Arrays.sort()

Parameter → an array

Arrays.sort(arr[]);

Result → It will make arr as sorted in Ascending order

arr[] : {100, 50, 60, 900, 60, -2}

Sorting Algorithm

↳ Bubble sort, selection sort, insertion sort

Arrays.sort() ← arr

↳ Tim Sort (Merge Sort + Insertion Sort)

Algorithm

Time Complexity → nlogn

Example:

arr: [100, 60, 30, 9, 1005]

↓

Arrays.sort(arr);

```
Arrays.sort(arr);
for(int i=0; i<n; i++){
  S.o.p/n(arr[i]);
  }
```

Sort Array in Descending order

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

↓ Descending order

Comparator

↳ custom sort → Using this we can have our own logic to sort the objects

Arrays.sort → ascending order.

Employee →

salary	age
100000	20
50000	30
60000	18
90000	20
85000	20

Comparator is an interface inside java.util package

```
int compare(<T>a, <T>b);
```

→ TimSort

→ TimSort

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

↓

Comparator

```
Class Main {
    p.s.v.m(String[] args) {
        Integer int[] = {250, 20, 30, 4};
Comparator<Integer> sortDesc
1st method          = (a,b) -> a-b ;
    Arrays.sort(arr, sortDesc);
}
```

2nd method

```
Class A implements Comparator<Integer>{
    int Compare(Integer a, Integer b) {
        return b-a;
    }
```

a-b

a: -5 ↳ -5 - (-6)

b: -6 ↳ -5 + 6 = (1)

```
Class B {
    public void method1() {
        A aobj = new A();
        aobj.compare(10, 30);
    }
```