Problem Link

https://practice.geeksforgeeks.org/problems/subarray-with-given-sum-1587115621/1

Given an unsorted array **A** of size **N** that contains only non-negative integers, find a continuous sub-array which adds to a given number **S**.

In case of multiple subarrays, return the subarray which comes first on moving from left to right.

Example 1:

```
Input:
N = 5, S = 12
A[] = {1,2,3,7,5}
Output: 2 4
Explanation: The sum of elements
from 2nd position to 4th position
is 12.
```

Example 2:

```
Input:
N = 10, S = 15
A[] = {1,2,3,4,5,6,7,8,9,10}
Output: 1 5
Explanation: The sum of elements
from 1st position to 5th position
is 15.
```

Your Task:

You don't need to read input or print anything. The task is to complete the function **subarraySum**() which takes arr, N and S as input parameters and returns an **arraylist** containing the **starting** and **ending** positions of the first such occurring subarray from the left where sum equals to S. The two indexes in the array should be according to 1-based indexing. If no such subarray is found, return an array consisting only one element that is -1.

Expected Time Complexity: O(N)

Expected Auxiliary Space: O(1)

Constraints:

$1 <= N <= 10^5$

$1 <= A_i <= 10^9$

Solution

```
def sumarr(startpoint,endpoint):
    sum=0
    elements=[]
    for i in range(startpoint,endpoint):
        if sum < S:
            sum += A[i]
            elements.append(A[i])
        elif sum > S:
            elements=[]
            startpoint += 1
            sumarr(startpoint,endpoint)
        elif sum == S:
            print(startpoint+1,i)
            break


if __name__=="__main__":
    N=5
    S=12
    A=[1,2,3,7,5]
    sumarr(0,len(A))
```