PRACTICAL – 13 CTSD

1 **Write a c program on Given an unsorted array arr[] of size N. Rotate the array to the left (counter-clockwise direction) by D steps, where D is a positive integer.**

```c
#include <stdio.h>
  int main()
{
   int arr[] = {1, 2, 3, 4, 5};
   int length = sizeof(arr)/sizeof(arr[0]);
   int n = 3;
   printf("Original array: \n");
   for (int i = 0; i < length; i++) {
       printf("%d ", arr[i]);
   }
     for(int i = 0; i < n; i++){
       int j, first;
       first = arr[0];
       for(j = 0; j < length-1; j++){
           arr[j] = arr[j+1];
       }
       arr[j] = first;
   }
     printf("\n");
     printf("Array after left rotation: \n");
   for(int i = 0; i < length; i++){
       printf("%d ", arr[i]);
   }
   return 0;
}
```

OUTPUT:

```
Original array:
1 2 3 4 5
Array after left rotation:
4 5 1 2 3
```

2 **Write a c Program on given two sorted arrays arr1 and arr2 of size N and M respectively and an element K. The task is to find the element that would be at the k□th position of the final sorted array.Explanation:**
Input :
Array 1 - 1 4 2 3 5
Array 2 - 7 8 6
k = 6
Because The final sorted array would be -1, 2, 3, 4, 5, 6, 7, 8, The 5th element of this array is 6.

```
#include <iostream>
using namespace std;

int kth(int arr1[], int arr2[], int m, int n, int k)
{
   int sorted1[m + n];
   int i = 0, j = 0, d = 0;
   while (i < m && j < n)
   {
      if (arr1[i] < arr2[j])
         sorted1[d++] = arr1[i++];
      else
         sorted1[d++] = arr2[j++];
   }
   while (i < m)
      sorted1[d++] = arr1[i++];
   while (j < n)
      sorted1[d++] = arr2[j++];
   return sorted1[k - 1];
}
int main()
{
   int arr1[5] = {2, 3, 6, 7, 9};
   int arr2[4] = {1, 4, 8, 10};
   int k = 5;
```

```
    cout << kth(arr1, arr2, 5, 4, k);
    return 0;
}
```

OUTPUT:

```
6
```