

1. What is the Collection framework in Java?

Ans:

Collection Framework is a combination of classes and interface, which is used to store data in the form of objects. It provides various classes such as ArrayList, Vector, Stack, and Has interfaces such as List, Queue, Set, etc. for this purpose.

2. What is the difference between ArrayList and LinkedList?

Ans

ArrayList	LinkedList
ArrayList uses a dynamic array.	. LinkedList uses a doubly linked list.
Linked List is efficient for manipulation.	.ArrayList is not efficient for manipulation because too much is required
ArrayList is better to store and fetch data .	LinkedList is better to manipulate data
ArrayList provides random access	LinkedList does not provide random access.
ArrayList takes less memory overhead as it stores only objects.	LinkedList takes more memory overhead, as it stores the object as well as the address of that object.

3. What is the difference between Iterator and ListIterator?

Ans:

Iterator	ListIterator
The Iterator traverses the elements in the forward direction only.	Listiterator traverses the elements in backward and forward directions both.
The Iterator can be used in List, Set, and Queue	ListIterator can be used in List only.
The Iterator can only perform a remove operation while traversing the collection	ListIterator can perform ?add,? ?remove,? and ?set? operation while traversing the collection

4. What is the difference between Iterator and Enumeration?

Ans:

Iterator	Enumeration
Iterator can travers legacy & non-legacy elements.	Enumeration can traverse only legacy elements.
The iterator is fail-fast.	Enumeration is not fail-fast.
The iterator is slower than Enumeration.	Enumeration is faster than Iterator.
The Iterator can perform a remove operation while traversing the collection.	The Enumeration can perform only traverse operations on the collection.

5. What is the difference between List and Set?

Ans:

The List and Set both extend the collection interface. However, there are some differences between the two which are listed below.

The List can contain duplicate elements whereas Set includes unique items.

- The List is an ordered collection which maintains the insertion order whereas Set is an unordered collection which does not preserve the insertion order.
- The List interface contains a single legacy class which is Vector class whereas the Set interface does not have any legacy class.
- The List interface can allow a number of null values whereas Set interface only allows a single null value.

6. What is the difference between HashSet and TreeSet?

Ans:

Both HashSet and TreeSet are implementations of the Set interface in Java, but they have some differences in terms of their properties and usage:

Ordering: HashSet is an unordered collection of elements, while TreeSet is a sorted set of elements based on their natural order or a custom comparator.

- Duplication: HashSet does not allow duplicate elements, while TreeSet does not allow duplicates as well.
- Implementation: HashSet is implemented using a hash table, while TreeSet is implemented using a self- balancing binary search tree (Red-Black tree).

- Performance: HashSet has constant-time complexity $O(1)$ for adding, removing, and testing the existence of an element, while TreeSet has a logarithmic-time complexity $O(\log n)$ for these operations due to the self-balancing property.

Memory usage: HashSet uses less memory than TreeSet because it only stores the elements, while TreeSet stores additional information for maintaining the order.

- Iteration: HashSet provides no guarantees regarding the order of iteration, while TreeSet guarantees the elements are iterated in sorted order.
- Usage: HashSet is suitable when ordering is not important, and fast access and membership tests are needed. TreeSet is suitable when elements need to be sorted or accessed in a specific order.

7. What is the difference between Array and ArrayList?

Ans:

Both arrays and ArrayLists are used to store collections of elements in Java, but they have some differences in terms of their properties and usage:

Type: Arrays can store elements of primitive data types as well as objects, while ArrayList can only store objects.

- Size: The size of an array is fixed once it is created, while the size of an ArrayList can be dynamically increased or decreased by adding or removing elements.

Mutability: Arrays are mutable, meaning that you can modify the elements in an array after it has been created. ArrayList is also mutable, but the only way to modify it is by adding, removing or modifying elements.

Performance: Arrays have better performance than ArrayLists for certain operations, such as accessing elements by index, because they are implemented as a continuous block of memory. ArrayLists, on the other hand, use dynamic memory allocation and are implemented as a dynamic array, which may result in more memory overhead and slower performance for certain operations.

- Methods: Arrays have a limited set of methods compared to ArrayLists, which provides more methods for manipulating the collection, such as adding, removing, and sorting elements.

Initialization: Arrays can be initialized with values at the time of creation, while ArrayList requires the use of methods to add elements to the collection.

Compatibility: Arrays are compatible with traditional for-loops and can be easily passed to other methods, while ArrayList requires the use of a special for-each loop and may require more code to be passed to other methods.

Assignment done by RAKESH GARAI