Equilibrium Index

Sum of all elements before ith idx = Sum of all elements after ith idx

left Sum | Pivot index / Equilibrium index

Ex : arr[ ] = (−2 , 0 , 1 , 2 , 3 , 0 , 1)
idx: 0 1 2 3 4 5 6

rightSum

As, if you take idx = 4 ↴

Sum of elements from idx (0 → 3) = 1

Sum of elements from idx (5 → 6) = 1

∴ idx = 4 → is an equilibrium index / pivot index

Algorithm

① Calculate prefix sum array (PS[ ])

② Iterate through (1 → (n-2))

↓ ↳ why not including first and last Elements ?

↓ those will be handled in edge cases

* Check { if PS(i-1) == PS(n-1) − PS(i) }

PS(i-1) → left Sum

PS(n-1) − PS(i) → rightSum

⇒ if they are equal, we got one equilibrium index

Edge Cases

① if (i == n-1) → There will be no right Sum to check with left Sum

that's why it's need to handle seperately

So, Check { PS(n-2) == 0 } → no right sum, so 0

PS(n-2) → left Sum

→ if true, we got one Equilibrium index

Case:2

if(i==0)

↳ Similarly, there will be no left Sum to check with right Sum

So, need to handle it seperately

check :

{ PS(n-1) − PS(0) == 0 } ⟶ if true, we got one Equilibrium index

PS(n-1) → right sum

PS(0) → leftsum