Description | Editorial | Solutions (1.4K) | Submissions

1318. Minimum Flips to Make a OR b Equal to c

Hint

Medium | 1.2K | 70

Companies

Given 3 positives numbers `a`, `b` and `c`. Return the minimum flips required in some bits of `a` and `b` to make (`a` OR `b` == `c`). (bitwise OR operation).
Flip operation consists of change **any** single bit 1 to 0 or change the bit 0 to 1 in their binary representation.

Example 1:


```
Input: a = 2, b = 6, c = 5
Output: 3
Explanation: After flips a = 1 , b = 4 , c = 5 such that (a OR b == c)
```

Example 2:

```
Input: a = 4, b = 2, c = 7
Output: 1
```

Example 3:

```
Input: a = 1, b = 2, c = 3
Output: 0
```

Java | Auto

```java
class Solution {
    public int minFlips(int a, int b, int c) {
        int flips = 0;
        while (a > 0 || b > 0 || c > 0) {
            int bitA = a & 1;
            int bitB = b & 1;
            int bitC = c & 1;

            if (bitC == 0) {
                flips += (bitA + bitB);
            } else {
                if (bitA == 0 && bitB == 0) {
```

Testcase | Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

a =
2

b =
6

c =
5

Console | Run | Submit

Description | Editorial | Solutions (7.4K) | Submissions

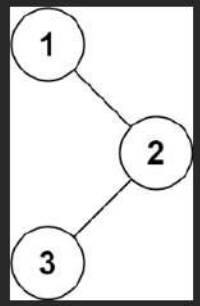
94. Binary Tree Inorder Traversal

Easy | 11.6K | 588

Companies

Given the `root` of a binary tree, return *the inorder traversal of its nodes' values.*

Example 1:




```
Input: root = [1,null,2,3]
Output: [1,3,2]
```

Example 2:

```
Input: root = []
Output: []
```

Example 3:

```
Input: root = [1]
Output: [1]
```

Java | Auto

```java
*/
class Solution {
    private List<Integer> res = new ArrayList<>();
    public List<Integer> inorderTraversal(TreeNode root) {
        traverse(root);
        return res;
    }

    private void traverse(TreeNode root) {
        if (root == null) {
            return;
        }
        traverse(root.left);
```

Testcase | Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

root =
[1,null,2,3]

Output

[1,3,2]

Expected

Console | Run | Submit