

1.字符串

计算器问题

- [224. 基本计算器](#)
- [227. 基本计算器 II](#)
- [770. 基本计算器 IV](#)
- 三叶通用题解

```
1 class Solution {
2 public:
3     //添加一个INT_MIN的原因: 运算符遇到(, 应该先入栈, 当前括号内有优先级更高的则计算, 低
4     unordered_map<char,int> oper_pri = { {'+',1}, {'-',1}, {'*',2}, {'/',2}, {'%'
5     stack<char> opers;
6     stack<long long> nums;
7     int calculate(string s) {
8         //处理所有空格
9         string temp = "";
10        for(char c : s){
11            if(c != ' ') temp += c;
12        }
13        s = temp;
14
15        //-除了减, 还有一元运算符负的概念
16        //所以: 避免-5的处理, 0-5更加通用
17        nums.push(0);
18        int n = s.size();
19        for(int i = 0; i < n; i++){
20            //处理左括号
21            if(s[i] == '(') {
22                opers.push('(');
23                //-除了减, 还有一元运算符负的概念
24                //所以: 避免-5的处理, 0-5更加通用
25                if(s[i + 1] == '-'){
26                    i++;
27                    nums.push(0);
28                    opers.push('-');
29                }
30            }
31            //处理右括号, 要把当前括号中没计算完的计算完
32            else if(s[i] == ')'){
33                while(opers.top() != '('){
34                    cal();
35                }
36                opers.pop();
37            }
38            //处理数字, 要读就一次性把数字读完
39            else if(s[i] >= '0' && s[i] <= '9'){
40                int l = i;
41                while(i + 1 < n && s[i + 1] >= '0' && s[i + 1] <= '9') i++;
42                nums.push(stoll(s.substr(l,i - l + 1)));
43            }
44            //处理运算符: 每次出现了优先级更低的运算符, 才去算前一个优先级高的运算符, 否则
```

```
45         else{
46             while(!opers.empty() && oper_pri[opers.top()] >= oper_pri[s[i]])
47                 cal();
48         }
49         opers.push(s[i]);
50     }
51 }
52 //计算所有剩余的运算法
53 while(!opers.empty() && opers.top() != '('){
54     cal();
55 }
56 return nums.top();
57 }
58 void cal(){
59     long long b = nums.top(); nums.pop();
60     long long a = nums.top(); nums.pop();
61     char oper = opers.top(); opers.pop();
62     long long res;
63     switch (oper) {
64         case '+': res = a + b; break;
65         case '-': res = a - b; break;
66         case '*': res = a * b; break;
67         case '/': res = a / b; break;
68         case '%': res = a % b; break;
69         case '^': res = pow(a, b); break;
70     }
71     nums.push(res);
72 }
73 };
```