|  |  |
| --- | --- |
| **ArrayList** | **Vector** |
| **ArrayList** is not synchronized. Hence it is not thread safe. | Vector is **synchronized**, which means only one thread at a time can access the code, while arrayList is**not synchronized**, which means multiple threads can work on arrayList at the same time. For example, if one thread is performing an add operation, then there can be another thread performing a remove operation in a multithreading environment. |
| ArrayList increments 50% of the current array size if the number of elements exceeds its capacity. | Vector increments 100% – essentially doubling the current array size. |
| **ArrayList is faster**, since it is non-synchronized. | vector operations give slower performance since they are synchronized (thread-safe) |
| ArrayList can only use **Iterator** for traversing. | Vector can use both [**Enumeration and Iterator**](https://www.geeksforgeeks.org/iterators-in-java/) for traversing over elements of vector. |
| **Syntax:**  ArrayList<T> al = new ArrayList<T>(); | **Syntax:**  Vector<T> v = new Vector<T>(); |

|  |  |
| --- | --- |
| **HashSet** | **SortedSet** |
| HashSet contains unique elements only. | SortedSet provides a total ordering of its elements |
| HashSet allows null value | SortedSet doesn’t allows null values |
| HashSet doesn't maintain the insertion order. Here, elements are inserted on the basis of their hashcode | It maintains total ordering of elements |
| HashSet class is non synchronized. |  |
| For operations like search, insert and delete. It takes constant time for these operations on average. |  |

|  |  |
| --- | --- |
| **HashSet** | **TreeSet** |
| Elements in HashSet are not ordered. | TreeSet maintains objects in Sorted order defined by either Comparable or Comparator method in Java. |
| HashSet allows null value | TreeSet doesn’t allows null value |
| Elements in HashSet are not ordered. | TreeSet maintains objects in Sorted order defined by either Comparable or Comparator method in Java. |
| HashSet allows null object. | TreeSet doesn’t allow null Object and throw NullPointerException, |
| For operations like search, insert and delete. It takes constant time for these operations on average. | TreeSet takes O(Log n) for search, insert and delete |

|  |  |
| --- | --- |
| **Array** | **List** |
| An array is basic functionality provided by Java. | List is part of collection framework in Java. |
| It doesn’t allows positional access and insertion of elements, elements are inserted at end. | it allows positional access and insertion of elements. |
| The whole Arrays is traversed to find any element**.** | List provides methods to search element and returns its numeric position. |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Set** | **List** |
| An array is basic functionality provided by Java. | List is part of collection framework in Java. |
| *IT doesn’t allow duplicate entries.* | it allows positional access and insertion of elements. |
|  | List provides methods to search element and returns its numeric position. |
| A Set is a Collection that cannot contain duplicate elements. |  |
|  |  |

|  |  |
| --- | --- |
| **NavigableSet** | **NavigableMap** |
| NavigableSet represents a navigable set in [Java Collection Framework](https://www.geeksforgeeks.org/collections-in-java-2/). | NavigableMap is an extension of [SortedMap](http://www.contribute.geeksforgeeks.org/sortedmap-in-java-with-examples/" \t "_blank) which provides convenient navigation method. |
|  |  |
|  |  |
|  |  |
|  |  |