Name: Rabiya Adnan
Roll No.: 16L-4356

Advanced Programming – Assignment 2

ArrayList vs Vector

ArrayList	Vector
ArrayList increments the size of the list by 50% when the total elements exceed the capacity.	Vector doubles the size of the array by 100%, i.e. it doubles the size of the array if the total elements exceeds the capacity.
ArrayList is fast because it is non-synchronized.	Vector is slow because it is synchronized.
ArrayList uses the *Iterator* interface to traverse the elements.	A vector can use the *Iterator* interface or *Enumeration* interface to traverse the elements.

HashSet vs SortedSet

HashSet	SortedSet
The underlying data structure to store data is a *Hash-table.*	The underlying data structure to store data is a *red-back tree,* which is a balanced binary tree.
HashSet stores elements in a unsorted order.	Sorted set stores the data in a sorted order.
The HashSet uses a hash table to do basic operations (Add, Remove, Search) independent of the input size.	SortedSet does basic operations (Add, Remove, Search) dependent on the input size.

TreeSet vs HashSet

TreeSet	HashSet
Null values are not allowed. Will throw a NullPointerException.	Null values are allowed in HashSet.
TreeSet maintains the order of the elements.	HashSet doesn't guarantee any order.
TreeSet is backed by TreeMap.	HashSet is backed by HashMap.

Array vs List

Array	List
Fixed size.	List is grow-able in nature.
No ready-made method support like sorting, iteration, etc.	List uses built in methods, e.g. iterator() to traverse lists.
Holds only homogeneous data, e.g. Student s[] = new Student[10]; s[0] = new *Student()*;	Can hold non-homogeneous data, e.g. Student s[] = new Student[10]; s[0] = new *Employee()*;

List vs Set

List	Set
A List is an ordered grouping of items.	A Set is an unordered grouping of items with no duplicates allowed.
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(forward and backward), but it cannot be used to traverse a set.	We use *Iterator* to traverse a set.

NavigableSet vs NavigableMap

NavigableSet	NavigableMap
The NavigableSet interface inherits from the SortedSet interface.	NavigableMap is an extension of SortedMap.
It behaves like a SortedSet with the exception that we have navigation methods available in addition to the sorting mechanisms of the SortedSet.	Along with these popular navigation method it also provide ways to create a Sub Map from existing Map in Java.
The classes that implement this interface are, TreeSet and ConcurrentSkipListSet.	An example class that implements NavigableMap is TreeMap.