Name: John Emil Ramzy **Task:** 4 (Search)

Collections

Vector:

Advantages:

- Synchronized *(thread-safe)*.
- Allows random access via index *(like an array)*.
- Can grow dynamically
- Maintains insertion order.

Disadvantages:

- Slower than alternatives *(like ArrayList)* due to synchronization overhead.

ArrayDeque:

Advantages:

- Resizable array implementation of a double ended queue
- Faster than **Stack** and **LinkedList** for stack/queue operations
- No capacity restrictions
- Supports insertion/removal from both ends in constant time

Disadvantages:

- Not thread safe
- Doesn't allow **null** elements
- No random access by index *(like a list)*

PriorityQueue:

Advantages:

- Implements a **min-heap** by default *(smallest element is at the head).*
- Elements are ordered based on **natural ordering** or a **custom comparator**
- Useful for scheduling,

Disadvantages:

- Not thread safe
- No random access; **only head access is fast.**
- Doesn't allow **null** elements.
- Doesn't maintain insertion order.

HashSet

Advantages:

- Fast operations **(add, remove, contains)** in constant time on average.
- No duplicates allowed.
- Backed by a **HashMap**.

Disadvantages:

- No guaranteed order of elements.
- Performance depends on proper **hashCode()** and **equals()** implementation.
- Not thread-safe.

LinkedHashSet:

Advantages:

- Maintains **insertion order**.
- No duplicates.
- Fast access and insertion *(slightly slower than HashSet due to ordering)*.

Disadvantages:

- Uses more memory than **HashSet** *(because of linked list internally)*.
- Not thread-safe.

TreeSet

Advantages:

- Elements are **sorted** in natural order *(or via comparator)*.
- No duplicates.
- Can be used to perform range queries, subset operations efficiently.

Disadvantages:

- Slower than **HashSet** and **LinkedHashSet** *(O(log n) time complexity)*.
- Requires elements to be **Comparable** or passed a Comparator.
- Not thread-safe.
- Doesn't allow null elements *(throws NullPointerException)*.