# Top K Numbers

## Description

You are given an array A of N integers which can be positive, negative, or zero. You want to select the Top (largest) K numbers, where k << N.

Design an efficient algorithm that can solve this problem. Expected case of O(NlogN) is not the most efficient solution for this problem. The required solution is better than O(NLogN).

### 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 two integers that represents the array size (N) and the number of top integers required (K), followed by the array items.

### Output: Already Implemented

The result is a K integer array in which the top K numbers are returned (**Top** First)

### Function: Implement it!

```
public static int[] RequiredFunction(int[] numbers, int k)
```

It takes an array of N integers and the k required numbers (N). It should return the k top elements with the **top first**. (k << N)

**PROBLEM_CLASS.cs** includes this method.

## Test Cases

| # | Input Array | Output |
|---|---|---|
| **1** | 9<br>3<br>-9 -5 2 -7 1 3 6 -8 7 | 7 6 3 |
| **2** | 8<br>4<br>4 0 -4 -7 9 3 7 3 | 9 7 4 3 |
| **3** | 9<br>1<br>-10 1 -7 7 2 5 -7 0 3 | 7 |
| **4** | 23<br>2<br>-12 -27 86 96 66 43 -56 94 85 78 11 -84 1 22 -31 45 -72 53 94 -6 -45 16 96 | 96 96 |

# C# Help

## 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);
```