表达式求值

闲扯

昨天发烧39.....

今天满血复活.....

这一天写的代码也就这一题比较顺利了.....

题目信息

描述

ACM队的mdd想做一个计算器，但是，他要做的不仅仅是一计算一个A+B的计算器，他想实现随便输入一个表达式都能求出它的值的计算器，现在请你帮助他来实现这个计算器吧。

比如输入：“1+2/4=”，程序就输出1.50（结果保留两位小数）

输入

第一行输入一个整数n，共有n组测试数据（n<10)。 每组测试数据只有一行，是一个长度不超过1000的字符串，表示这个运算式，每个运算式都是以“=”结束。这个表达式里只包含+-*/与小括号这几种符号。其中小括号可以嵌套使用。数据保证输入的操作数中不会出现负数。 数据保证除数不会为0

输出

每组都输出该组运算式的运算结果，输出结果保留两位小数。

样例输入

```
2
1.000+2/4=
((1+2)*5+1)/4=
```

样例输出

```
1.50
4.00
```

AC代码

```
#include <iostream>
#include <stack>
#include <sstream>
#include <map>
#include <cstdio>
#include <string>
#include <queue>

using namespace std;

stack<char> op;
stack<string> postexp;
stack<string> postexp1;
stack<double> postexpnum;
```

```
map<char,int> lpri;
map<char,int> rpri;


int main()
{
    lpri['=']=0;
    lpri['(']=1;
    lpri['+']=3;
    lpri['-']=3;
    lpri['*']=5;
    lpri['/']=5;
    lpri[')']=6;

    rpri['=']=0;
    rpri['(']=6;
    rpri['+']=2;
    rpri['-']=2;
    rpri['*']=4;
    rpri['/']=4;
    rpri[')']=1;
    int N;
    cin>>N;
    while(N--)
    {
        string str;
        cin>>str;
        op.push('=');
        string tmpnum;
        // cout<<str<<endl;
        for(unsigned int i=0;i<str.length();i++)
        {
            if(rpri.count(str[i]))
            {
                if(tmpnum.length()!=0)
                {
                    postexp.push(tmpnum);
                    tmpnum.clear();
                }

                if(rpri[str[i]]>lpri[op.top()])
                    op.push(str[i]);
                else
                {
                    if(str[i]==')')
                    {
                        while(op.top()!='(')
                        {
                            string str1;
                            str1+=op.top();
                            op.pop();
                            postexp.push(str1);
                        }
                        op.pop();
                    }
                    else
                    {
                        while(lpri[op.top()]>rpri[str[i]])
                        {
                            string str1;
                            str1+=op.top();
```

```
                        op.pop();
                        postexp.push(str1);
                    }
                    op.push(str[i]);
                }
            }
        }
        else
        {
            tmpnum+=str[i];
        }
    }
    while(!postexp.empty())
    {
        postexp1.push(postexp.top());
        postexp.pop();
    }
    while(!postexp1.empty())
    {
        double num;
        stringstream ss;
        if(postexp1.top()=="+")
        {
            num=postexpnum.top();
            postexpnum.pop();
            num+=postexpnum.top();
            postexpnum.pop();
            postexpnum.push(num);

        }
    else if(postexp1.top()=="-")
        {
            num=postexpnum.top();
            postexpnum.pop();
            num=postexpnum.top()-num;
            postexpnum.pop();
            postexpnum.push(num);

        }
        else if(postexp1.top()=="*")
        {
            num=postexpnum.top();
            postexpnum.pop();
            num*=postexpnum.top();
            postexpnum.pop();
            postexpnum.push(num);

        }
        else if(postexp1.top()=="/")
        {
            num=postexpnum.top();
            postexpnum.pop();
            num=postexpnum.top()/num;
            postexpnum.pop();
            postexpnum.push(num);


        }
        else
        {
            ss<<postexp1.top();
```

```
                ss>>num;
                postexpnum.push(num);
            }

            postexp1.pop();
        }
        //cout<<postexpnum.top();
        printf("%.2lf\n",postexpnum.top());
        postexpnum.pop();



    }
    return 0;
}
```

作者语

代码没有优化写了那么多行