Bubble Sort.
Selection Sort.
Inbuilt Sort —

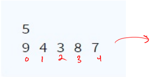
* Bubble Sort:- each itr move largest number at end.

* Selection Sort:- each itr move smallest number at beginning.

Arrays.sort( ); → ascending order.

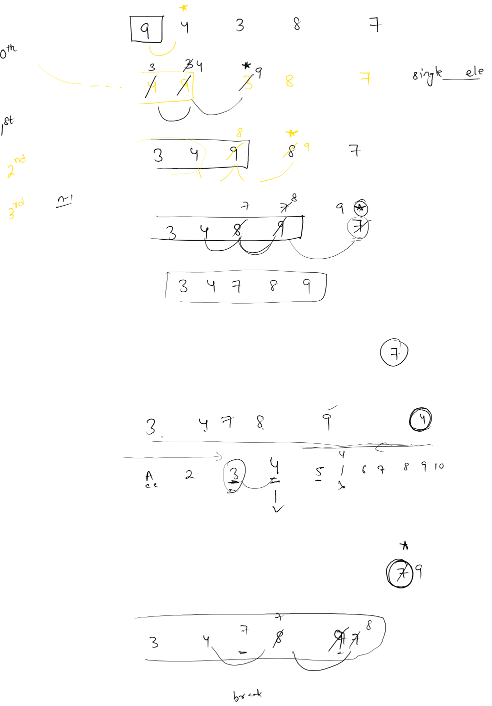
default behaviour in Java.

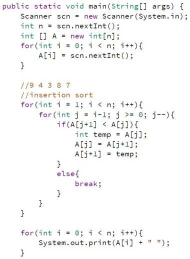
Insertion Sort:-



```
5
9 4 3 8 7
```

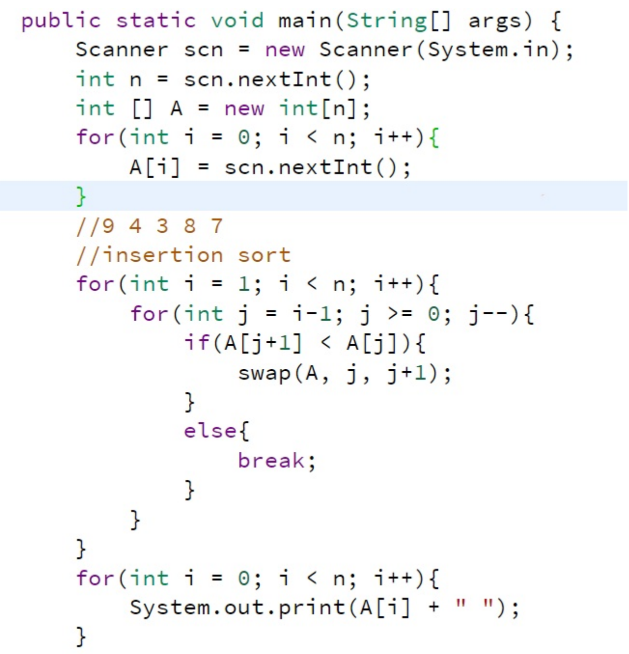
→ 3 4 7 8 9



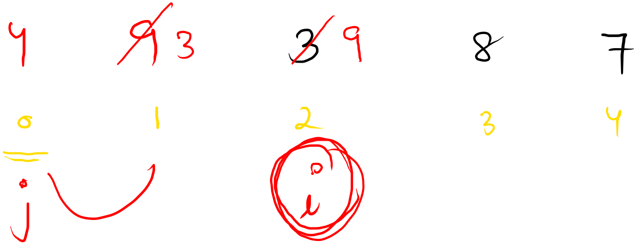


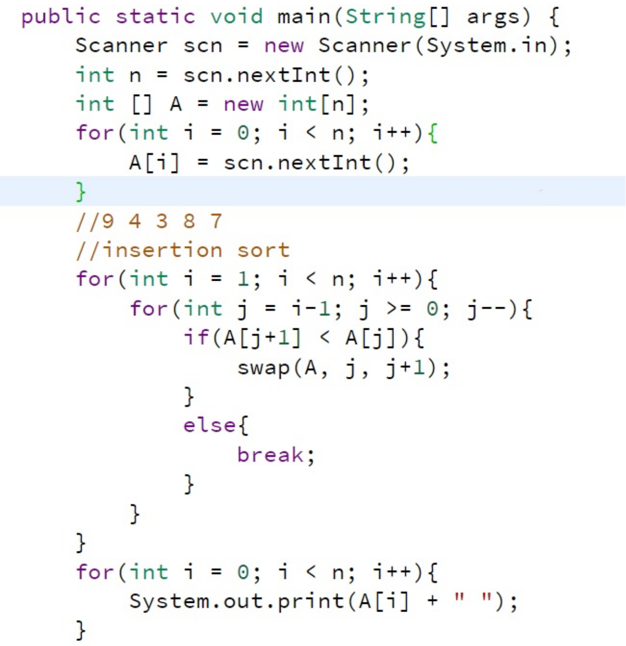
```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [] A = new int[n];
    for(int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }

    //9 4 3 8 7
    //insertion sort
    for(int i = 1; i < n; i++){
        for(int j = i-1; j >= 0; j--){
            if(A[j+1] < A[j]){
                int temp = A[j];
                A[j] = A[j+1];
                A[j+1] = temp;
            }
            else{
                break;
            }
        }
    }

    for(int i = 0; i < n; i++){
        System.out.print(A[i] + " ");
    }
```

Insertion sort with swap function.



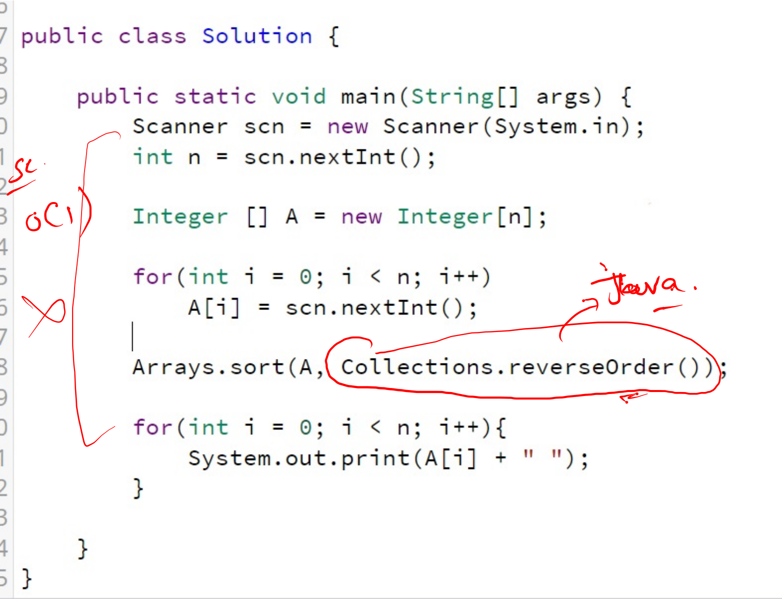
```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [] A = new int[n];
    for(int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }
    //9 4 3 8 7
    //insertion sort
    for(int i = 1; i < n; i++){
        for(int j = i-1; j >= 0; j--){
            if(A[j+1] < A[j]){
                swap(A, j, j+1);
            }
            else{
                break;
            }
        }
    }
    for(int i = 0; i < n; i++){
        System.out.print(A[i] + " ");
    }
```





```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [] A = new int[n];
    for(int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }
    //9 4 3 8 7
    //insertion sort
    for(int i = 1; i < n; i++){
        for(int j = i-1; j >= 0; j--){
            if(A[j+1] < A[j]){
                swap(A, j, j+1);
            }
            else{
                break;
            }
        }
    }
    for(int i = 0; i < n; i++){
        System.out.print(A[i] + " ");
    }
```

Decreasing order sorting using inbuilt method.



```java
public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        Integer [] A = new Integer[n];

        for(int i = 0; i < n; i++)
            A[i] = scn.nextInt();

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

        for(int i = 0; i < n; i++){
            System.out.print(A[i] + " ");
        }

    }
}
```

SC

O(1)

Java.

Comparator:

↳ used to compare 2 obj.

Comparator.

↳ compare function.

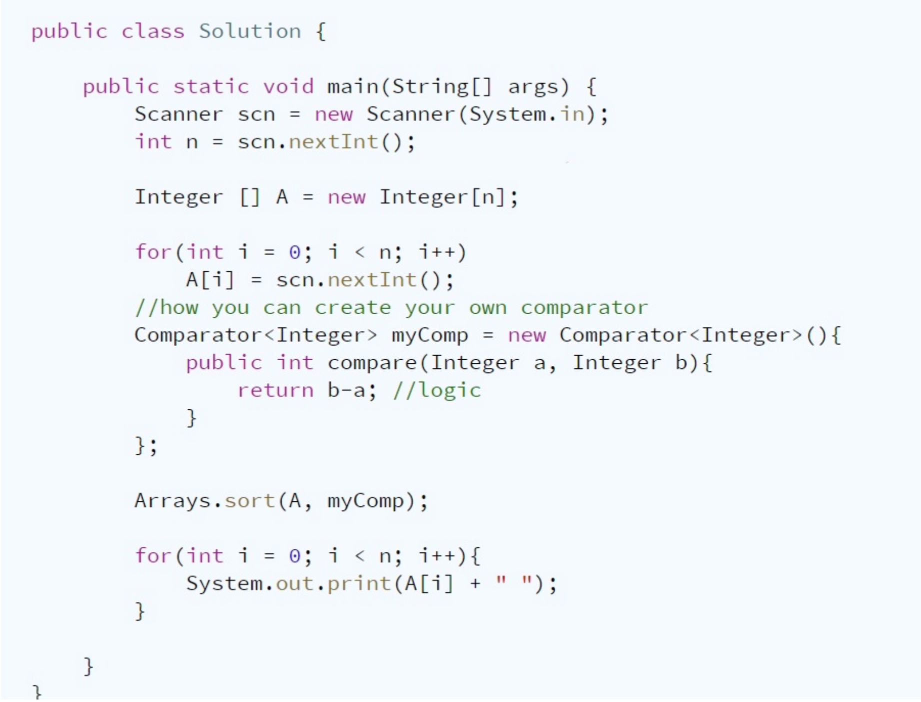
Integer a, Integer b

return → (int)

→ 1 (positive) ⇝ swap

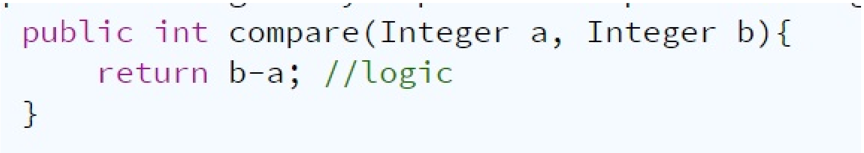
→ 0

→ -1 (negative)



```java
public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        Integer [] A = new Integer[n];

        for(int i = 0; i < n; i++)
            A[i] = scn.nextInt();
        //how you can create your own comparator
        Comparator<Integer> myComp = new Comparator<Integer>(){
            public int compare(Integer a, Integer b){
                return b-a; //logic
            }
        };

        Arrays.sort(A, myComp);

        for(int i = 0; i < n; i++){
            System.out.print(A[i] + " ");
        }

    }
}
```

b - a → desc.



```
public int compare(Integer a, Integer b){
    return b-a; //logic
}
```

a - b → incear.

asce

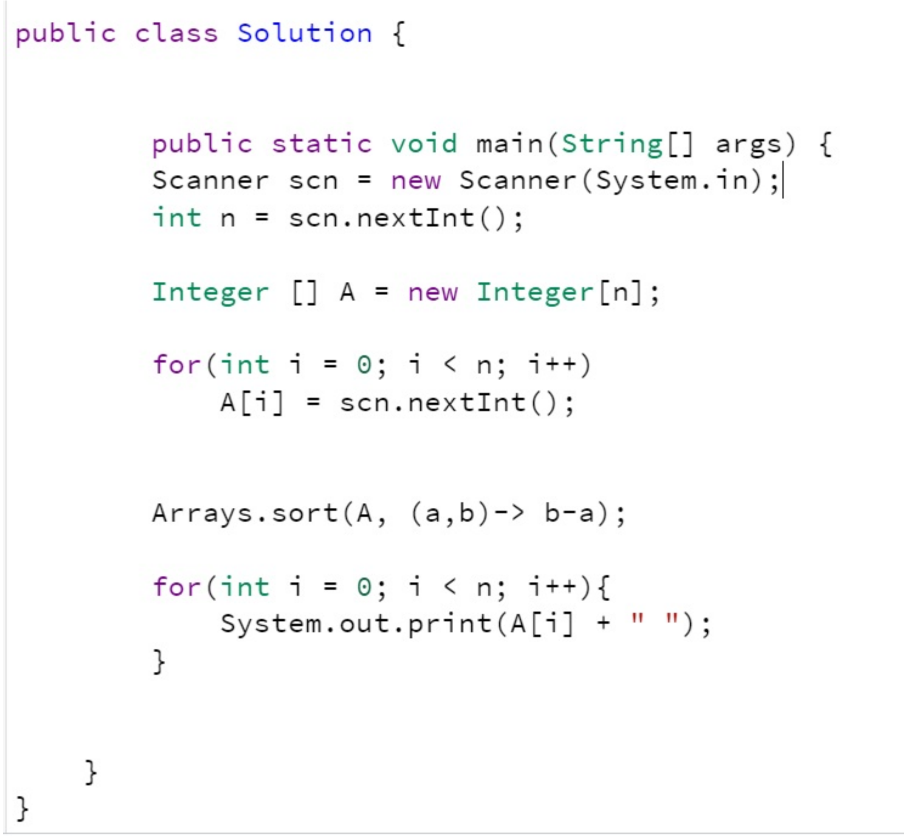
a b

a b

2 4

2 - 4 -ve.

a < b .



```
public class Solution {


        public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        Integer [] A = new Integer[n];

        for(int i = 0; i < n; i++)
            A[i] = scn.nextInt();


        Arrays.sort(A, (a,b)-> b-a);

        for(int i = 0; i < n; i++){
            System.out.print(A[i] + " ");
        }


    }
}
```