Legacy cursor because it is introduced in java version 1.0. ➢ We can perform only read operation with the help of enumeration.	➢ It is applicable across all the collection object. ➢ It is also known as universal cursor introduced in version 1.2. ➢ We can perform read and remove operation with the help of iterator.

27.Difference between ListIterator and Enumeration?

ListIterator	Enumeration
➢ ListItrerator is applicable only for List interface and its implementation classes. ➢ It is not a legacy iterator. ➢ It is a bidirectional iterator because it can move in both forward and backword direction. ➢ With the help of ListIterator we can do retrieve , replace , add and remove operation.	➢ It is applicable only for vector and Stack. ➢ It is also known as Legacy iterator. ➢ It is a single directional cursor, it can move only in forward direction. ➢ With help of enumeration we can only retrieve the objects.

28.Differentiate between HashSet and LinkedHashSet?

HashSet	LinkedHashSet
➢ Underlying datastructure is hashtable. ➢ Duplicate objects are not allowed and insertion order also not preserved in Hashset. ➢ Introduced in 1.2 version.	➢ Underlying datastructure is combination of Linkedlist and hashtable. ➢ Duplicate objects are not allowed but insertion order preserved. ➢ Introduced in 1.4 version of java.

29.Difference between comparable and comparator?

Comparable	Comparator
➢ It is present in java.lang package. ➢ If our need is default natural sorting order then we should go for comparable interface. ➢ It contains only 1 method that is compareTo method. ➢ All the wrapper classes and String class implements comparable.	➢ It is present in java.util package. ➢ If we need our own customized sorting order then we should go for comparator. ➢ It contains two methods compare and equals. ➢ Collator and RuleBasedCollator implements comparator.

30.Write a program to insert integer objects to the treeset where the sorting order is descending order?

```
public class Comparator1 {

    public static void main(String[] args) {
TreeSet <Integer> t=new TreeSet<>(new Comparator2());
        t.add(10);
        t.add(0);
        t.add(5);
        t.add(1);
```

```
        System.out.println(t);// [10, 5, 1, 0]
    }


public class Comparator2 implements Comparator<Object> {

    @Override
    public int compare(Object o1, Object o2) {
        Integer I1=(Integer)o1;
        Integer I2=(Integer)o2;
        if(I1>I2) {
            return -1;
            }else if(I1<I2) {
                return +1;
            }else {
                return 0;
            }
```

31.Write a program to insert String objects to the tree set where all elements should inserted according to reverse of alphabetical order?

```
    public static void main(String[] args) {
        TreeSet <String> t=new TreeSet<>(new C4());
        t.add("Ravi");
        t.add("Tusar");
        t.add("Rocky");
        t.add("Rocky");
        t.add("Nil");
        t.add("Abd");
        System.out.println(t);
    }
public class C4 implements Comparator<Object>{

    @Override
    public int compare(Object o1, Object o2) {
        String I1=o1.toString();
        String I2=o2.toString();
        return I2.compareTo(I1);}
```

32. **Write a program to insert StringBuffer objects into treeset where the sorting order is alphabetical order?**

```
public class C5 implements Comparator<Object> {

	public static void main(String[] args) {
		TreeSet <StringBuffer> t=new TreeSet<>(new C5());
		t.add(new StringBuffer("A"));
		t.add(new StringBuffer("Z"));
		t.add(new StringBuffer("C"));
		System.out.println(t);
	}

	@Override
	public int compare(Object o1, Object o2) {
		String s1=o1.toString();
		String s2=o2.toString();
		return s1.compareTo(s2);
	}

}
```

33. **Differentiate between HashSet, LinkedHashSet and TreeSet?**

- HashSet is the child class of Set interface, LinkedHashSet is the child class of HashSet and TreeSet is the child class of NavigableSet.

Properties	HashSet	LinkedHashSet	TreeSet
1. Duplicate objects.	**NA**	**NA**	**NA**
2. Insertion order.	**Not Preserved**	**Preserved**	**Not Preserved**
3. Sorting Order	**Not Followed**	**Not Followed**	**Follow sorting order (Default sorting and customized sorting)**
4. Underlying Data Structure	**HashTable**	**LinkedList and HashTable**	**Balanced Tree**
5. Heterogeneous Objects	**Allowed**	**Allowed**	**Not Allowed**
6. Null insertion	**Allowed**	**Allowed**	**Not Allowed from 1.7 version of java(Before 1.7 version it is allowed but as the first element in a empty treeset)**