Q1.What is Collection in Java?
Ans - A collection represents a group of objects, known as its elements. Some collections allow duplicate elements and others do not. Some are ordered and others unordered.

Q2. Differentiate between Collection and collections in the context of Java.
Ans -

| Collection | Collections |
|---|---|
| It is an interface. | It is a utility class. |
| It is used to represent a group of individual objects as a single unit. | It defines several utility methods that are used to operate on collection. |
| The Collection is an interface that contains a static method since java8. The Interface can also contain abstract and default methods. | It contains only static methods. |

Q3. What are the advantages of the Collection framework?
Ans - We need not to learn multiple ad hoc collection APIs.

1. It provides a standard interface for collections that fosters software reuse and also provides algorithms to manipulate them.

2. Reduces the effort required to design and implement APIs by eliminating the need to produce ad hoc collections APIs.
3. It provides useful data structures and algorithms that reduces programming effort due to which we need not to write them ourselves.
4. It provides high-performance implementations of useful data structures and algorithms that increases the performance.

Q4.Explain the various interfaces used in the Collection framework.
Ans -j ava Collection framework provides many interfaces (Set, List, Queue, Deque) and classes (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet).

Q5.Differentiate between List and Set in Java.
Ans -

| List | Set |
|---|---|
| 1. The List is an indexed sequence. | 1. The Set is an non-indexed sequence. |
| 2. List allows duplicate elements | 2. Set doesn't allow duplicate elements. |
| 3. Elements by their position can be accessed. | 3. Position access to elements is not allowed. |

Q6.What is the Differentiate between Iterator and ListIterator in Java.
Ans -

| Iterator | ListIterator |
|---|---|
| Can traverse elements present in Collection only in the forward direction. | Can traverse elements present in Collection both in forward and backward directions. |

| | |
|---|---|
| Helps to traverse Map, List and Set. | Can only traverse List and not the other two. |
| Indexes cannot be obtained by using Iterator. | It has methods like nextIndex() and previousIndex() to obtain indexes of elements at any time while traversing List. |
| Cannot modify or replace elements present in Collection | We can modify or replace elements with the help of set(E e) |
| Cannot add elements and it throws ConcurrentModificationException. | Can easily add elements to a collection at any time. |

Q7.What is the Differentiate between Comparable and Comparator
Ans -

| **Comparable** | **Comparator** |
|---|---|
| 1) Comparable provides a **single sorting sequence**. In other words, we can sort | The Comparator provides **multiple sorting sequences**. In other words, we can sort |

| | |
|---|---|
| the collection on the basis of a single element such as id, name, and price. | the collection on the basis of multiple elements such as id, name, and price etc. |
| 2) Comparable **affects the original class**, i.e., the actual class is modified. | Comparator **doesn't affect the original class**, i.e., the actual class is not modified. |
| 3) Comparable provides **compareTo() method** to sort elements. | Comparator provides **compare() method** to sort elements. |
| 4) Comparable is present in **java.lang** package. | A Comparator is present in the **java.util** package. |

Q8.What is collision in HashMap?
Ans - A collision, or more specifically, a hash code collision in a HashMap, is a situation where two or more key objects produce the same final hash value and hence point to the same bucket location or array index

Q9.Distinguish between a hashmap and a Treemap.
Ans -

| Basis | HashMap | TreeMap |
|---|---|---|
| **Definition** | Java **HashMap** is a hashtable based implementation of Map interface. | Java **TreeMap** is a Tree structure-based implementation of Map interface. |
| **Interface Implements** | HashMap implements **Map, Cloneable**, and **Serializable** interface. | TreeMap implements **NavigableMap, Cloneable**, and **Serializable** interface. |

| | | |
|---|---|---|
| **Null Keys/ Values** | HashMap allows a **single** null key and **multiple** null values. | TreeMap does not allow **null** keys but can have **multiple** null values. |
| **Homogeneous/ Heterogeneous** | HashMap allows heterogeneous elements because it does not perform sorting on keys. | TreeMap allows homogeneous values as a key because of sorting. |

Q10.Define LinkedHashMap in Java?

Ans - A LinkedHashMap contains values based on the key. It implements the Map interface and extends the HashMap class. It contains only unique elements.