LeetCode 1360. 日期之间隔几天

1360. 日期之间隔几天

难度 简单 10

请你编写一个程序来计算两个日期之间隔了多少天。

日期以字符串形式给出，格式为 `YYYY-MM-DD`，如示例所示。

示例 1：

```
输入：date1 = "2019-06-29", date2 = "2019-06-30"
输出：1
```

示例 2：

```
输入：date1 = "2020-01-15", date2 = "2019-12-31"
输出：15
```

提示：

- 给定的日期是 `1971` 年到 `2100` 年之间的有效日期。

```
class Solution {
public:
    int daysBetweenDates(string date1, string date2) {
        return abs(get(date1) - get(date2));
    }
    bool isLeap(int year){
        return year % 100 && year % 4 == 0 || year % 400 == 0;
    }

    int MonthDays[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

    int get(string date)
    {
        int year, month, day;
        sscanf(date.c_str(), "%d-%d-%d", &year, &month, &day);//读取字符串c_str()
读取第一个字符地址。

        int days = 0;
        for(int i = 1971;i < year;i ++) days += 365 + isLeap(i);
        for(int i = 0;i < month;i ++)
        {
            if(i == 2) days += 28 + isLeap(year);
            else days += MonthDays[i];
        }
```

```
        return days + day;
    }
};
```

LeetCode 1361. 验证二叉树

1361. 验证二叉树

难度 中等 9

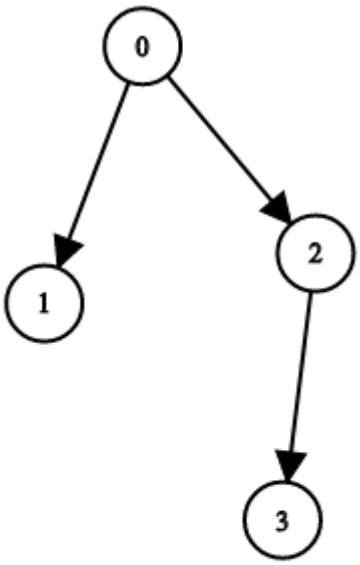
二叉树上有 `n` 个节点，按从 `0` 到 `n - 1` 编号，其中节点 `i` 的两个子节点分别是 `leftChild[i]` 和 `rightChild[i]`。

只有 **所有** 节点能够形成且 **只** 形成 **一颗** 有效的二叉树时，返回 `true`；否则返回 `false`。

如果节点 `i` 没有左子节点，那么 `leftChild[i]` 就等于 `-1`。右子节点也符合该规则。

注意：节点没有值，本问题中仅仅使用节点编号。

示例 1：



输入： `n = 4, leftChild = [1,-1,3,-1], rightChild = [2,-1,-1,-1]`
输出： `true`

```
/*搜索过程中1.不能有重复的点。2.所有点都被找到。*/
class Solution {
public:
    bool validateBinaryTreeNodes(int n, vector<int>& leftChild, vector<int>&
rightChild) {
        vector<int> d(n);//入度数。
        for (int i = 0; i < n; i ++ ) {
            if (leftChild[i] != -1) d[leftChild[i]] ++ ;
            if (rightChild[i] != -1) d[rightChild[i]] ++ ;
        }//计算所有入度
```

```
        int root = 0;
        while (root < n && d[root]) root ++ ;//寻找根节点

        if (root == n) return false;//没有根节点。

        vector<bool> st(n);//是否被遍历过
        st[root] = true;
        queue<int> q;//宽搜
        q.push(root);

        while (q.size()) {
            int t = q.front();
            q.pop();

            int sons[] = {leftChild[t], rightChild[t]};//遍历左右儿子
            for (auto s : sons) {
                if (s != -1) {
                    if (st[s]) return false;//是否重复遍历了
                    st[s] = true;
                    q.push(s);
                }
            }
        }

        for (auto state : st)//是否所有点都被遍历了
            if (!state)
                return false;

        return true;
    }
};
```

LeetCode 1362. 最接近的因数

1362. 最接近的因数

难度 中等 6

给你一个整数 `num`，请你找出同时满足下面全部要求的两个整数：

- 两数乘积等于 `num + 1` 或 `num + 2`
- 以绝对差进行度量，两数大小最接近

你可以按任意顺序返回这两个整数。

示例 1：

```
输入：num = 8
输出：[3,3]
解释：对于 num + 1 = 9，最接近的两个因数是 3 & 3；对于 num + 2 = 10, 最接近的两个因数是 2 & 5，因此返回 3 & 3 。
```

示例 2：

```
输入：num = 123
输出：[5,25]
```

示例 3：

```
输入：num = 999
输出：[40,25]
```

```cpp
class Solution {
public:
    vector<int> closestDivisors(int num) {
        for (int i = sqrt(num + 2); i; i -- ) {
            if ((num + 1) % i == 0) return {i, (num + 1) / i};
            if ((num + 2) % i == 0) return {i, (num + 2) / i};
        }
        return {};
    }
};
```