Previous

↳ 1. Max. Sum Subarray

4 -12 4 1 -2 3

ans =

c =

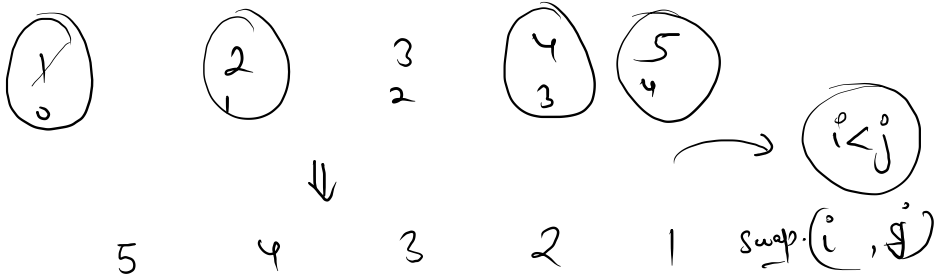
5

csum < 0
↳ start new sub

csum ≥ 0
↳ contribute to previous sub.

2. Max. Product Sub array.

3. reverse an array.



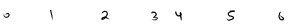
Rotate Right

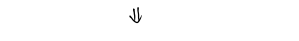
Given an array, rotate the array to the right by k steps, where k is non-negative.

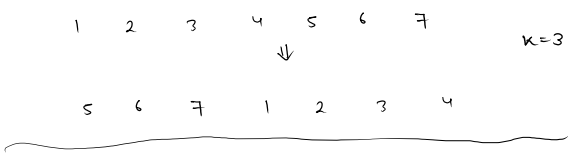
Sample Input 0

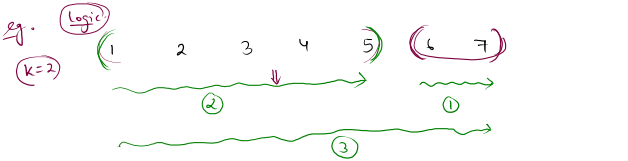
Sample Output 0

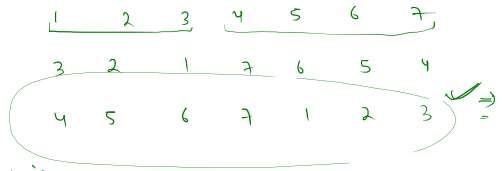
6 7 1 2 3 4 5

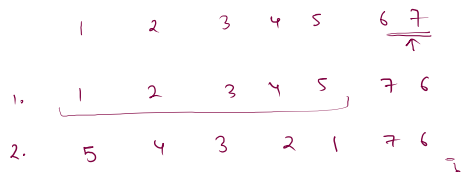


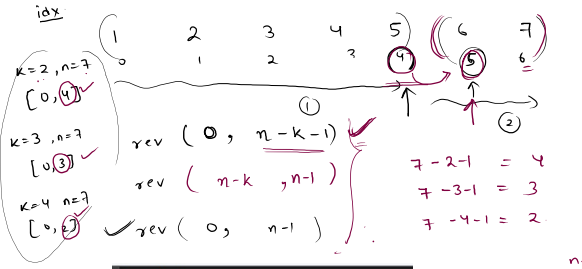












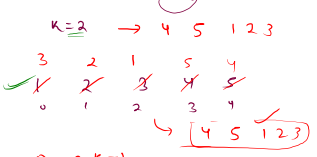

```java
import java.io.*;
import java.util.*;

public class Solution {
    public static void reverse(int [] arr, int i , int j){
        while(i < j){
            int t = arr[i];
            arr[i] = arr[j];
            arr[j] = t;
            i++;
            j--;
        }
    }

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] arr = new int[n];
        for(int i = 0; i< n; i++){
            arr[i] = scn.nextInt();
        }
        int k = scn.nextInt();

        reverse(arr, 0, n-k-1);
        reverse(arr, n-k, n-1);
        reverse(arr, 0 , n-1);

        for(int i = 0; i < arr.length; i++){
            System.out.print(arr[i] + " ");
        }
    }
}
```



Zeroes and Ones

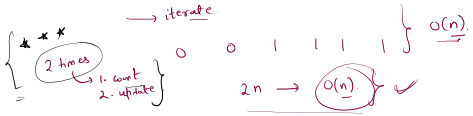
Given an array having n elements consisting of only 0s and 1s. Sort the array in O(N) time complexity.

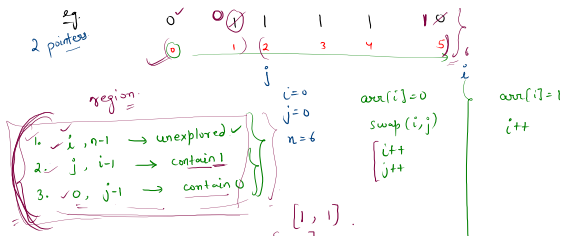
Sample Input 0

```
6
0 1 1 1 1 0
```

Sample Output 0

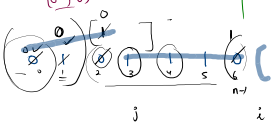
```
0 0 1 1 1 1
```






```
int i = 0;
int j = 0;
while(i < n){
    if( arr[i] == 0){
        swap(arr, i, j);
        i++;
        j++;
    }
    else{
        i++;
    }
}
for( i = 0; i < arr.length; i++){
    System.out.print(arr[i] + " ");
}
```

eg.




```
import java.io.*;
import java.util.*;

public class Solution {
    public static void swap(int [] arr, int i, int j){
        int tmp = arr[i];
        arr[i] = arr[j];
        arr[j] = tmp;
    }


    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] arr = new int[n];
        for(int i = 0; i < n; i++){
            arr[i] = scn.nextInt();
        }
        int i = 0;
        int j = 0;
        while(i < n){
            if( arr[i] == 0){
                swap(arr, i, j);
                i++;
                j++;
            }
            else{
                i++;
            }
        }
        for( i = 0; i < arr.length; i++){
            System.out.print(arr[i] + " ");
        }
    }
}
```

Sort 0 1 2 (Dutch National Flag)

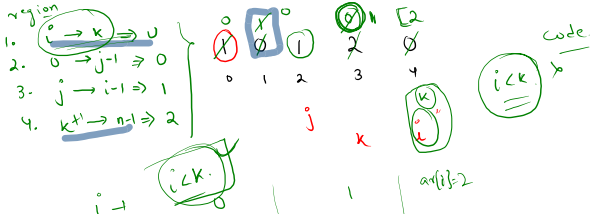
You have given an array **arr** with 0's , 1's and 2's in it. You need to sort the array but the condition is you are not allowed to use the any sorting algorithum.

Sample Input 0

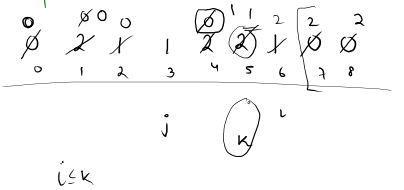
```
5
1 0 1 2 0
```

Sample Output 0

```
0 0 1 1 2
```




```
int i = 0;
int j = 0;
int k = n-1;
while(i <= k){
    if( arr[i] == 0){
        swap(arr, i, j);
        i++;
        j++;
    }
    else if(arr[i] == 1){
        i++;
    }
    else{
        swap(arr, i, k);
        k--;
    }
}
```



0 0 0 1 1 1 2 2 2

[i → k] → 0
0 → j-1 → 0
j → i-1 → 1
k → n-1 → 2

```
import java.io.*;
import java.util.*;

public class Solution {
    public static void swap(int [] arr, int i, int j){
        int tmp = arr[i];
        arr[i] = arr[j];
        arr[j] = tmp;
    }


    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] arr = new int[n];
        for(int i = 0; i< n; i++){
            arr[i] = scn.nextInt();
        }
        int i = 0;
        int j = 0;
        int k = n-1;
        while(i <= k){
            if( arr[i] == 0){
                swap(arr, i, j);
                i++;
                j++;
            }
            else if(arr[i] == 1){
                i++;
            }
            else{
                swap(arr, i, k);
                k--;
            }
        }
        for( i = 0; i < arr.length; i++){
            System.out.print(arr[i] + " ");
        }

    }
}
```