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.

Example 1:

```
Input: N = 8
A[] = {15,-2,2,-8,1,7,10,23}
Output: 5 Explanation: The largest subarray with
sum 0 will be -2 2 -8 1 7.
```

Your Task:
You just have to complete the function **maxLen()** which takes two arguments an array **A** and **n,** where n is the size of the array A and returns the length of the largest subarray with 0 sum.

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

```
class Solution:
    def maxLen(self, n, arr):
        #Code here
        freq = {}
        freq[0] = -1
        prefixSum = 0
        maxL = 0
        for i, ele in enumerate(arr):
            prefixSum = prefixSum + ele
            if prefixSum in freq:
                maxL = max(maxL, i - freq[prefixSum])
            else:
                freq[prefixSum] = i
        return maxL
```