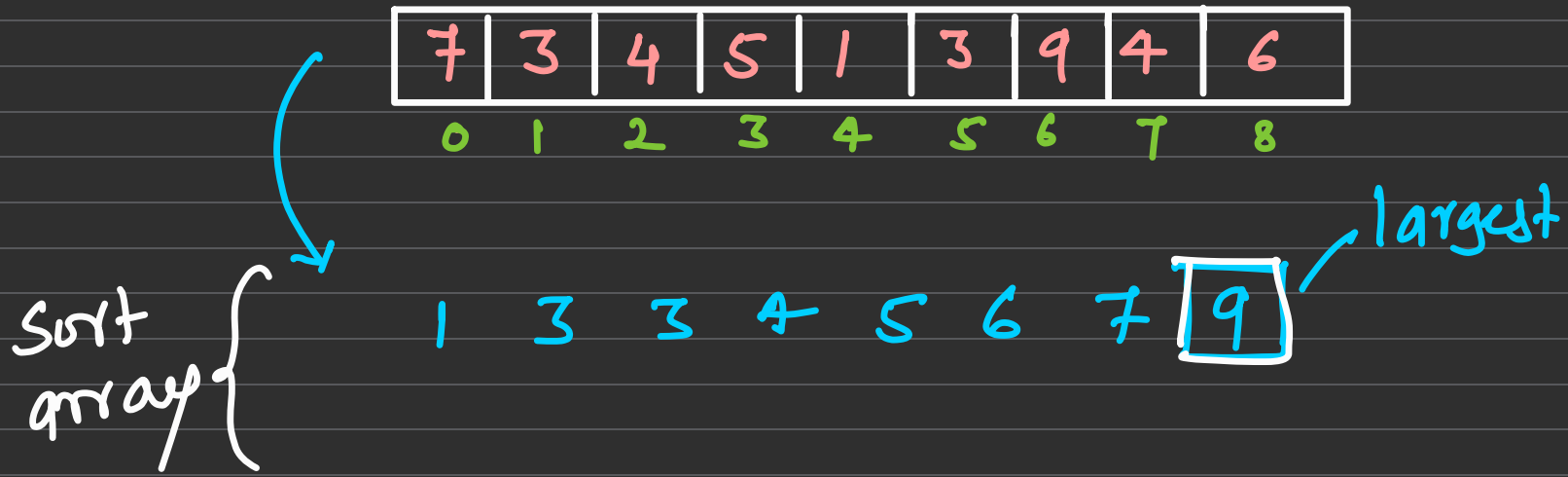
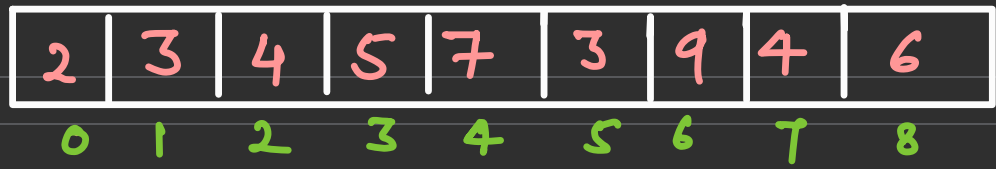
Arrays and Its Problems

Find the greatest number from the array.



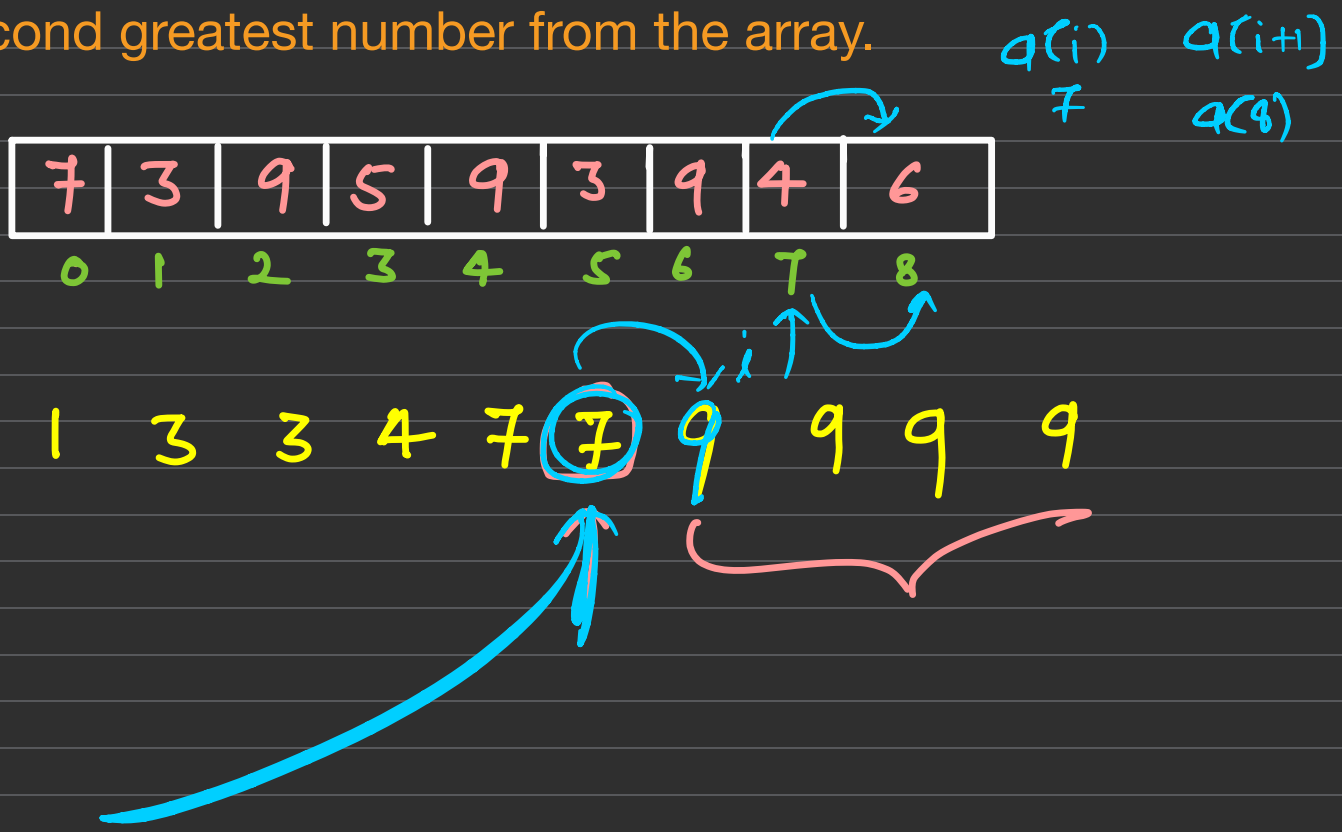



```
int largestNum(arr[], N)
{
    int largest = a[0]

    for(i=1 ; i<N ; i++)
    {
        if(a[i] > largest)
        {
            largest = a[i]
        }
    }
    return largest
}
```

O(N)

Find the second greatest number from the array.



Sort (arr)

N = 1

(7)

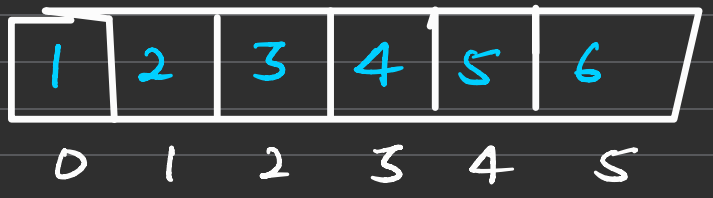
-1

```
for (i = n-2 ; i >= 0 ; i--)
{     if (a[i] != a[i+1])
          return a[i]
}

return -1
```

```
if (n == 1)
    return -1
```

Rotate array to left by K Position



N = 6

K = 3

K = 1

0 1 2 3 4 5

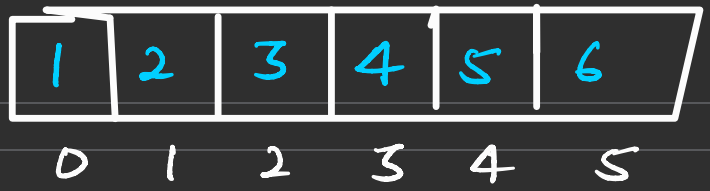
6 1 2 3 4 5

K = 2

5 6 1 2 3 4

K = 3

4 5 6 1 2 3



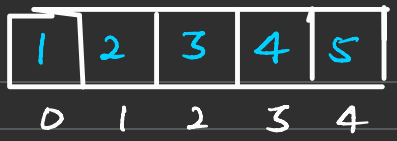
N = 6

K = 3

```
for(j=1 ; j≤K ; j++)                    O(K)
{
    temp = a[n-1]                       O(N)
    for(i=n-2 ; i≥0 ; i--)
    {
        a[i+1] = a[i]
    }
    a[0] = temp
}
```

O(N x K)

4 % N = 4



N = 5

K = 9 % N = 4

K=1 5 1 2 3 4

K=2 4 5 1 2 3

K=3 3 4 5 1 2

K=4 2 3 4 5 1

K=5 1 2 3 4 5

K=6 5 1 2 3 4

K=7 4 5 1 2 3

K=8 3 4 5 1 2

K=9 2 3 4 5 1

K=10 1 2 3 4 5

$$K = K \% N$$

K = 3

N = 7

5 6 7 1 2 3 4

7 6 5 4 3 2 1

K N-K

5 6 7 1 2 3 4

K % N == 0 ——— arr

- find smallest element from array
- - 1 = second smallest
- Rotate Right by K time

1 2 3 4 5

K=2 3 4 5 1 2

Find Union of two arrays.

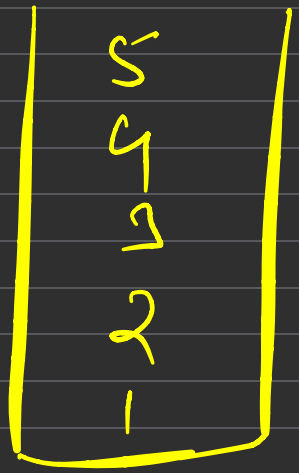
Sorted

duplicates

Input: a[] = [1, 2, 3, 4, 5], b[] = [1, 2, 3, 6, 7]
Output: 1 2 3 4 5 6 7
Explanation: Distinct elements including both the arrays are: 1 2 3 4 5 6 7.

Input: a[] = [2, 2, 3, 4, 5], b[] = [1, 1, 2, 3, 4]
Output: 1 2 3 4 5
Explanation: Distinct elements including both the arrays are: 1 2 3 4 5.

Input: a[] = [1, 1, 1, 1, 1], b[] = [2, 2, 2, 2, 2]
Output: 1 2
Explanation: Distinct elements including both the arrays are: 1 2.



What
Why
How
Use

No duplicate

Quick look

find

Intersection of Two arrays with Duplicate Elements

Difficulty: **Easy** Accuracy: **61.4%** Submissions: **40K+** Points: **2** Average Time: **20m**

Given two integer arrays **a[]** and **b[]**, you have to find the **intersection** of the two arrays. **Intersection** of two arrays is said to be elements that are common in both the arrays. The intersection should not have **duplicate** elements and the result should contain items in **any order**.

Note: The driver code will **sort** the resulting array in increasing order before printing.

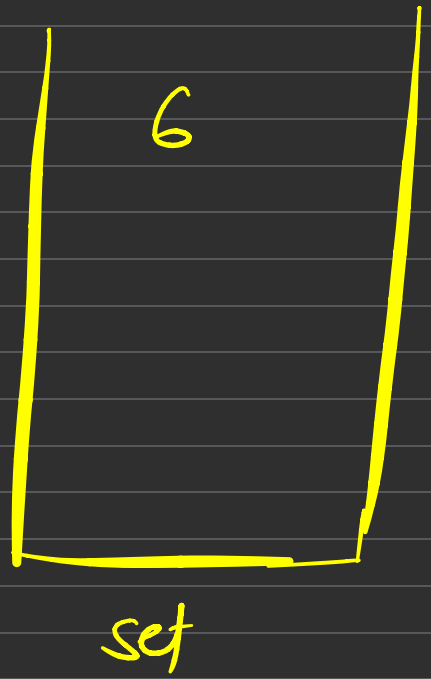
Examples:

Input: a[] = [1, 2, 1, 3, 1], b[] = [3, 1, 3, 4, 1]
Output: [1, 3]
Explanation: 1 and 3 are the only common elements and we need to print only one occurrence of common elements.

Input: a[] = [1, 1, 1], b[] = [1, 1, 1, 1, 1]
Output: [1]
Explanation: 1 is the only common element present in both the arrays.

$A = \{ 1\ 2\ 3\ 4\ 6 \}$

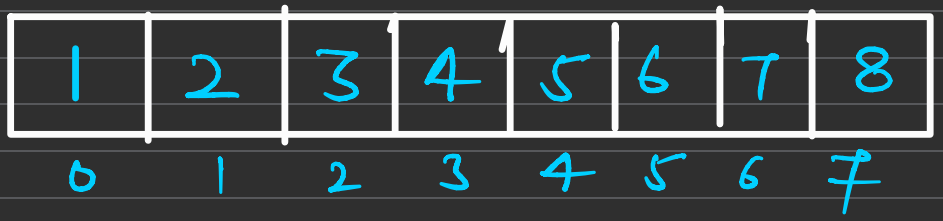
$B = \{ 1\ 2\ 5\ 3\ 4\ 2 \}$



1 2 3 4

1. put all elements of A in set
2. Compare elements of B in set
   1. if found print & erase from set
3. Return ans

arr:



Wave Array

Difficulty: **Medium** Accuracy: **63.69%** Submissions: **287K+** Points: **4**

Average Time: **20m**

Given an **sorted** array **arr[]** of integers. Sort the array into a **wave-like** array(In Place). In other words, **arrange the elements** into a sequence such that arr[1] >= arr[2] <= arr[3] >= arr[4] <= arr[5] ..... and so on. If there are multiple solutions, find the **lexicographically smallest** one.

Note: The given array is sorted in ascending order, and modify the given array in-place without returning a new array.

Examples:

Input: arr[] = [1, 2, 3, 4, 5]
Output: [2, 1, 4, 3, 5]
Explanation: Array elements after sorting it in the waveform are 2, 1, 4, 3, 5.

Input: arr[] = [2, 4, 7, 8, 9, 10]
Output: [4, 2, 8, 7, 10, 9]
Explanation: Array elements after sorting it in the waveform are 4, 2, 8, 7, 10, 9.