

LeetCode 20

题目要求

给定一个只包括 '(', ')', '{', '}', '[', ']' 的字符串 *s*，判断字符串是否有效。

有效字符串需满足：

- 左括号必须用相同类型的右括号闭合。
- 左括号必须以正确的顺序闭合。

题目实现

思路

该问题为括号匹配问题，适合利用栈结构实现，遇到左括号就将对应的右括号压入栈中，遇到右括号就弹出栈顶元素，然后判断栈是否为空，如果为空则说明括号可以完全匹配

特判：如果输入的字符串长度为奇数，就不可能完全匹配，直接返回false

代码

```
class Solution {
public:
    bool isValid(string s) {
        stack<int> st;
        //特判
        if(s.size() % 2 != 0) return false;

        for(int i = 0; i < s.size(); i++){
            if(s[i] == '(') st.push(')');
            else if(s[i] == '[') st.push(']');
            else if(s[i] == '{') st.push('}');
            else if(st.empty() || s[i] != st.top()) return false;
            else st.pop();
        }

        return st.empty();
    }
};
```

执行结果

执行结果: **通过** [显示详情](#)

[添加备注](#)

执行用时: **0 ms**, 在所有 C++ 提交中击败了 **100.00%** 的用户

内存消耗: **6.1 MB**, 在所有 C++ 提交中击败了 **91.71%** 的用户

炫耀一下:



[写题解, 分享我的解题思路](#)

提交结果	执行用时	内存消耗	语言	提交时间	备注
通过	0 ms	6.1 MB	C++	2021/08/05 10:41	添加备注

OJ 263 火车进栈

题目要求

有 n 列火车按1到 n 的顺序进站, 车站只有一个进出口, 先进后出 (FILO)

进站的火车编号顺序为 $1 \sim n$, 按照火车编号从小到大的顺序, 输出前20中可能的出站方案

输入: 输入一行一个整数 n 。($n \leq 20$)

输出: 输出前20种答案, 每行一种, 不要空格

题目实现

思路

火车站的结构是典型的栈结构, n 列火车相当于 n 个数据节点, 题目要求 n 列火车进站再出站, 就可以类比为 n 个数据都进栈再出栈

针对每个火车来说, 有两个操作: 进栈 (+)、出栈 (-)

所以对 n 列火车完成进栈出栈的操作就是 n 个元素两种操作的排列数, 回溯算法是实现求排列组合的比较好的算法

使用深度优先搜索实现回溯算法, 并利用栈数据结构模拟火车进出站的操作

代码

```
#include<iostream>
#include<vector>
#include<stack>
#include<algorithm>

using namespace std;
typedef long long ll;
```

```


int n;
stack<int> st;
int num=1;
vector<int> vec;
int cnt=20;

void dfs(){
    if(!cnt) return;
    if(vec.size()==n){
        cnt--;
        for(int i=0;i<n;i++){
            cout<<vec[i];
        }
        cout<<endl;
        return;
    }
    //如果栈中有元素优先弹出并加入结果数组中，然后向下深搜
    if(st.size()){
        vec.push_back(st.top());
        st.pop();
        dfs();
        st.push(vec.back()); //回溯
        vec.pop_back(); //回溯
    }
    if(num<=n){
        st.push(num);
        num++;
        dfs();
        num--;
        st.pop();
    }
}

int main(){
    cin>>n;
    dfs();
    return 0;
}

```

执行结果

#263. 火车进栈	CPP023_03_gaoxiang 	100	4ms	2580kb	C++11	808b	5 秒前提交
------------	---	-----	-----	--------	-------	------	--------

OJ 265 括号画家

题目要求

定义美观的括号序列具备以下性质：

- 空的括号序列是美观的
- 若括号序列'A'是美观的，则括号序列'(A)、[A]、{A}'也是美观的（即在A序列的外面再套一层对称的括号所得的序列也是美观的）

- 若括号序列'A' 'B'都是美观的，则括号序列'AB'也是美观的（即两个美观的括号序列拼接在一起也是美观的）

找出一段连续的括号序列中满足美观的最长长度是多少

输入：1一个长度为N的括号序列($5 \leq N \leq 10000$)

输出：一个整数，表示最长的美观的连续子序列的长度

题目实现

思路

该题也是个括号匹配问题，利用栈实现配对

1. 为了获取美观括号子串的长度，栈中存储pair<char, int>，同时记录括号字符和对应的下标
2. 在开始遍历字符串前先向栈中压入一个垫底节点{'e', -1}，避免边界的讨论
3. 遍历字符串，遇到左括号，压栈
4. 遇到右括号尝试与栈顶元素配对
 1. 如果配对成功，将栈顶元素弹出，用i - top().second更新当前的美观括号子串长度cur（取大）
 2. 如果配对不成功，说明当前的字符不能再保证后续字符能构成美观子串了，需要重新统计
 1. 用cur 更新最终结果ans（取大）
 2. 重置cur
 3. 重置栈，将栈清空，然后再压入一个垫底元素，注意垫底元素的下标要取当前循环的i

代码


```
#include<iostream>
#include<stack>
#include<string>
using namespace std;

stack<pair<char, int>> st;

int main() {
    string s;
    cin >> s;
    int cur = 0; //用于存储当前美观括号子序列的长度
    int ans = 0; //用于存储最长美观括号子序列的长度
    st.push({'c', -1});
    for (int i = 0; i < s.size(); ++i) {
        if (s[i] == '(' || s[i] == '[' || s[i] == '{') {
            st.push({s[i], i}); //遇到左括号压栈
        } else { //遇到右括号，尝试配对
            if ((s[i] == ')' && st.top().first == '(') || (s[i] == ']' && st.top().first == '[') || (s[i] == '}' && st.top().first == '{')) { //配对成功
                st.pop(); //弹出栈顶
                cur = max(i - st.top().second, cur); //更新当前美观括号子序列的长度
            } else { //配对不成功
                ans = max(cur, ans); //更新结果
                cur = 0; //重置当前美观括号子序列长度
                while (!st.empty()) st.pop(); //重置栈
                st.push({'c', i});
            }
        }
    }
}
```

```
    }  
}  
ans = max(cur, ans);  
cout << ans << endl;  
return 0;  
}
```

执行结果

#265. 括号画家	CPP023_03_gaoxiang 	100	0ms	2528kb	C++11	1.2kb	7 秒前提交
------------	---	-----	-----	--------	-------	-------	--------