1) Given a permuted array of length N of first N natural numbers, we need to tell the minimum number of swaps required in the sorted array of first N natural number to reach given permuted array where a number can be swapped with at most 2 positions left to it. If it is not possible to reach permuted array by above swap condition then print not possible.

```
Input : arr = [1, 2, 5, 3, 4]
Output : 2
We can reach to above-permuted array in total 2 swaps as shown below,
[1, 2, 3, 4, 5] -> [1, 2, 3, 5, 4] -> [1, 2, 5, 3, 4]

Input : arr[] = [5, 1, 2, 3, 4]
Output : Not Possible
It is not possible to reach above array just by swapping numbers 2
positions left to it.
```

2) Given two arrays A1[] and A2[], sort A1 in such a way that the relative order among the elements will be same as those are in A2. For the elements not present in A2, append them at last in sorted order.

Input: A1[] = {2, 1, 2, 5, 7, 1, 9, 3, 6, 8, 8} A2[] = {2, 1, 8, 3}
Output: A1[] = {2, 2, 1, 1, 8, 8, 3, 5, 6, 7, 9}
Input: A1[] = {4, 5, 1, 1, 3, 2} A2[] = {3, 1}
Output: A1[] = {3, 1, 1, 2, 4, 5}

3) Given an array of n strings. The task is to print the strings in sorted order. The approach should be such that no string should be copied to another string during the sorting process.

```
Input : {"ball", "pen", "apple", "kite"}
Output : apple ball kite pen
```

4) Given an array **arr[]** of size **'n'** and a positive integer **k**. Consider series of natural numbers and remove arr[0], arr[1], arr[2], …, arr[p] from it. Now the task is to find k-th smallest number in the remaining set of natural numbers. If no such number exists print "-1".

```
Input : arr[] = { 1 } and k = 1.
Output: 2
Natural numbers are {1, 2, 3, 4, .... }
After removing {1}, we get {2, 3, 4, ...}.
Now, K-th smallest element = 2.

Input : arr[] = {1, 3}, k = 4.
Output : 6
First 5 Natural number {1, 2, 3, 4, 5, 6,  .. }
After removing {1, 3}, we get {2, 4, 5, 6, ... }.
```