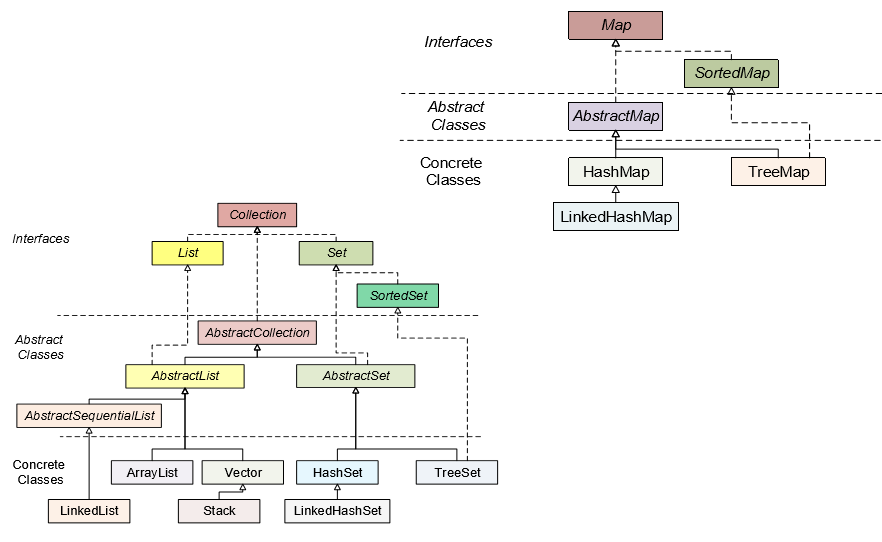
1. Draw Collections Framework Class Diagram



2. What is HashMap and Map?

Map is an interface and Hashmap is a particular implementation of that interface

3. Difference between HashMap and HashTable? Can we make hashmap synchronized?

Differences:

1. HashMap is not synchronized where as Hashtable is synchronized.
2. HashMap can contain one null key and multiple null values where as Hashtable cannot contain any null key or null value

Can we make hashmap synchronized?

Yes. In order to synchronize it we need to use [Collections.synchronizedMap(hashmap)](http://docs.oracle.com/javase/7/docs/api/java/util/Collections.html#synchronizedMap(java.util.Map)).

4. Difference between Vector and ArrayList?

Vector Arraylist

1. Vector is synchronized. ArrayList is not synchronized
2. Vector is a legacy class ArrayList is not a legacy class.
3. Vector increases its size by doubling ArrayList increases its size by 50% of the

the array size array size

5. What is an Iterator?

Iterator traverses the elements in forward direction and it can be used in List, Set and Queue.

6. List vs Set vs Map. Purposes and definitions.

The Set interface provides an unordered collection of unique objects, i.e. Set doesn't allow duplicates, while Map provides a data structure based on key-value pair and hashing. 

List allows duplicates while Set doesn't allow duplicates and Map may contain duplicate values but keys are always unique  
  
The list allows null elements and you can have many null objects in a List because it also allowed duplicates. Set just allow one null element as there is no duplicate permitted while in Map you can have null values and at most one null key

7. Pros and cons of ArrayList and LinkedList

Pros of ArrayList

1. ArrayList uses internally an array for internal storage. That makes it particularly fast for random access.

Cons of ArrayList

1. ArrayList is slower for modification operations like add or delete elements in the beginning or middle of the collection. This is due to the need of relocate all subsequent elements one position to the right (or left in case of deletion) in order to make space to the new element.

2. Similar to before described process. ArrayList has some performance downside when the internal array is completely full, and therefore has to create a bigger array and relocate all elements to new array.

Pros of LinkedList

1. LinkedList follows a different approach. It's more efficient in adding or deleting elements in the beggining or middle of the collection. If you ever programmed from scratch a list data structure, you will remember you have nodes with pointers/references to the next element.

2. Given the nature of the internal structure which is not restricted to an initial size, LinkedList has no growing problems as ArrayList.

Cons of LinkedList

1. Random access to LinkedList elements are expensive, because in worst case scenarios the entire list has to be traversed to retrieve the desired element

8. TreeSet vs LinkedHashSet

1. LinkedHashSet maintains insertion order of elements. i.e elements are placed as they are inserted. And TreeSet orders the elements according to natural order.

2. LinkedHashSet also allows maximum one null element and TreeSet doesn’t allow even a single null element

9. What are relationships between equals and hash codes?

1) If two objects are equal, then they must have the same hash code.  
2) If two objects have the same hash code, they may or may not be equal.

10. What are the advantages of ArrayList over arrays ?

1.Arrays are of fixed length. You can not change the size of the arrays once they are created.

2.You can not accommodate an extra element in an array after they are created.

3.Memory is allocated to an array during it’s creation only, much before the actual elements are added to it.

4. Arraylist is dynamic.

5. Arraylist contains more functionalities than arrays.

11. Principle of storing data in a hashtable

Hashing is implemented in two steps:

1. An element is converted into an integer by using a hash function. This element can be used as an index to store the original element, which falls into the hash table.
2. The element is stored in the hash table where it can be quickly retrieved using hashed key.

hash = hashfunc(key)  
 index = hash % array\_size

In this method, the hash is independent of the array size and it is then reduced to an index (a number between 0 and array\_size − 1) by using the modulo operator (%).

12. Differences between Hashtable, ConcurrentHashMap and Collections.synchronizedMap()

**1. HashTable**

Hashtable is an implementation of Map data structure

This is a legacy class in which all methods are synchronized on Hashtable instances using synchronized keyword.

Thread-safe as it’s method are synchronized

**2. SynchronizedMap**

This is wrapper of Map, and can be initialize by using Collections.synchcronizedMap(new HashMap())

similar to Hashtable

The only difference between Hashtable and Synchronized Map is that later is not a legacy and you can wrap any Map to create it’s synchronized version by using Collections.synchronizedMap() method.

**3. ConcurrentHashMap**

ConcurrentHashMap implements Map data structure and also provide thread safety like Hashtable.

It works by dividing complete hashtable array in to segments or portions and allowing parallel access to those segments.

The locking is at a much finer granularity at a hashmap bucket level.

Use ConcurrentHashMap when you need very high concurrency in your application.

It is thread-safe without synchronizing the whole map.

Reads can happen very fast while write is done with a lock on segment level or bucket level.

There is no locking at the object level.

ConcurrentHashMap doesn’t throw a ConcurrentModificationException if one thread tries to modify it while another is iterating over it.

ConcurrentHashMap does not allow NULL values, so key can not be null in ConcurrentHashMap

ConcurrentHashMap doesn’t throw a ConcurrentModificationException if one thread tries to modify it, while another is iterating over it.

13. How are hash codes computed?

A hashcode is an integer value that represents the state of the object upon which it was called. That is why an Integer that is set to 1 will return a hashcode of "1" because an Integer's hashcode and its value are the same thing. A character's hashcode is equal to it's ASCII character code. If you write a custom type you are responsible for creating a good hashCode implementation that will best represent the state of the current instance.

14. Is it possible that hash code is not unique?

Yes.Its possible

15. Can we put two elements with equal hash code to one hash map?

Yes. We will call it as collision

16. Iterator and modification of a List. ConcurentModificationException.

java.util.ConcurrentModificationException is a very common exception when working with [java collection](https://www.journaldev.com/1260/collections-in-java-tutorial)classes. Java Collection classes are fail-fast, which means if the Collection will be changed while some thread is traversing over it using iterator, the *iterator.next()* will throw **ConcurrentModificationException**. Concurrent modification exception can come in case of multithreaded as well as single threaded java programming environment.

17. What is the significance of ListIterator? What is the difference b/w Iterator and ListIterator?

ListIterator traverses the elements in forward and backward direction.

Iterator traverses the elements in forward direction only where as ListIterator traverses the elements in backward and forward directions both.

Iterator can be used in List, Set and Queue where as ListIterator can be used in List only

18. What is the Collections API?

The Collection API is a set of classes and interfaces that support operation on collections of objects. These classes and interfaces are more flexible, more powerful, and more regular than the vectors, arrays, and hashtables if effectively replaces.

19. How can we access elements of a collection?

We can access elements of a collection using Iterator or ListIterator or Enumeration

20. What is the difference between a queue and a stack?

Stack – Represents the collection of elements in Last In First Out order.   
Operations includes testing null stack, finding the top element in the stack, removal of top most element and adding elements on the top of the stack.  
  
Queue - Represents the collection of elements in First In First Out order.  
Operations include testing null queue, finding the next element, removal of elements and inserting the elements from the queue.   
Insertion of elements is at the end of the queue  
Deletion of elements is from the beginning of the queue.

21. What is the Properties class?

Properties is a subclass of Hashtable. It is used to maintain lists of values in which the key is a String and the value is also a String. It can be used to get property value based on the property key. The Properties class provides methods to get data from properties file and store data into properties file. Moreover, it can be used to get properties of system.

22. Which implementation of the List interface provides for the fastest insertion of a new element into the middle of the list?

Linkedlist

23. How can we use hashset in collection interface?

Java HashSet class is used to create a collection that uses a hash table for storage. It inherits the AbstractSet class and implements Set interface. The important points about Java HashSet class are: HashSet stores the elements by using a mechanism called hashing.

25. Can you limit the initial capacity of vector in java?

Yes.we can limit the initial capacity   
  
we can construct an empty vector with specified initial capacity  
  
public vector(int initialcapacity)

26. What method should the key class of Hashmap override?

Hashcode and Equals methods.

27. What is the difference between Enumeration and Iterator?

1. Iterator can traverse legacy and non-legacy elements where as Enumeration can traverse only legacy elements

2. Iterator is fail-fast and Enumeration is not fail-fast

3. Iterator is slower than Enumeration

28. Collections class and Arrays class

The Arrays class of the java.util package contains several static methods that we can use to fill, sort, search, etc in arrays. This class is a member of the Java Collections Framework and is present in **java.util.arrays**.

Collections class in java is a useful utility class to work with [collections in java](https://www.journaldev.com/1260/collections-in-java-tutorial). The java.util.Collectionsclass directly extends the Object class and exclusively consists of the static methods that operate on Collections or return them.