1. What is the Collection framework in Java?

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

2. What is the difference between ArrayList and LinkedList?
Ans:ArrayList uses a dynamic array to store the elements.
This means that the ArrayList maintains a contiguous block of memory to hold the elements, and when the ArrayList needs to grow or shrink in size, it
creates a new, larger or smaller array and copies the elements over. This can be inefficient for large arrays or frequent insertions/deletions.

LinkedList, on the other hand, uses a doubly-linked list to store the elements. This means that each element has a reference
to the next and previous elements in the list, allowing for efficient insertion and deletion of elements at any position in the list.
However, accessing elements in the middle of the list can be slower than in ArrayList, since it requires traversing the list from the beginning or end
to reach the desired element.

3. What is the difference between Iterator and ListIterator?
Ans:The main difference between Iterator and ListIterator is that ListIterator is a subinterface of Iterator that provides additional functionality for iterating over lists. Here are some key differences between the two:

Direction: Iterator can only iterate in one direction (forward), whereas ListIterator can iterate in both directions (forward and backward).
Access: Iterator can only access the next element in the collection, whereas ListIterator can access the next element, the previous element, and the current position in the list.
Modification: Iterator can only remove elements from the collection during iteration, whereas ListIterator can both remove and add elements at any position in the list.
Implementation: Iterator is implemented by all collection classes, whereas ListIterator is only implemented by list classes, such as ArrayList and LinkedList.

4. What is the difference between Iterator and Enumeration?
Ans:Direction: Iterator can only iterate in one direction (forward), whereas Enumeration can only iterate in one direction too (forward).

Access: Iterator can access the next element in the collection and optionally remove it, whereas Enumeration can only access the next element in the collection.

Modification: Iterator can remove elements from the collection during iteration, whereas Enumeration cannot remove elements from the collection.

Implementation: Enumeration is an older interface that was used in earlier versions of Java, whereas Iterator is a newer interface that was introduced in Java 1.2. As a result, Iterator has some advantages over Enumeration, such as being more flexible and efficient.

5. What is the difference between List and Set?
Ans: The List anp Set both extenp the collection interfaceA However, there are some pifferences between the two
which are listep below@
o The List can contain puplicate elements whereas Set inclupes unique items@
o The List is an orperep collection which maintains the insertion orper whereas Set is an unorperep collection
which poes not preserve the insertion orper@
o The List interface contains a single legacy class which is Vector class whereas the Set interface poes not
have any legacy class@
o 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 anp TreeSet are implementations of the Set interface in 2ava, but they have some
pifferences in terms of their properties anp usage1
o Ordering: HashSet is an unorperep collection of elements, while TreeSet is a sortep set of elements basep on
their natural orper or a custom comparator@

o Duplication: HashSet poes not allow puplicate elements, while TreeSet poes not allow puplicates as well@

o Implementation: HashSet is implementep using a hash table, while TreeSet is implementep using a self-
balancing binary search tree (Rep-Black tree)@

o Performance: HashSet has constant-time complexity O(1) for apping, removing, anp testing the existence of

an element, while TreeSet has a logarithmic-time complexity O(log n) for these operations pue to the self-
balancing property@

o Memory usage: HashSet uses less memory than TreeSet because it only stores the elements, while TreeSet
stores appitional information for maintaining the orper@
o Iteration: HashSet provipes no guarantees regarping the orper of iteration, while TreeSet guarantees the
elements are iteratep in sortep orper@
o Usage: HashSet is suitable when orpering is not important, anp fast access anp membership tests are
neepepA TreeSet is suitable when elements neep to be sortep or accessep in a specific orper.

7. What is the difference between Array and ArrayList?
Ans: Both arrays anp ArrayLists are usep to store collections of elements in 2ava, but they have some
pifferences in terms of their properties anp usage1
o Type: Arrays can store elements of primitive pata types as well as objects, while ArrayList can only store
objects@
o Size: The size of an array is fixep once it is createp, while the size of an ArrayList can be pynamically
increasep or pecreasep by apping or removing elements@
o Mutability: Arrays are mutable, meaning that you can mopify the elements in an array after it has been
createpA ArrayList is also mutable, but the only way to mopify it is by apping, removing or mopifying
elements@
o Performance: Arrays have better performance than ArrayLists for certain operations, such as accessing
elements by inpex, because they are implementep as a continuous block of memoryA ArrayLists, on the other
hanp, use pynamic memory allocation anp are implementep as a pynamic array, which may result in more
memory overheap anp slower performance for certain operations@

o Methods: Arrays have a limitep set of methops comparep to ArrayLists, which provipes more methops for
manipulating the collection, such as apping, removing, anp sorting elements.

E IniFializaFion: Arrays can be initialiOed with values at the time of creation, while ArrayList requires the use of
methods to add elements to the collectionN

+ E CompaFibiliFy: 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.