# ALG'18: Assignment-4
# Top $K^{th}$ Items

## Description

You are given two sorted arrays Arr1 and Arr2 of sizes N and M integers which can be positive, negative, or zero. You want to get the top $K^{th}$ element in the final sorted array.

Design an efficient algorithm that can solve this problem.  O(N) solution is not the most efficient solution for this problem. The required solution is better than O(N).

### Input: **Already Implemented**

The first line of input is an integer T (T < 30), that indicates the number of test cases.

Each case consists of 6 lines. Line 1 is the N, denoting N number of elements in Arr1. Line 2 is the M which is the number of elements in Arr2. The third line is the kth position. The fourth and the fifth lines consists of the elements of A and B respectively.

### Output: **Already Implemented**

The result is the item at the top $K^{th}$ position in the final sorted array.

### Function: **Implement it!**

```
public static int GetKthItem(int[] arr1, int[] arr2, int N, int M, int K)
```

It takes two arrays of N  and M integers and the $K^{th}$ position. It should return the element at the top $K^{th}$ position.

## Template

- C# template

# BONUS Description

Starting from the TOP $K^{th}$ item you got in the solution of the main problem, the bonus is to get the TOP $K^{th}$ items. You don't need to sort these items, only retrieve them.

So, design an efficient algorithm that can solve this problem. O(N) solution is not the most efficient solution for this problem. The required solution is better than O(N).

## Input [SAME as the main problem]: Already Implemented

The first line of input is an integer T (T < 30), that indicates the number of test cases.

Each case consists of 6 lines. Line 1 is the N, denoting N number of elements in Arr1. Line 2 is the M which is the number of elements in Arr2. The third line is the kth position. The fourth and the fifth lines consists of the elements of A and B respectively.

## Output: Already Implemented

The result is the top K integer array**.**

## Function [In the SAME template of the main problem]: Implement it!

```
public static int[] GetTopKItems(int[] arr1, int[] arr2, int N, int M, int
                                 K)
```

It takes two arrays of N and M integers and the $K^{th}$ position. It should return ALL the elements starting from the top $K^{th}$ position (TOP $K^{th}$ items).

# Test Cases

To test your code, there are two types of cases: Sample and complete cases.

## SAMPLE TEST CASES

**# of cases = 7, to run them:**

1- Run your program
2- Enter '1'

YOU MUST SEE **"Succeed"** for each sample case. Then congratulations at the end of all.

**Some of the sample cases:**

| # | Input Array | Output |
|---|---|---|
| 1 | 9<br>8<br>10<br>1 2 3 4 5 6 7 8 9<br>10 11 12 13 14 15 16 17<br>***[The 10th item FROM THE END (greatest 10th) if the two arrays merged = 8]*** | 8 |
| 2 | 9<br>4<br>1<br>1 1 1 2 2 2 3 3 3<br>4 4 5 5<br>***[The first item FROM THE END (greatest ONE) if the two arrays merged = 5]*** | 5 |
| 3 | 5<br>4<br>8<br>2 3 6 7 9<br>1 4 8 10<br>***[The 8th item FROM THE END (greatest 8th) if the two arrays merged = 2]*** | 2 |

## COMPLETE Test Cases

**# of cases = 3, To run them:**

1- Run your program
2- Enter '2'

OR CONTINUE ON THE RUN of the sample cases by choosing 'Y' when you asked to run the complete cases.

YOU MUST SEE **"Succeed"** for each sample case. Then congratulations at the end of all.

## BONUS Sample Test Cases

**# of cases = 7, to run them:**

1- Run your program
2- Enter '3'

YOU MUST SEE **"Succeed"** for each sample case. Then congratulations at the end of all.

OR CONTINUE ON THE RUN of the sample cases by choosing 'Y' when you asked to run the bonus sample cases.

## BONUS Complete Test Cases

**# of cases = 3, To run them:**

1- Run your program
2- Enter '4'

OR CONTINUE ON THE RUN of the sample cases by choosing 'Y' when you asked to run the complete cases.

YOU MUST SEE **"Succeed"** for each sample case. Then congratulations at the end of all.

# C# Help

If you need any help regarding the syntax of C#, **ask any TA**.

## Creating 1D array

```
int [] array1D = new int [size]
```

## Creating 2D array

```
int [,] array2D = new int [size1, size2]
```

## Getting the size of 1D array

```
int size = array1D.GetLength(0);
```

## Getting the size of 2D array

```
int size1 = array2D.GetLength(0);

int size2 = array2D.GetLength(1);
```