

01.

01. Sum of unique elements in Array**Problem Statement**

You are given a function,
`int SumUniqueElements(int* arr, int length);`

The function accepts an integer array 'arr' of size 'length' as input. Implement the function such that it returns the sum of unique (non-duplicate) elements in the array.

Note:

- Return 0 if no unique element is found.
- Computed values lie within integer range.

Example:**Input:**

2 5 4 7 9 2 4 8

Output:

29

Explanation:

Elements 2 and 4 occur twice in the input array and are not included in the sum. Hence, the sum of unique elements in the array is 29.

The custom input format for the above case:

8

2 5 4 7 9 2 4 8

(The first line represents the size of the array, the second line represents the elements of the array)

Sample input

10 12 22 10 12 98

Sample Output

120

The custom input format for the above case:

6

02. **Example:**
Input:
2 5 4 7 9 2 4 8
03. **Output:**
29

Explanation:
Elements 2 and 4 occur twice in the input array and are not included in the sum. Hence, the sum of unique elements in the array is 29.
The custom input format for the above case:
8
2 5 4 7 9 2 4 8

(The first line represents the size of the array, the second line represents the elements of the array)
Sample input
10 12 22 10 12 98

Sample Output
120
The custom input format for the above case:
6
10 12 22 10 12 98

(The first line represents the size of the array, the second line represents the elements of the array)
Instructions :

- Create a solution of $O(n)$ complexity to avoid deduction of marks.

Now let's start coding :

Language: C (Gcc 7.5) ✓



Day Mode

03. Trees identical?

Problem Statement

A binary tree is represented by the following structure:

```
struct TreeNode
{
    int data;
    struct TreeNode* left;
    struct TreeNode* right;
};
```

Two trees are said to be structurally identical if they are made of nodes with the same values arranged in the same way. Implement the following function:

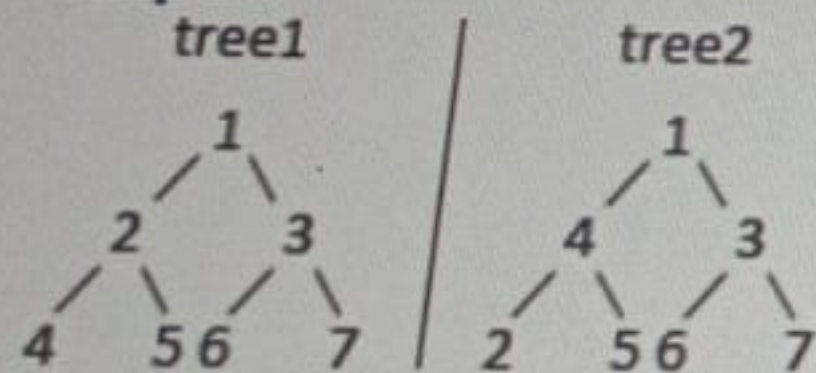
```
int IsIdentical(struct TreeNode* tree1, struct TreeNode* tree2);
```

The function accepts two binary trees 'tree1' and 'tree2'. If 'tree1' and 'tree2' are identical, return 1, else return 0.

Note: Empty trees are identical.

Example:

Input:



Output:

0

The custom input format for the above case:

2 1 2 3 4 5 6 7

2 1 4 3 2 5 6 7

The custom input format for the above case:
 2 1 2 3 4 5 6 7
 2 1 4 3 2 5 6 7

(In the first line, the first integer represents the height of the 'tree1' and the rest of the integers represent the level order traversal of the 'tree1', in the second line, the first integer represents the height of the 'tree2' and the rest of the integers represent the level order traversal of the 'tree2')

Sample input

tree1:

2

1 3

tree2:

2

1 3

Sample Output

1

The custom input format for the above case:

1 2 1 3

1 2 1 3


(In the first line, the first integer represents the height of the 'tree1' and the rest of the integers represent the level order traversal of the 'tree1', in the second line, the first integer represents the height of the 'tree2' and the rest of the integers represent the level order traversal of the 'tree2')


Instructions :

- This is a template based question. DO NOT write the "main" function


- "Save and Test" only checks for basic test cases, more rigorous cases will be used to judge your code while scoring.
- Additional score will be given for writing optimized code both in terms of memory and execution time.

Now let's start coding :

Language: Java 1.8 

 Read-only code below ...

```
1 class TreeNode
2 {
3     public int data;
4     public TreeNode left;
5     public TreeNode right;
6 }
7
```

 Write your code below ...

```
8 static int IsIdentical(TreeNode tree1, TreeNode tree2) throws java.lang.Exception
9 {
10     /* Write your code here. */
11 }
12
13
14
```


02.

19
3
1 5 17

(The first line represents 'x', the second line represents 'len', the third line represents the elements of the array 'arr')

03.

Sample input

x: 27

arr: {2, 5, 8, 18, 19}

Sample Output

1

The custom input format for the above case:

27

5

2 5 8 18 19

(The first line represents 'x', the second line represents 'len', the third line represents the elements of the array 'arr')

Instructions :

- This is a template based question, DO NOT write the "main" function.
- Your code is judged by an automated system, do not write any additional welcome/greeting messages.
- "Save and Test" only checks for basic test cases, more rigorous cases will be used to judge your code while scoring.
- Additional score will be given for writing optimized code both in terms of memory and execution time.

Now let's start coding :

Language: C (Gcc 7.5) ▾

> Read-only code below ...

```
1 int IsSumPossible(int* arr, int len, int x);
```

03.

Implement the following function:
`int IsSumPossible(int* arr, int len, int x);`

The function accepts an integer array 'arr' of length 'len' and a number 'x'. Return 1 if a subarray exists in 'arr' for which sum of all its elements is equal to 'x'. Return 0 otherwise.

Assumption: All the elements in the array are positive integers, greater than 0.

Note: Subarray refers to any combination of elements of an array.

Example

Input:

x: 19

arr: {1, 5, 17}

Output:

0

Explanation:

Following are the possible subarrays:

Subarray	Sum
{1}	1
{5}	5
{17}	17
{1, 5}	6
{1, 17}	18
{5, 17}	22
{1, 5, 17}	23

There is no subarray of 'arr' having sum 19. Thus, output is 0.

The custom input format for the above case:

19