GeeksforGeeks
A computer science portal for geeks

GeeksQuiz

Login

Home Algorithms DS GATE Interview Corner Q&A C C++ Java Books Contribute Ask a Q About
Array Bit Magic C/C++ Articles GFacts Linked List MCQ Misc Output String Tree Graph

Equilibrium index of an array

Equilibrium index of an array is an index such that the sum of elements at lower indexes is equal to the sum of elements at higher indexes. For example, in an arrya A:

A[0] = -7, A[1] = 1, A[2] = 5, A[3] = 2, A[4] = -4, A[5] = 3, A[6]=0

3 is an equilibrium index, because:
A[0] + A[1] + A[2] = A[4] + A[5] + A[6]

6 is also an equilibrium index, because sum of zero elements is zero, i.e., A[0] + A[1] + A[2] + A[3] + A[4] + A[5]=0

7 is not an equilibrium index, because it is not a valid index of array A.

Write a function *int equilibrium(int[] arr, int n)*; that given a sequence arr[] of size n, returns an equilibrium index (if any) or -1 if no equilibrium indexes exist.

Method 1 (Simple but inefficient)
Use two loops. Outer loop iterates through all the element and inner loop finds out whether the current index picked by the outer loop is equilibrium index or not. Time complexity of this solution is O(n^2).

```
#include <stdio.h>

int equilibrium(int arr[], int n)
{
  int i, j;
  int leftsum, rightsum;

  /* Check for indexes one by one until an equilibrium
    index is found */
```

Google™ Custom Search

GeeksforGeeks

53,522 people like GeeksforGeeks.

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
  for ( i = 0; i < n; ++i)
  {
    leftsum = 0;  // initialize left sum for current index i
    rightsum = 0; // initialize right sum for current index i

    /* get left sum */
    for ( j = 0; j < i; j++)
      leftsum  += arr[j];

    /* get right sum */
    for( j = i+1; j < n; j++)
      rightsum += arr[j];

    /* if leftsum and rightsum are same, then we are done */
    if (leftsum == rightsum)
      return i;
    }

  /* return -1 if no equilibrium index is found */
  return -1;
}

int main()
{
  int arr[] = {-7, 1, 5, 2, -4, 3, 0};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printf("%d\n", equilibrium(arr, arr_size));

  getchar();
  return 0;
}
```

Time Complexity: O(n^2)

Method 2 (Tricky and Efficient)
The idea is to get total sum of array first. Then Iterate through the array and keep updating the left sum which is initialized as zero. In the loop, we can get right sum by subtracting the elements one by one. Thanks to Sambasiva for suggesting this solution and providing code for this.

```
1) Initialize leftsum  as 0
2) Get the total sum of the array as sum
3) Iterate through the array and for each index i, do following.
    a) Update sum to get the right sum
```

Popular Posts
All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

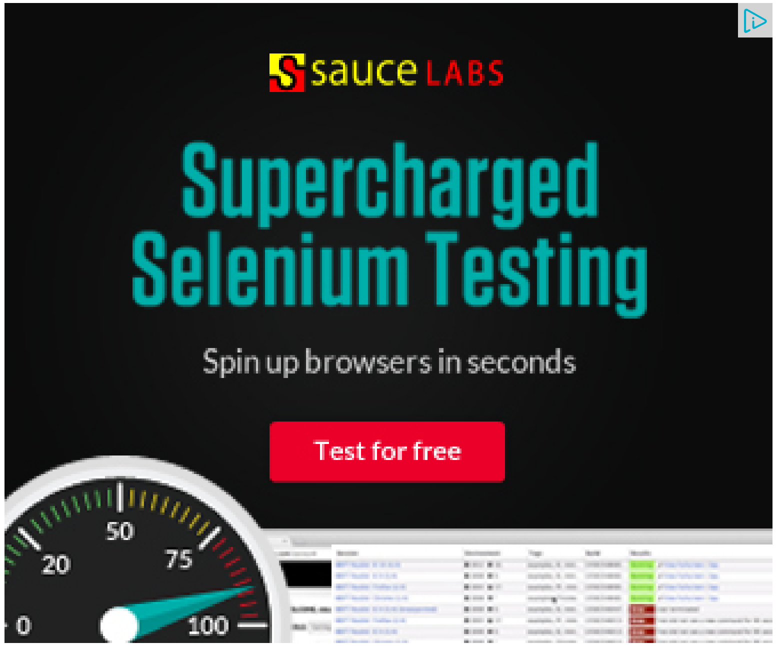
```
    a)  Update sum to get the right sum.
           sum = sum - arr[i]
       // sum is now right sum
    b) If leftsum is equal to sum, then return current index.
    c) leftsum = leftsum + arr[i] // update leftsum for next iteration.
4) return -1 // If we come out of loop without returning then
             // there is no equilibrium index
```

```c
#include <stdio.h>

int equilibrium(int arr[], int n)
{
   int sum = 0;      // initialize sum of whole array
   int leftsum = 0; // initialize leftsum
   int i;

   /* Find sum of the whole array */
   for (i = 0; i < n; ++i)
        sum += arr[i];

   for( i = 0; i < n; ++i)
   {
      sum -= arr[i]; // sum is now right sum for index i

      if(leftsum == sum)
        return i;

      leftsum += arr[i];
   }

    /* If no equilibrium index found, then return 0 */
    return -1;
}

int main()
{
  int arr[] = {-7, 1, 5, 2, -4, 3, 0};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printf("First equilibrium index is %d\n", equilibrium(arr, arr_size)

  getchar();
  return 0;
}
```

Time Complexity: O(n)

As pointed out by Sameer, we can remove the return statement and add a print statement to print all equilibrium indexes instead of returning only one.

Please write comments if you find the above codes/algorithms incorrect, or find better ways to solve the same problem.



Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 24 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 28 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 53 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 54 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

- Divide and Conquer | Set 5 (Strassen’s Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

2 Tweet 0 2

Writing code in comment? Please use **ideone.com** and share the link here.