**Largest subarray with 0 sum**

Given an array having both positive and negative integers. The task is to compute the length of the largest subarray with sum 0.

**Examples:**

**Input:** arr[] = {15,-2,2,-8,1,7,10,23}, n = 8

**Output:** 5

**Input:** arr[] = {2,10,4}, n = 3

**Output:** 0

**Input:** arr[] = {1, 0, -4, 3, 1, 0}, n = 6

**Output:** 5

Step 1 : Initialize a map with {0 , 1}

Step 2: Suppose if there is sum 10 and after some times again it comes while summing up the value , then there exist a subarray with value 0 between the previous 10 and current .

Step 3: This is the logic behind the scene .But when the sum is zero , then there is no need to check , it will be the largest sub array so far.

class GfG

{

int maxLen(int arr[], int n)

{

// Your code here

Map<Integer , Integer> map = new HashMap<>();

int sum = 0;

int max = 0;

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

sum += arr[i];

if(sum == 0){

max = i + 1;

}

else if (!map.containsKey(sum)){

map.put(sum , i);

}

else{

max = Math.max(max , i - map.get(sum));

}

}

return max;

}

}