Longest subarray with sum divisible by K

Given an array containing **N** integers and a positive integer **K**, find the length of the longest sub array with sum of the elements divisible by the given value **K**.

Example 1:

```
Input: A[] = {2, 7, 6, 1, 4, 5}
K = 3
Output: 4
Explanation:The subarray is {7, 6, 1, 4}
with sum 18, which is divisible by 3.
```

Example 2:

```
Input: A[] = {-2, 2, -5, 12, -11, -1, 7}
K = 3
Output: 5
Explanation: The subarray is {2,-5,12,-11,-1} with
sum -3, which is divisible by 3.
```

Your Task:
The input is already taken care of by the driver code. You only need to complete the function **longSubarrWthSumDivByK()** that takes an array **(arr)**, sizeOfArray **(n)**, positive integer **K**, and return the length of the longest subarray which has sum divisible by **K**. The driver code takes care of the printing.
Expected Time Complexity: O(N).
Expected Auxiliary Space: O(N)

```
class Solution:
    def longSubarrWthSumDivByK (self,nums,  n, k) :
        #Complete the function
        freq = {}
        prefix = 0
        freq[0] = -1
        count = 0
        for i,ele in enumerate(nums):
            prefix+=ele
            rem = prefix%k
            if rem<0:
                rem = rem+k
```

```
                #To make sure we deal with negative remainders.
                Like -3 is remainder then -3+7 is 4 and another
                occurences of 4 will result in an arr but if we
                look for -3 again that would mean we are wanting the sub-
array whose sum is 0.
            if rem in freq:
                count = max(count,i-freq[rem])
            else:
                freq[rem] = i

        return count
```