ELogo STUDENT REPORT  Name 3CD 3A 3HR13CD 3A	38273CV
	.,5
	32h
Cho, Ky, Cho, Ky, Cho, Ky,	ak 3BK
DETAILS SANT COST SERVICE COST SANT COST SANT COST SANT COST COST COST COST COST COST COST COS	3CD0'3
Name of Strain Control of the Contro	38 R23 CD 3 N 3 RP 3 CD C
K AMBIKA	
3BR23CD034	30 3 A 3 B R 13 C D 3 A 3 B R 13 C D C
3BR23CD034	
EXPERIMENTS AS	30000
EXPERIMENT  Title  EQUILIBRIUM  Description  A 20103 A 3441 3C103 A 3441 A 3	3BR22 50C
EOUILIBRIUM CO	33K 3CV
38500 34503 35 35 35 35 35 35 35 35 35 35 35 35 35	03h
EQUIEIBRIUM  SCY  Description  A35C1  A35C1  A36C1	R13CV
You are given an array A of N integers. An equilibrium position is a position where the sum of all integers on its left is equal t	to the sum
of all integers on its right in the array A. Print the index of the equilibrium position.	
of all integers on its right in the array A. Print the index of the equilibrium position.  Note: For any given array there is only a single equilibrium position, if no equilibrium position is found then print "NOT FOUND quotes."	D" without
The array is 1 indexed	34
No array to 1 indexed.	
Input Format:	BRISCOC
	382
The input consists of two lines:  The first line contains an integer denoting N.  The second line contains N appear constraint integers denoting the elements of the error A.	4
The second line contains N space-separated integers denoting the elements of the array A.	5CD(3 <sup>A</sup> 3 <sup>c</sup>
	500
Input will be read from the STDIN by the candidate  Output Format:	٥
Print the index of the equilibrium position. If no index is found, print "NOT FOUND"	3 <sup>A</sup> 38R2 <sup>2</sup> .
	3"
Sample Input  5	9
24733	F. G. Char.
Sample Output  3	
	GB-1255V
Source Code: 3CDV 3APA STORM AND STO	A Charles
38th COC34 State Colored State	Carried Sept.
Source Code: 35 CD 3 A 36 Pt 25 CD 3 A 36 Pt 2	388 Jan 1
A Service of the serv	
April 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 - 1992 -	3803
CD, Why	350
~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~	360

```
def find_equilibrium_position(N, A):
       total_sum = sum(A)
       left_sum = 0
       for i in range(N):
           right_sum = total_sum - left_sum - A[i]
           if left_sum == right_sum:
               return i + 1
           left_sum += A[i]
       return "NOT FOUND"
   # Input reading
   N = int(input())
   A = list(map(int, input().split()))
   result = find_equilibrium_position(N, A)
   print(result)
RESULT
 5 / 5 Test Cases Passed | 100 %
```