**Max Circular Subarray Sum**

**Hard**Accuracy: 45.16% Submissions: 38981 Points: 8

You are given an array **arr[ ]** of **N** integers arranged in a **circular** fashion. Your task is to find the **maximum contiguous subarray sum**.

**Example 1:**

**Input:**

N = 7

arr[] = {8,-8,9,-9,10,-11,12}

**Output:**

22

**Explanation:**

Starting from the last element

of the array, i.e, 12, and

moving in a circular fashion, we

have max subarray as 12, 8, -8, 9,

-9, 10, which gives maximum sum

as 22.

**Example 2:**

**Input:**

N = 8

arr[] = {10,-3,-4,7,6,5,-4,-1}

**Output:**

23

**Explanation:** Sum of the circular

subarray with maximum sum is 23

**Your Task:**  
The task is to complete the function **circularSubarraySum**() which returns a sum of the circular subarray with maximum sum.

**Expected Time Complexity:**O(N).  
**Expected Auxiliary Space:**O(1).

**Constraints:**  
1 <= N <= 106  
-106<= Arr[i] <= 106

class Solution{

    public:

    // arr: input array

    // num: size of array

    //Function to find maximum circular subarray sum.

    int kadane(int arr[], int n){

        // Your code here

        int max=INT\_MIN, curr=0;

        for (int i=0; i<n; i++) {

            if (curr>=0) curr+=arr[i];

            else  curr=arr[i];

            if (curr>max) max=curr;

        }

        return max;

    }

    int circularSubarraySum(int arr[], int n){

        // your code here

        int sum=0;

        int sum1=kadane(arr, n);

        for (int i=0; i<n; i++) {

            sum+=arr[i];

            arr[i]=-arr[i];

        }

        int sum2=kadane(arr, n);

        if (sum1<0) return sum1;

        else return max(sum1, sum+sum2);

    }

};