Program 1: Given an array of integers, write an algorithm and a program to left rotate the array by specific number of elements.

```
#include <stdio.h>
int main()
{
  int arr[50];
  int n,k;
  scanf("%d",&n);
  scanf("%d",&k);
  for(int i=0;i<n;i++)
  {
     scanf("%d",&arr[i]);
  }
  while(k--)
  {
  for(int i=0;i<n;i++)
  {
     printf("%d ",arr[i]);
  }
  printf("\n");
  int temp=arr[0];
  for(int i=1;i<n;i++)
  {
     arr[i-1]=arr[i];
  }
  arr[n-1]=temp;

  }
   for(int i=0;i<n;i++)
  {
     printf("%d ",arr[i]);
  }
}
```

Program 1.2

```
#include <stdio.h>

/* Function to reverse a portion of the array */
void reverse(int arr[], int start, int end) {
    while (start < end) {
        int temp = arr[start];
        arr[start] = arr[end];
        arr[end] = temp;
        start++;
        end--;
    }
}

/* Function to rotate an array by d elements to the left */
void rotateArray(int arr[], int d, int n) {
    d %= n;
    // Reverse the first d elements
    reverse(arr, 0, d - 1);
    // Reverse the remaining n-d elements
    reverse(arr, d, n - 1);
    // Reverse the entire array
    reverse(arr, 0, n - 1);
}
int main() {
    int arr[] = { 1, 2, 3, 4, 5, 6 };
    int n = sizeof(arr) / sizeof(arr[0]);
    int d = 2;
    rotateArray(arr, d, n);
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Program 2: Given an unsorted array of integers and two numbers a and b. Design an algorithm and write a program to find minimum distance between a and b in this array. Minimum distance between any two numbers a and b present in array is the minimum difference between their indices.

```
#include <stdio.h>
#include <limits.h>  // For INT_MAX and INT_MIN
#include<stdlib.h>
int findMinDistance(int arr[], int n, int a, int b) {
    int last_pos_a = -1, last_pos_b = -1;
    int min_distance = INT_MAX;

    for (int i = 0; i < n; i++) {
        if (arr[i] == a) {
            last_pos_a = i;
        }
        if (arr[i] == b) {
            last_pos_b = i;
        }

        // If both a and b have been found at least once
        if (last_pos_a != -1 && last_pos_b != -1) {
            int distance = abs(last_pos_a - last_pos_b);
            if (distance < min_distance) {
                min_distance = distance;
            }
        }
    }

    // If min_distance was updated, return it; otherwise, return -1
    if (min_distance == INT_MAX) {
        return -1;  // a and b were not both found in the array
    }

    return min_distance;
}

int main() {
    int arr[] = {3, 5, 4, 2, 6, 5, 6, 7, 3, 2, 7, 2, 6, 3, 1, 4, 2};
    int n = sizeof(arr) / sizeof(arr[0]);
    int a = 3;
    int b = 6;

    int result = findMinDistance(arr, n, a, b);
    if (result != -1) {
        printf("Minimum distance between %d and %d is: %d\n", a, b, result);
    } else {
        printf("One or both of the numbers %d and %d are not in the array.\n", a, b);
    }

    return 0;
}
```

Program 3: Given an array of nonnegative integers, where all numbers occur even number of times except one number which occurs odd number of times. Write an algorithm and a program to find this number. (Time complexity = O(n))

```
#include <stdio.h>
#include <limits.h>  // For INT_MAX and INT_MIN
#include<stdlib.h>
int main() {
    int nums[] = {1,1,2,3,3,4,4,8,8};
    int n=sizeof(nums)/sizeof(nums[0]);
    int ele=nums[0];
      for(int i=1;i<n;i++)
      {
          ele=ele^nums[i];
      }
    printf("%d",ele);

}
```