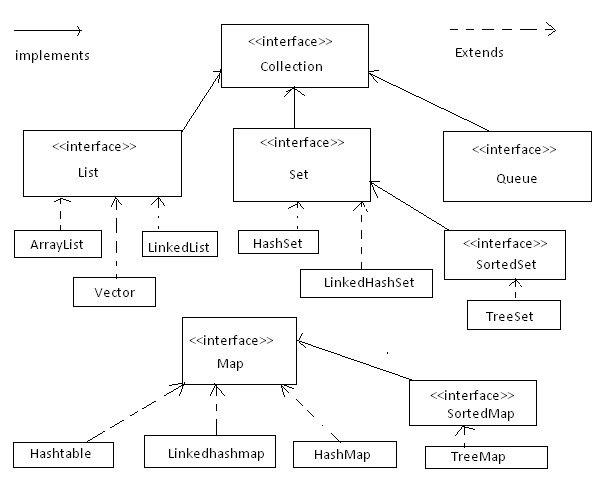
Collection Framework:



1. What are the limitations of object Arrays?

The main limitations of Object arrays are

* These are fixed in size i.e. once we created an array object there is no chance of increasing or decreasing size based on our requirement. Hence If we don’t know size in advance , arrays are not recommended to use
* Arrays can hold only homogeneous elements.
* There is no underlying data structure for arrays and hence no readymade method support for arrays. Hence for every requirement programmer has to code explicitly

To overcome these problems collections are recommended to use

2. What are differences between arrays and collections?

|  |  |
| --- | --- |
| **Arrays** | **Collections** |
| 1. Arrays r fixed in size and hence once we created an array we are not allowed to increase or decrease the size based on our requirement. | 1. Collections are grow able in nature and hence based on our requirement we can increase or decrease the size. |
| 2. Memory point of view arrays are not recommended to use | 2. Memory point of view collections are recommended to use. |
| 3. Performance point of view arrays are recommended to use | 3. Performance point of view collections are not recommended to use. |
| 4. Arrays can hold only homogeneous elements | 4. Collections can hold both homogeneous and heterogeneous elements. |
| 5. Arrays can hold both primitives as well as objects | 5. Collections can hold only objects. |
| 6. For any requirement, there is no ready method support compulsory programmer has to code explicitly. | 6. For every requirement readymade method support is available. Being a programmer we have to know how to use those methods and we are not responsible to implement those. |

3. What are differences between arrays and ArrayList?

Refer the answer of 2 (differences between arrays and collections)

4. What are differences between arrays and Vector?

Refer the answer of 2 (differences between arrays and collections)

5. Explain Java Collections Framework or collection API?

Java Collections Framework provides a well designed set of interfaces and classes that support operations on a collection of objects.

6. What is difference between Collections and Collection?

**Collection** is an interface which can be used for representing a group of individual objects as single entity and it acts as root interface of collection frame work.

**A collection is** a utility class to define several utility methods for Collection implemented class objects.

7. Explain about Collection interface?

* This interface can be used to represent a group of objects as a single entity.
* It acts as root interface for entire collection framework.
* It defines the most commonly used methods which can be applicable for any collection implemented class object

8. Explain Iterator Interface.

An Iterator is similar to the Enumeration interface. With the Iterator interface methods, you can traverse a collection from start to end and safely remove elements from the underlying Collection. The Iterator () method generally used in query operations.

**Basic methods:**

* iterator.remove();
* iterator.hasNext();
* iterator.next();

9. What is difference between Iterator and ListIterator?

|  |  |
| --- | --- |
| **Iterator** | **ListIterator** |
| Iterator is an unidirectional | ListIterator is bidirectional and it is the child interface of the Iterator |
| While iterating the elements by Iterator we can perform only read and remove operations. | ListIterator we can perform read, removal, replace also |
| Iterator is applicable for every Collection implemented class objects | ListIterator is applicable only for List implemented class objects. |
| Iterator iterates only in forward direction using hasNext() and next() methods | ListIterator iterates in both directions using next(), hasNext(), previous() and hasPrevious() methods |

10. Explain Enumeration Interface.

The Enumeration interface allows you to iterate through all the elements of a collection. Iterating through an Enumeration is similar to iterating through an Iterator. However, there is no removal support with Enumeration.

**Basic methods:**

* boolean hasMoreElements();
* Object nextElement();

11. What is the difference between Enumeration and Iterator interface?

|  |  |
| --- | --- |
| **Enumeration** | **Iterator** |
| 1. It is legacy interface and introduced in 1.0 version | 1 It is non-legacy and introduced in 1.2 version |
| 2 Applicable only for legacy classes and it is not universal cursor | 2 Applicable for any Collection implemented class object. |
| 3 While iterating the elements we are not allowed to remove the objects just we can perform only read operation | 3 While iterating we can perform removal also in addition to read operation. |
| 4 By using elements() method we can get Enumeration object | 4 By using iterator() method we can get Iterator  object |

12. What is Comparable interface?

This interface can be used for defining natural sorting order of the objects.

* It is present in java.lang package
* It contains a method **public int compareTo(Object obj1)**

13. What is Comparator interface?

This interface can be used for implementing customized sorting order.

* + It is present in java.util package
  + It contains two methods

**public int compare(Object ,Object)**

**public boolean equals(Object)**

14. What are differences between Comparable and Comparator?

|  |  |
| --- | --- |
| Comparable | Comparator |
| 1 This can be used for natural sorting order | 1This can be used for implementing customized sorting |
| 2 This interface present in java.lang package | 2 This is present in java.util package |
| 3 Contains only one method:  public int compareTo(Object obj1) | 3 It contains two methods.  public int compare(Object ,Object)  public Boolean equals(Object) |
| 4 It is marker interface | 4 It is not a marker interface. |

15. What is Marker Interface?

* Marker interface is the one which doesn’t contain any methods.
* Marker interfaces are Serializable, Clonable, SingleThreadModel, Event listener. Marker Interfaces are implemented by the classes or their super classes in order to add some functionality.
* Suppose the interface Clonable is implemented by a class named Myclass, then a call to the method clone() on Myclass's object will give an error. This means, to add this functionality one should implement the Clonable interface. While the Clonable is an empty interface but it provides an important functionality.

16. What are legacy classes?

The classes which are synchronized (thread-safe) are called legacy classes.

17. What are legacy classes and interfaces present in Collections framework?

* Enumeration ---Interface
* Dictionary ------Abstract class
* Hashtable -----Concrete class
* Properties -----Concrete class
* Vector -----Concrete class
* Stack -----Concrete class

18. Explain Set Interface?

Set is a child interface of Collection interface. it can be used to represent a group of individual objects as a single entity where

* Duplicate objects are not allowed.
* Insertion order is not preserved
* In Set implementations null is valid entry, but allowed only once.

19. What are the two types of Set implementations available in the Collections Framework?

HashSet and TreeSet are the two Set implementations available in the Collections Framework.

20. Explain about HashSet class?

* The underlying data structure is Hashtable
* Null values are accepted
* Duplicates are not allowed
* Insertion order is based on hashcode of the object hence insertion order is not preserved
* Best suitable if frequent operation is search operations
* HashSet class implements Serializable and Cloneable
* it is implementation class for Set interface
* Heterogeneous objects are allowed
* it is introduced in 1.2 version

21. If we are trying to insert duplicate values in Set what will happen?

If we are trying to insert duplicate objects to the HashSet, we won’t get any compile time or run time errors just the add(Object o) returns false and it doesn’t add that object.

22. What is LinkedHashSet?

It is the child class of HashSet. The main difference between HashSet and LinkedHashSet is:

In the case of **HashSet insertion order is not preserved**, but in the case **of LinkedHashSet insertion order will be preserved**.

23. What is the difference between HashSet and LinkedHashSet?

|  |  |
| --- | --- |
| HashSet | LinkedHashSet |
| 1 The Underlying data structure is Hashtable | 1 The underlying data structure is combination of LinkedList and Hashtable |
| 2 Insertion Order is not preserved | 2 Insertion order is preserved. |
| 3 Introduced in 1.2 version | 3 Introduced in 1.4 version |

24. Explain about SortedSet interface?

It is child interface of Set interface. It can be used to represent a group of individual objects in to a single entity where

* All the objects are arranged in some sorting order (Can be natural sorting order or customizede).
* Duplicates are not allowed.

25. Explain about NavigableSet?

It is child interface of SortedSet and provides several utility methods for navigation purposes

* + It doesn’t allows duplicates
  + Insertion order is preserved
  + It is introduced in 1.6 version

26. Explain about TreeSet?

It is Collection object which can be used to represent a group of objects according to some sorting order.

* The underlying data structure is Balanced tree
* Duplicates are not allowed
* All objects are stored according to some sorting order hence insertion order is not preserved
* Heterogeneous objects are not allowed violation leads to ClassCastException
* For an Empty TreeSet as firs element null value can be inserted but after inserting that first value if we are trying to insert any other objects then we will get NullPointerException
* For an non empty TreeSet if we are trying to insert null value at run time you will get NullPointerException

27. What is difference between HashSet and TreeSet?

|  |  |
| --- | --- |
| **HashSet** | **TreeSet** |
| 1 The underlying data structure is Hashtable | 1The underlying data structure is balanced tree |
| 2 Heterogeneous objects are allowed | 2 Heterogeneous objects are not allowed by default |
| 3 Insertion order is not preserved and it is based on hashcode of the objects | 3 Insertion order is not preserved and all the objects are inserted according to some sorting order. |
| 4 Null insertion is possible | 4 As the first element only null insertion is possible and in all other cases we will get NullPointerException |

28. What is a List?

List interface is a child interface of Collection interface. This can be used to represent group of individual objects in as a single entity where

* Duplicates are allowed
* Insertion order is preserved

29. What are the two types of List implementations available in the Collections Framework?

ArrayList and LinkedList are the two List implementations available in the Collections Framework.

30. Explain about ArrayList class?

ArrayList is a Collection which can be used to represent a group of objects as a single entity.

* it is a implemented class for List interface
* Introduced in 1.2 version
* The underlying data structure is resizable or grow able array.
* Insertion order is preserved
* Duplicates are allowed
* Heterogeneous objects are allowed
* Null insertion is possible
* This class implements RandomAccess , Serializable , Cloneable interfaces
* Best choice for retrieval purpose
* Worst if our frequent operation is insertion or deletion in the middle
* All methods present in ArrayList class are not synchronized (Single-Threaded) hence ArrayList class object are not thread safe.

31. Explain about LinkedList class?

LinkedList is a Collection implemented class which can be used for representing a group of objects as a single entity.

* LinkedList is the implementation class for List interface
* Introduced in 1.2 version
* Underlying data Structure is DoubleLinkedList
* Allows duplicates
* Insertion order is preserved
* Allows heterogeneous objects
* null insertion is possible
* LinkedList class implements Serializable and Cloneable interface but not RandomAccess interface
* Best choice if frequent operation is insertion or deletion an objects in middle
* Worst choice if frequent operation is retrieval.

32. Explain about Vector class?

Vector is a legacy collection class which can be used to represent a group of objects.

* Introduced in 1.0 version. it is legacy class
* The underlying data structure is resizable or grow able array.
* Insertion order is preserved
* Duplicates are allowed
* Heterogeneous objects are allowed
* It is a implemented class for List interface
* Null insertion is possible
* Vector class implements RandomAccess ,Serializable,Cloneable interfaces
* Best Choice if frequent operation is retrieval and worst choice if frequent operation is insertion or deletion in the middle.
* All methods present in Vector class are synchronized (multi-Threaded) hence Vector class object is thread safe.

33. What is the difference between ArrayList and LinkedList?

|  |  |
| --- | --- |
| **ArrayList** | **LinkedList** |
| 1. The underlying data structure is resizable or grow able array. | 1. The underlying data structure is Double Linked List. |
| 2. This is Best choice if frequent operation is retrieval and worst choice if frequent operation is insertion or deletion in the middle. | 2. This is Best choice if frequent operation is insertion or deletion in the middle and worst choice if frequent operation is retrieval. |
| 3. This class implements Serializable, Cloneable and RandomAccess interfaces. | 3. This class implements Serializable, Cloneable but not RandomAccess interface. |

34. What is difference between ArrayList and Vector?

|  |  |
| --- | --- |
| ArrayList | Vector |
| 1. No method is synchronized in the ArrayList class | 1. All methods in Vector are synchronized. |
| 2. ArrayList object is not thread safe. | 2. Vector is thread safe. |
| 3. Relatively performance is high | 3. Relatively performance is low |
| 4. Introduced in 1.2 version and it is non legacy | 4. Introduced in 1.0 version and it is legacy |

35. What are differences between List and Set interfaces?

|  |  |
| --- | --- |
| List | Set |
| A list represents the ordered collection of elements | A set represents a collection of elements |
| List preserves the order of objects in which they are entered. | Insertion Order is not preserved |
| Duplicate Objects are allowed | Duplicate Objects are not allowed |
| The implemented classes are ArrayList, LinkedList , Vector and Stack classes | The implemented classes are HashSet, LinkedHashSet and Tree |
| List is index based | Set in not an index based |
| List will allow null elements to be stored | Set will not allow null elements |

36. How we can get synchronized version of ArrayList?

Collections class contains synchronizedList() method for this

* Public static List synchronizedList(List l)

EX

ArrayList l= new ArrayList();

List l2=Collections.synchronizedList(l);

Similarly we can get synchronized versions of Set and Map objects by the following methods.

* Public static List synchronizedSet(Set s)
* Public static List synchronizedMap(Map m)

37. We are planning to do an indexed search in a list of objects. Which of the two Java collections should you use: ArrayList or LinkedList?

ArrayList

38. Why ArrayList is faster than Vector?

All methods present in the Vector are synchronized and hence any method can be executed by only one thread at a time. It slows down the execution.

But in ArrayList, no method is synchronized and hence multiple threads are allowed execute simultaneously which speed up the execution.

39. Explain the functionality of Vector Class?

Once array size is set you cannot change size of the array. To deal with this kind of situations in Java uses Vector, it grows and shrinks its size automatically. Vector allows storing only objects not primitives. To store primitives, convert primitives into objects using wrapper classes before adding them into Vector. The Vector reallocates and resizes itself automatically.

40. What does the following statement convey?

Vector vt = new Vector(3, 10);

vt is an instance of Vector class with an initial capacity of 3 and grows in increment of 10 in each relocation

41. How do you store a primitive data type within a Vector or other collections class?

You need to wrap the primitive data type into one of the wrapper classes found in the java.lang package, like Integer, Float, or Double, as in:

Integer in = new Integer(5);

42. What is RandomAccess Interface?

* If a collection class implements RandomAccess interface then we can access any of its element with the same speed.
* RandomAccess interface is marker interface and it dosent contains any methods.
* An ArrayList and vector class implements this interface.

43. What collection will you use to implement a queue?

LinkedList

44. Explain about Queue interface?

If we want to represent a group of individual objects prior to processing, then we should go for Queue interface. It is child interface of Collection interface.

It has introduced in 1.5 version.

45. Explain Map Interface.

Remember it is not a child Interface of Collection Interface and hence Map and Collection Interfaces doesn’t have any relationship.

* It can be used for representing a group of Objects as key, value pairs.
* Both keys and values should be objects
* Keys can t be duplicated but values can be duplicated.
* it has introduced in 1.2 version

46. Explain about SortedMap?

If we want to represent a group of objects as key value pairs where all the entries are arranged according some sorting order of keys then we should go for SortedMap.

* It is child interface of Map.
* It has introduced in 1.2 version

47. What are the two types of Map implementations available in the Collections Framework?

HashMap and TreeMap are two types of Map implementations available in the Collections Framework.

48. What is Entry interface?

It is inner interface of Map.

In the Map each key value pair is considered as Entry object.

|  |
| --- |
| interface Map{  //more code here  interface Entry{  Object getKey()  Object getValue()  Object setValue(Object new)  }  } |

49. What is Hashtable?

Hashtable is a legacy Map and can be used to store objects as key value pairs.

* The underlying data sturucture is Hashtable
* Duplicates keys are not allowed but duplicate values are allowed
* null insertion is not possible for both keys and values
* all methods are synchronized
* insertion order is not preserved because it is based on hashcode of keys
* heterogeneous Objects are allowed for both keys and values
* introduced in 1.0 version it is legacy class

50. Explain about HashMap?

It is a Map Object which can be used to represent a group of objects as key-value pairs.

* + The underlying data structure is Hashtable
  + Duplicate keys are not allowed duplicate values are allowed
  + Insertion order is not preserved because insertion is based on hashcode of keys.
  + Heterogeneous objects are allowed for both keys and values
  + Null key is allowed only once
  + Null values are allowed multiple times
  + Introduced in 1.2 version

51. Explain about LinkedHashMap?

* It is child class of HashMap. It is exactly same as HashMap except the following difference.
* In the case of HashMap the insertion order is not preserved but in the case of LinkedHashMap insertion order is preserved. Introduced in 1.4 version
* It is child class of HashMap. It is exactly same as HashMap except the following difference.

52. Differences between HashMap and LinkedHashMap?

|  |  |
| --- | --- |
| **HashMap** | **LinkedHashMap** |
| 1.The underlying data structure is Hashtable | 1.The underlying data structure is a combination of Hashtable and LinkedList |
| 2.Insertion order is not preserved and it is based on hashcode of keys | 2 Insertion order is preserved |
| 3.Introduced in 1.2 version | 3 Introduced in 1.4 version |

53. Differences between HashMap and Hashtable?

|  |  |
| --- | --- |
| **HashMap** | **Hashtable** |
| The underlying data structure is Hashtable | The underlying data structure of Hashtable |
| No method is synchronized and hence HashMap object is not thread safe | All methods are synchronized and hence it is thread safe |
| Performance is high | Performance is low |
| Null insertion is possible for both keys and values | Null insertion is not possible for both key and value violation leads to NullPointerException |
| Introduced in 1.2 version and it is non legacy | Introduced in 1.0 version and it is legacy |

54. What is TreeMap?

TreeMap can be used to store a group of objects as key-value pairs where all the entries are arranged according to some sorting order of keys.

* The underlying data structure is RED-BLACK Tree
* Duplicates keys are not allowed but values can be duplicated.
* Insertion order is not preserved because insertion is based on some sorting order
* If we are depending on Natural sorting order then keys should be homogeneous(violation leads to ClassCastException) but values need not homogeneous
* In case of customized sorting order we can insert heterogeneous keys and values
* For empty TreeMap as first entry with null values are allowed but after inserting that entry if we are trying to insert any other entry we will get NullPointerException
* For non empty TreeMap if we are trying to insert null keys we will get NullPointerException
* There are no restrictions for null values.

55. What is the difference between HashMap and TreeMap?

|  |  |
| --- | --- |
| HashMap | TreeMap |
| The underlying data structure is Hashtable | The underlying data structure is RED-BLACK Tree |
|  |  |

56. How do I make an array larger?

You cannot directly make an array larger. You must make a new (larger) array and copy the original elements into it, usually with **System.arraycopy ().** If you find yourself frequently doing this, the Vector class does this automatically for you, as long as your arrays are not of primitive data types.

57. Which is faster, synchronizing a HashMap or using a Hashtable for thread-safe access?

Because a synchronized HashMap requires an extra method call, a Hashtable is faster for synchronized access.

58. In which package would you find the interfaces and classes defined in the Java Collection Framework?

java.util

59. What method in the System class allows you to copy elements from one array to another?

System. arraycopy()

60. When using the System.arraycopy() method, What exception is thrown if the destination array is smaller than the source array?

ArrayIndexOutofBoundsException

61. What is the use of Locale class?

The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region

62. What is the use of GregorianCalendar class?

The GregorianCalendar provides support for traditional Western calendars

63. What is the use of SimpleTimeZone class?

The SimpleTimeZone class provides support for a Gregorian calendar

64. What is the use of ResourceBundle class?

The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.