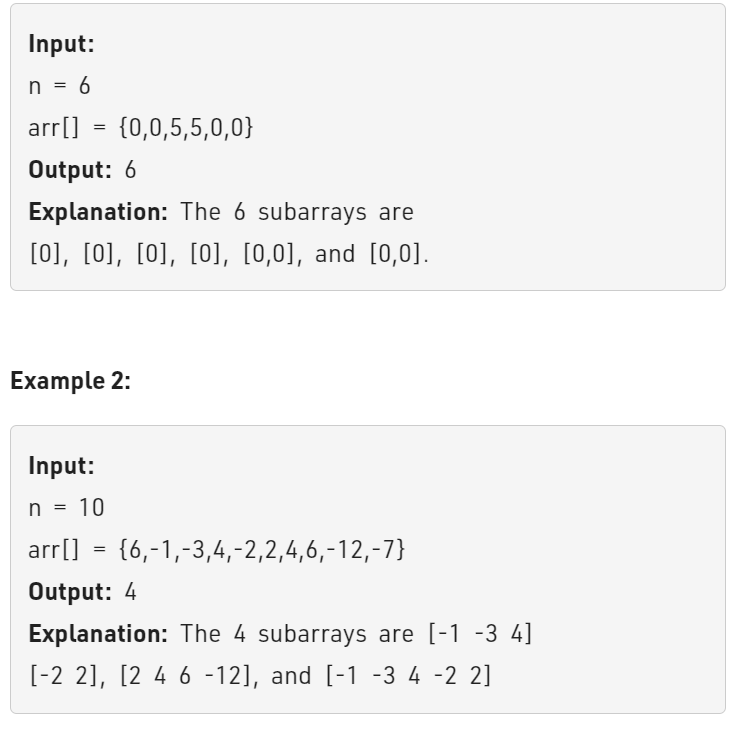
1.int k=0;

    int n=6;

    int a[n]={0,0,5,5,0,0};

    for(int i=0;i<n;i++){

        int curr\_counter=0;

        for(int j=i;j<n;j++){

            curr\_counter+=a[j];

            cout<<curr\_counter<<endl;

            if(curr\_counter==0) k++;

            cout<<k<<endl;

        }

    }

    cout<<k;

2. using hash maps with toc o(n);

unordered\_map<int,int> map;

    long long int pre\_sum=0;

    long long int k=0;

    for(int i=0;i<n;i++){

        pre\_sum += a[i];

        if(pre\_sum==0) k++;

        if(map[pre\_sum]) k+=map[pre\_sum] ;

        map[pre\_sum]++;

    }

    return k;

2.

#include <iostream>

using namespace std;

// Function to Reverse 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 k elements to right

void Rotateeletoright(int arr[], int n, int k)

{

// Reverse first n-k elements

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

// Reverse last k elements

Reverse(arr, n - k, n - 1);

// Reverse whole array

Reverse(arr, 0, n - 1);

}

int main()

{

int arr[] = {1, 2, 3, 4, 5, 6, 7};

int n = 7;

int k = 2;

Rotateeletoright(arr, n, k);

cout << "After Rotating the k elements to right ";

for (int i = 0; i < n; i++)

cout << arr[i] << " ";

cout << endl;

return 0;

}

**Output:**

After Rotating the k elements to right 6 7 1 2 3 4 5

**Time Complexity –**O(N) where N is the number of elements in an array

**Space Complexity –**O(1) since no extra space is required