Lesson:

Problems based on sortings - 3

Pre-Requisites

- Loops
- Arrays
- Sorting algorithms

List of concepts involved

- Program to find Kth smallest element in an array using QuickSort.
- Given two sorted arrays, Write a program to merge them in a sorted manner.

Problem 1 Write a program to find Kth smallest element in an array using QuickSort.

Input
Enter the elements of array
3 5 2 1 4 7 8 6
Enter the value for k
5
Output:
K'th smallest element is 5

Approach:
Run quick sort algorithm on the input array

- In this algorithm pick a pivot element and move it to it's correct position
- Now, if index of pivot is equal to K then return the value, else if the index of pivot is greater than K, then recur for the left subarray, else recur for the right subarray
- Repeat this process until the element at index K is not found

Code
https://pastebin.com/GPjyyZ1u

```
Enter the size of array
8
Enter the elements of array
3 5 2 1 4 7 8 6
Enter the value for k
5
K'th smallest element is 5
```

Time Complexity: O(N^2)
Auxiliary Space: O(1)

Problem 2 Given two sorted arrays, Write a program to merge them in a sorted manner.

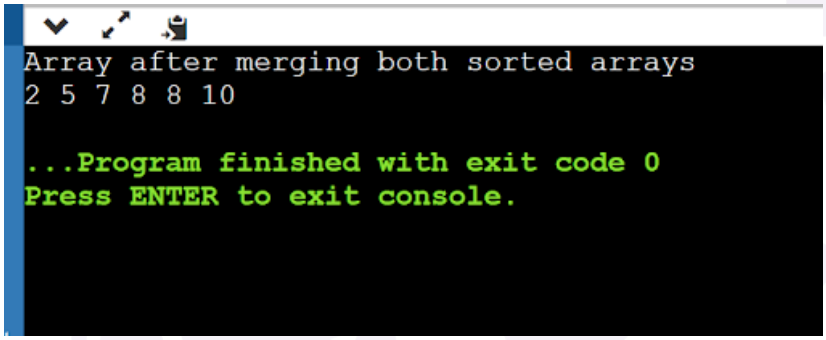
Examples:
Input: num1[] = { 5,8,10}, num2[] = {2,7, 8}
Output: num3[] = {2,5,7,8,8,10}

Simple approach is to take all the elements of num1 and num2 in num3. Then simply sort the num3 but this is not the efficient approach .

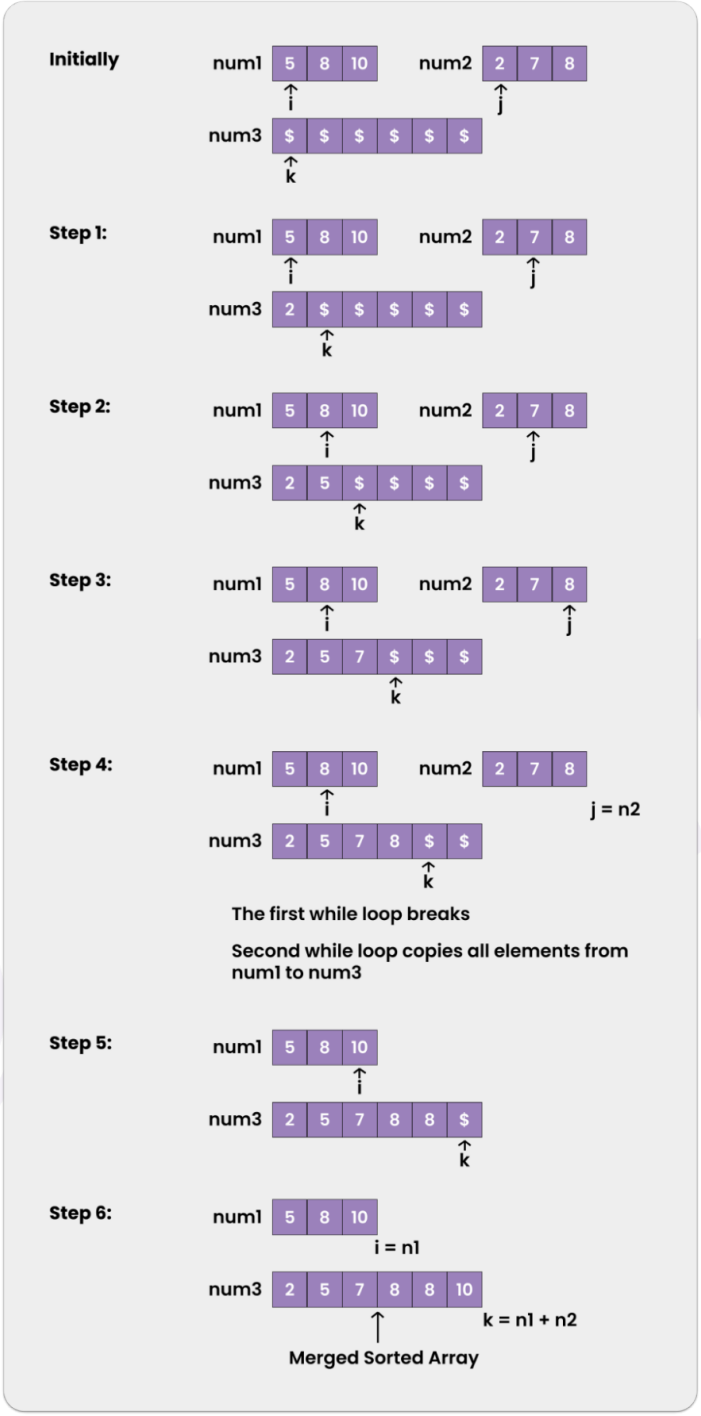
Efficient Approach is to use Merge function of Merge sort algorithm.

1. Create an arraynum3[] of size n1 + n2.
2. Simultaneously traverse num1[] and num2[].
   - Pick smaller of current elements in num1[] and arr2[], copy this smaller element to next position in num3[] and move ahead in num3[] and the array whose element is picked.
3. If there are remaining elements in num1[] or num2[], copy them also in num3[].

Code :
https://pastebin.com/8ww6UeXA



Below image is a dry run of the above approach:



Time Complexity : O(n1 + n2)
Auxiliary Space : O(n1 + n2)

Upcoming class Teasers:

- Binary search