Collections (Interface)

List {ordered, duplicate, has index, null, empty, o/p same order

Array List:

add, addAll, remove, removeall

LinKed List:

same as Array List but no index

Vector: Doesnot allow null & duplicate

same as Array list

contains, insertElementAt, capacity, removeElementAt, firstElement, size, lastElement, clone, isEmpty.

Stack:

push or add, peek, remove, pop, empty

Queue: No ordered

↳ Priority Queue

add, element, peek, remove, poll

Null - null pointer exception

allows - empty, duplicate.

poll - remove first

No queue - exception in .element(),

No queue - null value in .peek().

Deque: ordered, no index

↳ Array Deque

is faster than Array list & stack

Null - null pointer exception

allows - empty, duplicate

add, offer first, offer last

poll - remove first word, poll last

Set {no index, no duplicate, unordered o/p }

Hash Set

add, addAll

LinKed Hash Set:

allows null & duplicate.

order, has index because it uses index engine

add, addAll

Sorted Set

Tree Set: faster than both the hash Sets.

No duplicate

Null - null pointer exception

prints the o/p in ascending order

add, addAll.

Map

Hash Map:

unordered

put, putifabsent, remove, replace

putAll

Allows null

<, > dmble object

Key, value can be duplicate

LinKed Hash Map

<, >

Key, value

ordered

Same as hashmap

TreeMap:

put, remove, descending map.

allows empty

Null - null pt exception

<, > - dmble objects

Key, value

allows duplicate value

prints the O/P in ascending order

classes

- getOrDefaultH (2, "DefaultValue"); <-

o/p: prints the value

Hash Table (Map) - like list

<, > no null in Key & value

O/P: descending based on key