13 June 2023 19:38

Sorting - 2

arr ⟶ [5, 4, 2, 1, 6]

↓ sort

arr ⟶ [6, 5, 4, 2, 1]

Inbuilt sort

1). Handle duplicates

2). Empty array / array of size one

3). Negative elements

4). Custom sorting

arr = { 1, 2, 3, 7, 5, 0 } → { 0, 1, 2, 3, 5, 7 }

Arrays.sort(arr); ⇒ Increasing order (by default)

→ fn

→ Class

Data types

- Primitive (defined by Java) ⟶ int, float, boolean
- Non-primitive (user-defined) ⟶ String, ArrayList, user-defined, Wrapper classes

1). Primitive arrays ⇒ int[], float[], boolean[]

→ Stack

→ No Custom Sorting

→ only sort in incr. order

Dual pivot quick sort ⇒ Time - O(nlogn)

Space - O(1)

2). Non-primitive arrays ⇒ Integer[], String[]

→ Heap

Tim sort

- → Insertion sort (n ≤ 16)
- → merge sort (n > 16)

⇓⇓

Time - O(nlogn)

Space - O(n)

Data type to store my salary

- → int
- → Integer → Non-primitive data type

→ Integer → Non-primitive data type

↓

Wrapper class (numerical value)

Comparator ⟶ logic to compare two values

Comparator will only work for non-primitive values.

default :⇒ 10, 15 ✓

15, 10 ⇒ 10, 15 ⇒ swap

Comparator → class
→ Anonymous class
→ Lambda expn or arrow fn.

```
Arrays.sort(arr, (a,b) -> integer);
```

(a, b)

a before b ⇒ ⊖ve value {no swap}

b before a ⇒ ⊕ve value {swap}

a == b (same priority) ⇒ 0 value

Increasing order

arr ⟶ [10, 15]	arr ⟶ [8, 5]
a: 10, b: 15	a: 8, b: 5
⊖ve value ⇒ a - b	⊕ value ⇒ a - b

a - b

⇓

```
Arrays.sort(arr,(a,b) -> a - b);
```

Decreasing order

arr ⟶ [9, 7]	arr ⟶ [6, 10]

arr → [9 , 7]

a : 9 , b : 7

⊖ ve value ⇒ b - a

arr → [6 , 10]

a : 6 , b : 10

⊕ ve value ⇒ b - a

b - a

```
Arrays.sort(arr, (a,b) -> b-a);
```

int	Integer
float	Float
boolean	Boolean
double	Double

Data type

Wrapper class

Sort the array according to their Square of each element

Sample Input 0

```
5
4 -1 0 -5 6
```

Sample Output 0

```
0 -1 4 -5 6
```

-5 -1 0 4 6

4 1 0 5 6

0 1 4 3 6

0 -1 4 -5 6

$(-1)^2 = 1$

$(-5)^2 = 25$

arr[i] > arr[j] ✗

arr[i]*arr[i] > arr[j]*arr[j] ✓

Arrays.sort

arr → [-5, 6] → [-5, 6]

a = -5 , b : 6

⊖ ve value ⇒ a*a - b*b

arr → [6, -5] → [-5, 6]

a : 6 , b : -5

⊕ ve value ⇒ a*a - b*b

arr → [7 , 9] → [7, 9]

a: 7 , b: 9

⊖ ve value :⇒ a*a - b*b

arr → [-9 , -7] → [-7, -9]

a: -9 , b: -7

⊕ ve value :⇒ a*a - b*b

arr → [-9 , 7] → [7 , -9]

a: -9 , b: 7

⊕ ve value :⇒ a*a - b*b