Circular tour

Suppose there is a circle. There are **N** petrol pumps on that circle. You will be given two sets of data.
1. The amount of petrol that every petrol pump has.
2. Distance from that petrol pump to the next petrol pump.
Find a starting point where the truck can start to get through the complete circle without exhausting its petrol in between.
Note : Assume for 1 litre petrol, the truck can go 1 unit of distance.

Example 1:

Input:

N = 4

Petrol = 4 6 7 4

Distance = 6 5 3 5

Output: 1

Explanation: There are 4 petrol pumps with

amount of petrol and distance to next

petrol pump value pairs as {4, 6}, {6, 5},

{7, 3} and {4, 5}. The first point from

where truck can make a circular tour is

2nd petrol pump. Output in this case is 1

(index of 2nd petrol pump).

Your Task:
Your task is to complete the function **tour**() which takes the required data as inputs and returns an integer denoting a point from where a truck will be able to complete the circle (The truck will stop at each petrol pump and it has infinite capacity). If there exists multiple such starting points, then the function must return the first one out of those. (return -1 otherwise)

Expected Time Complexity: O(N)
Expected Auxiliary Space : O(1)

Constraints:

$2 \leq N \leq 10000$

$1 \leq petrol, distance \leq 1000$

```
/*
The structure of petrolPump is
struct petrolPump
{
    int petrol;
    int distance;
};*/

/*You are required to complete this method*/
class Solution{
  public:

    //Function to find starting point where the truck can
start to get through
    //the complete circle without exhausting its petrol in
between.
    int tour(petrolPump p[],int n)
    {
       //Your code here
       int start = 0;
       int total_petrol = 0;
       int total_distance = 0;
       int curr_balance = 0;

       for(int i=0;i<n;i++){
           total_petrol+=p[i].petrol;
           total_distance+=p[i].distance;
           curr_balance +=(p[i].petrol-p[i].distance);

           if(curr_balance<0){
               start = i+1;
               curr_balance = 0;

           }
       }

       if(total_petrol >= total_distance)
           return start;
       return -1;
    }
};
```