

1. 算法思路：排序+贪心算法

复杂度分析： $O(N\log N) + O(N) = O(N\log N)$

```
class Solution {
public:
    int findContentChildren(vector<int>& g, vector<int>& s) {
        int i = 0, j = 0, n = 0;
        sort(g.begin(), g.end());
        sort(s.begin(), s.end());
        while ( i < g.size() && j < s.size() ){
            if ( g[i] <= s[j] ){
                n++;
                i++;
                j++;
            }
            else{
                j++;
            }
        }
        return n;
    }
};
```

The screenshot shows the LeetCode submission page for the 'Assign Cookies' problem. The page includes the following information:

- Execution Results:** The code passed the test cases. Execution time: 44 ms (14.11% of users). Memory usage: 17.2 MB (5.38% of users).
- Submission History Table:**

提交结果	执行用时	内存消耗	语言	提交时间	备注
通过	44 ms	17.2 MB	C++	2021/06/07 10:52	添加备注
解答错误	N/A	N/A	C++	2021/06/07 10:50	添加备注
解答错误	N/A	N/A	C++	2021/06/07 10:49	添加备注

The code editor shows the following C++ code:

```
1 class Solution {
2 public:
3     int findContentChildren(vector<int>& g, vector<int>& s) {
4         int i = 0, j = 0, n = 0;
5         sort(g.begin(), g.end());
6         sort(s.begin(), s.end());
7         while ( i < g.size() && j < s.size() ){
8             if ( g[i] <= s[j] ){
9                 n++;
10                i++;
11                j++;
12            }
13            else{
14                j++;
15            }
16        }
17        return n;
18    }
19 }
```

The test cases show the following input and output:

输入	输出	预期结果
[10,9,8,7] [5,6,7,8]	2	2

2. 算法思路：多轮循环，在 a 之间插入 b

复杂度分析：O(N)

```
class Solution {
public:
    string strWithout3a3b(int a, int b) {
        string s;
        s.insert(0, a, 'a');
        int i = 2;
        while(i<s.length()+1&&b>0){
            s.insert(i, 1, 'b');
            i+=3;
            b--;
        }
        //cout << b << endl;
        //cout << s << endl;
        i = 2;
        while(i<s.length()+1&&b>0){
            s.insert(i, 1, 'b');
            i += 4;
            b--;
        }
        i = 1;
        while(i<s.length()+1&&b>0){
            s.insert(i, 1, 'b');
            i += 5;
            b--;
        }
        i = 1;
        while(i<s.length()+1&&b>0){
            s.insert(i, 1, 'b');
            i += 6;
            b--;
        }
        if(b>0){
            s.insert(0, 1, 'b');
            b--;
        }
        if ( b>0 ){
            s.insert(0, 1, 'b');
            b--;
        }
        if ( b>0 ){
            s += 'b';
            b--;
        }
    }
};
```

```

    }
    if ( b > 0 ){
        s+='b';
    }
    return s;
}
};

```

leetcod.com/problems/string-without-aaa-or-bbb/submissions/

LeetCode Explore Problems Interview Contest Discuss Stocks June LeetCode Challenge 2021 Premium

力扣「中文社区」现已上线，全新「个人主页」更优体验，即刻加入先人一步攒积分！

新功能推荐：全新积分商场礼品 | 上万社区题解 | 企业题库 | 面试模拟 | 更多竞赛 | 轻松数据同步使用已有积分

Description Solution Discuss (333) Submissions

Success Details >

Runtime: 0 ms, faster than 100.00% of C++ online submissions for String Without AAA or BBB.

Memory Usage: 6.3 MB, less than 12.87% of C++ online submissions for String Without AAA or BBB.

Next challenges:

Wiggle Subsequence Minimum Number of K Consecutive Bit Flips

Maximum Value after Insertion

Show off your acceptance: f t in

Console - Contribute i

Run Code Submit

3. 算法思路：从底层往上搜索，每一层逐步缩小数组

复杂度分析：时间复杂度 $O(N^2)$ (N 为层数)

空间复杂度 $O(N)$ (只需要一个额外的 $O(N)$ 数组)

```

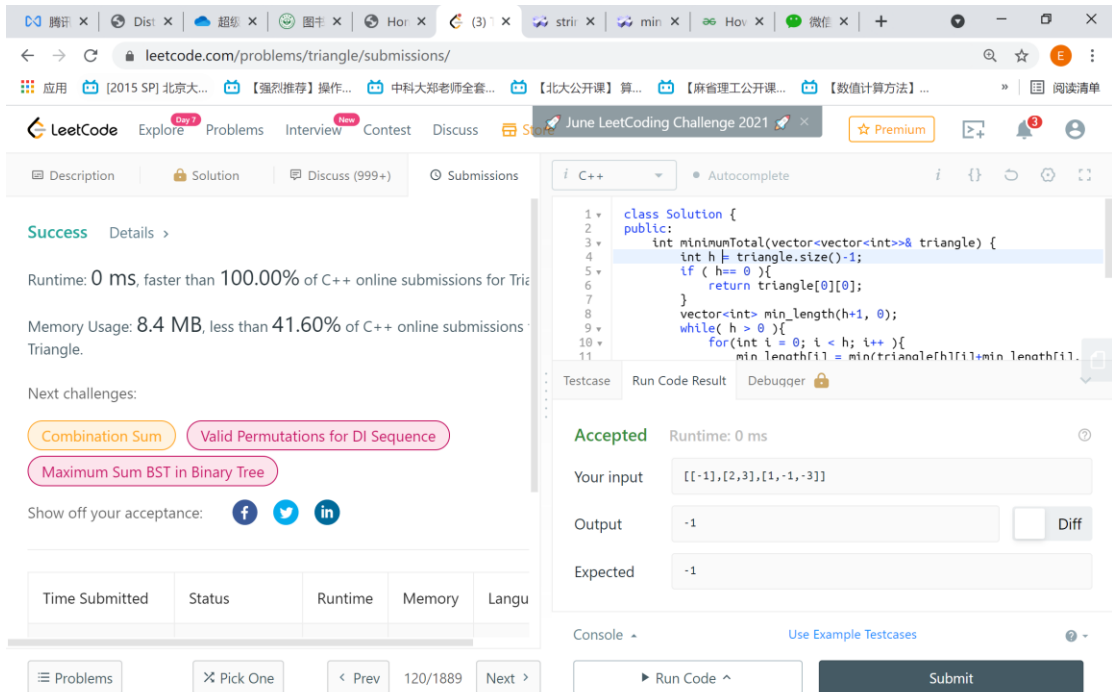
class Solution {
public:
    int minimumTotal(vector<vector<int>>& triangle) {
        int h = triangle.size()-1;
        if ( h== 0 ){
            return triangle[0][0];
        }
        vector<int> min_length(h+1, 0);
        while( h > 0 ){
            for(int i = 0; i < h; i++){
                min_length[i] = min(triangle[h][i]+min_length[i],
                                    triangle[h][i+1]+min_length[i+1]);
            }
            min_length.pop_back();
        }
    }
};

```

```

        h--;
    }
    return min_length[0]+triangle[0][0];
}
};

```



4. 算法思路：动态规划，对于每一天，计算出当天能有的最大现金数和若持有股票的最大收益，最后输出最后一天能有的最大现金数
复杂度：遍历每一天， $O(N)$

```

class Solution {
public:
    int maxProfit(vector<int>& prices, int fee) {
        int cash = 0;
        int hold = -prices[0];
        for (auto & price:prices) {
            cash = max(cash, hold+price-fee);
            hold = max(hold, cash - price);
        }
        return cash;
    }
};

```

Time to Buy and Sell Stock with Transaction Fee.

Memory Usage: 55.1 MB, less than 59.55% of C++ online submissions
Best Time to Buy and Sell Stock with Transaction Fee.

Next challenges:

Best Time to Buy and Sell Stock II

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
06/07/2021 20:23	Accepted	72 ms	55.1 MB	cpp

```

1  class Solution {
2  public:
3      int maxProfit(vector<int>& prices, int fee) {
4          int cash = 0;
5          int hold = -prices[0];
6          for (auto & price: prices) {
7              cash = max(cash, hold + price - fee);
8              hold = max(hold, cash - price);
9          }
10         return cash;
11     }
12 }

```

Your previous code was restored from your local storage. [Reset to default](#)

Problems Pick One < Prev 714/1889 Next > Console Contribute i Run Code ^ Submit

5. 算法思路：动态规划，遍历字符串的每个字符并更新编码方案数

复杂度分析：遍历字符串， $O(N)$

```

class Solution {
public:
    int numDecodings(string s) {
        if ( s[0] == '0' ){
            return 0;
        }
        int n1 = 0, n2 = 1, hold;
        for ( int i = 1; i < s.length(); i++ ){
            if ( s[i] == '0' ){
                if ( s[i-1] != '1' && s[i-1] != '2' ){
                    return 0;
                }
                hold = n1;
                n1 = n2;
                n2 = hold == 0? 1:hold;
            }
            else{
                string s1;
                int n = stoi(s1 + s[i-1] + s[i]);
                if ( n >= 11 && n <= 26 ){
                    hold = n2;
                    n2 = n1 == 0? 1+n2:n1+n2;
                    n1 = hold;
                }
            }
        }
    }
}

```

```

        else{
            n1 = n2;
        }
    }
}
return n2;
}
};

```

腾讯企 | Distrib | 超级计 | 图书 | Home | (3) x | string: | min - | How t | 微信(1) | c++ | + | - | X

leetcode.com/problems/decode-ways/submissions/

应用 [2015 SP] 北京大... 【强烈推荐】操作... 中科大郑老师全套... 【北大公开课】算... 【麻省理工公开课... 【数值计算方法】... 阅读清单

LeetCode Explore Problems Interview Contest Discuss June LeetCode Challenge 2021 Premium

力扣「中文社区」现已上线，全新「个人主页」更优体验，即刻加入先人一步攒积分！
新功能推荐：全新积分商场礼品 | 上万社区题解 | 企业题库 | 面试模拟 | 更多竞赛 | 轻松数据同步使用已有积分

Description Solution Discuss (999+) Submissions

Success Details >

Runtime: 0 ms, faster than 100.00% of C++ online submissions for Decode Ways.

Memory Usage: 6.1 MB, less than 81.80% of C++ online submissions for Decode Ways.

Next challenges:
Decode Ways II

Show off your acceptance: f t in

Problems Pick One < Prev 91/1889 Next >

https://support.leetcode-cn.com/hc/kb/article/1286274/?utm...

```

1 class Solution {
2 public:
3     int numDecodings(string s) {
4         if (s[0] == '0') {
5             return 0;
6         }
7         int n1 = 0, n2 = 1, hold;
8         for (int i = 1; i < s.length(); i++) {
9             if (s[i] == '0') {
10                if (s[i-1] != '1' && s[i-1] != '2') {
11                    return 0;
12                }
13                hold = n1;
14                n1 = n2;
15                n2 = hold == 0 ? 1 : hold;
16            }
17            else {
18                string s1;
19                int n = stoi(s1 + s[i-1] + s[i]);
20                if (n >= 11 && n <= 26) {
21                    hold = n2;
22                    n2 = n1 == 0 ? 1 + n2 : n1 + n2;
23                    n1 = hold;
24                }
25            }
26        }
27        return n2;
28    }
29 };

```

Console Contribute i

Run Code ^ Submit