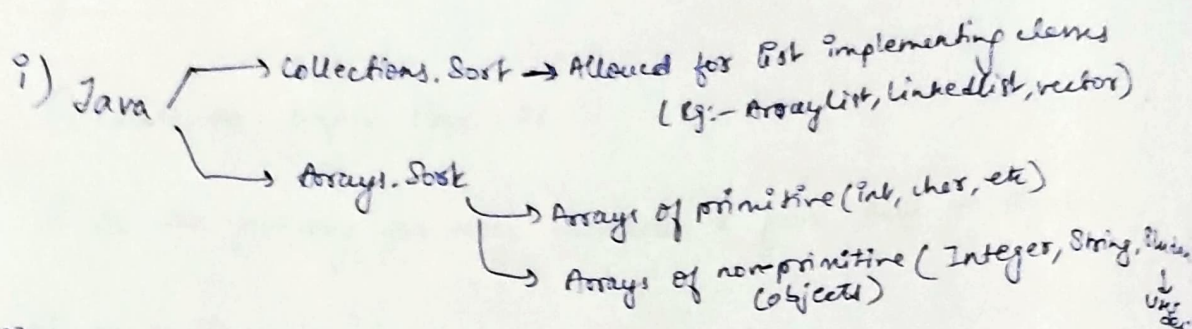


Sorting

① Arrays.sort in Java



ii) In objects sorting is to be stable.
not important in case of primitives.

iii) Arrays.sort by → java.util.Arrays;
Sorts in place i.e changes the array.

→ Only sorts in ascending (increasing) order.

for decreasing order, Iterator can be used to reverse after sorting
Collections.reverse() can also be used.

iv) Arrays.sort(arr);

Arrays.sort(arr, 1, 4); → Sorting a subarray.

★ ★ In user defined class for using Arrays.sort(), comparable class
is to be used.

→ replaces int, to implement collections.

v) Wrapper class eg:- Integer arr[] = {5, 20, 10, 12};

Arrays.sort(arr, Collections.reverseOrder());

vi) Collections.sort → Import java.util.*

Eg:- → `List<Integer> list1 = new ArrayList<Integer>();`
`list1 list1.add(20);`
`list1.add(5);`
`Collections.sort(list1);`
`System.out.println(list1);`

vii) → `Arrays.sort(arr, new MyComp);` } while using comparator
`Collections.sort(arr);` } interface.

viii) Stability in Sorting.

Original Array → ('CSE', 10), ('ECE', 15), ('CSE', 5), ('ECE', 20)

Stable sorted → ('CSE', 5), ('CSE', 10), ('ECE', 15), ('ECE', 20)

Unstable sorted → Any other combination.

∴ The point is (In above ~~array~~ ^{obj}) if sorted by numbers then the original order in which names appeared must be preserved. if numbers are same.