**National University**

**Of Computer & Emerging Sciences**

|  |  |
| --- | --- |
| **ArrayList** | **Vector** |
| * arrayList is**not synchronized**, which means multiple threads can work on arrayList at the same time | * Vector is **synchronized**, which means only one thread at a time can access the code |
| * **ArrayList is faster** | * vector operations give slower performance |
| * ArrayList increments 50% of the current array size | * vector increments 100% |
| * Vector can use both [Enumeration and Iterator](https://www.geeksforgeeks.org/iterators-in-java/) for traversing | * ArrayList can only use **Iterator** for traversing. |

**Advance Programming**

**Roll #:16-4255**

**Name: Syed Najam Ul Hasan**

|  |  |
| --- | --- |
| **HashSet** | **SortedSet** |
| * HashSet doesn't maintain the insertion order. Here, elements are inserted on the basis of their hashcode. | * set sorted in an ascending order |

|  |  |
| --- | --- |
| **HashSet** | **TreeSet** |
| * constant time performance for the basic operations | * log(n) time cost for the basic operations |
| * does not guarantee that the order of elements will remain constant over time | * that elements of set will be sorted |
| * iteration performance depends on the initial capacity and the load factor of the HashSet. | * doesn't offer any tuning parameters for iteration performance |
| * duplicate-free collection of elements | * duplicate-free collection of elements |

|  |  |
| --- | --- |
| **Array** | **LIST** |
| * An array is the data structure that contains a collection of similar type data elements | * Linked list is considered as non-primitive data structure contains a collection of unordered linked elements known as nodes. |
| * In the array the elements belong to indexes | * You have to iterate to get the element |
| * Takes linear time to search | * Takes O(n) time to search |
| * Arrays are of fixed size | * Linked lists are dynamic and flexible and can expand and contract its size |
| * memory is assigned during compile time | * memory is assigned during compile time |
| * memory utilization is inefficient in the array | * memory utilization is inefficient in the array |

|  |  |
| --- | --- |
| **List** | **Set** |
| * List is an ordered collection it maintains the insertion order, which means upon displaying the list content it will display the elements in the same order in which they got inserted into the list. | * Set is an unordered collection, it doesn’t maintain any order. There are few implementations of Set which maintains the order such as LinkedHashSet |
| * List allows duplicates | * Set doesn’t allow duplicate elements |
| * List allows any number of null values | * Set can have only a single null value at most. |
| * ListIterator can be used to traverse a List in both the directions | * it cannot be used to traverse a Set |
| * List interface has one legacy class called [Vector](https://beginnersbook.com/2013/12/vector-in-java/) | * + Set interface does not have any legacy class. |

|  |  |
| --- | --- |
| **NavigableSet** | **NavigableMap** |
| * doesn't allow duplicates | * It may contain duplicate values but keys are always unique |
| * Set just allow one null element as there is no duplicate permitted | * null element as there is no duplicate permitted while in Map you can have null values and at most one null key. |
| * If you want to create a collection of unique elements and don't want any duplicate than choosing any Set implementation | * If you store data in form of key and value than Map is the way to go |