# 数学表达式计算

王金华 2023141460081

## 1. 问题描述

从键盘输入中缀算术表达式（包括括号），计算表达式的值。

## 2. 基本要求

(1) 程序能对输入的表达式进行简单的错误判断，并提供适当提示。

(2) 支持单目运算符：正（+）、负（-）。

## 3. 工具/准备工作

计算机，安装有visual studio 2022集成开发环境。

## 4. 实验步骤

#include<iostream>

#include<stack>

#include<cmath>

using namespace std;

template<class T>

class  Calculator

{

private:

    stack<T> opnd;

    stack<char> optr;

    char Getchar();

    int operPrior(char op);

    void get2Operands(T& left, T& right);

    T operate(T left, char op, T right);

    bool isOperator(char ch);

public:

    Calculator() {};

    virtual ~Calculator() {};

    void Run();

};

template<class T>

inline char Calculator<T>::Getchar()

{

    char ch;

    cin >> ch;

    return ch;

}

template<class T>

inline int Calculator<T>::operPrior(char op)

{

    switch (op)

    {

    case'=':

        return 1;

    case '(':

    case ')':

        return 2;

    case  '+':

    case'-':

        return 3;

    case '\*':

    case '/':

    case '%':

        return 4;

    case '^':

        return 5;

    default:

        return 0;

    }

}

template<class T>

inline void Calculator<T>::get2Operands(T& left, T& right)

{

    if (opnd.size() > 1) {

        right = opnd.top();

        opnd.pop();

        left = opnd.top();;

        opnd.pop();

    }

    else {

        throw "表达式错误!";

    }

}

template<class T>

inline T Calculator<T>::operate(T left, char op, T right)

{

    switch (op)

    {

    case '+':

        return left + right;

    case '-':

        return left - right;

    case '\*':

        return left \* right;

    case '/':

        if (right == 0)throw "0不能作为除数";

        return left / right;

    case '^':

        return pow(left, right);

    case '%':

        if (is\_integral<T>::value) {

            return left % right;

        }

        else {

            throw "取模只适用于整数";

        }

    }

}

template<class T>

inline bool Calculator<T>::isOperator(char ch)

{

    if (operPrior(ch) > 0) {

        return true;

    }

    else {

        return false;

    }

}

template<class T>

inline void Calculator<T>::Run()

{

    while (optr.empty() == false)

    {

        optr.pop();

    }

    while (opnd.empty() == false)

    {

        opnd.pop();

    }

    optr.push('=');

    char ch;

    char priorChar;

    char optrTop;

    T operand;

    char op;

    priorChar = '=';

    ch = Getchar();

    if (optr.empty())throw "表达式错误!";

    optrTop = optr.top();

    while (optrTop != '=' || ch != '=')

    {

        if (isdigit(ch) || ch == '.')

        {

            cin.putback(ch);

            cin >> operand;

            opnd.push(operand);

            priorChar = '0';

            ch = Getchar();

        }

        else  if (!isOperator(ch))

        {

            throw "表达式错误!";

        }

        else

        {

            if ((priorChar == '=' || priorChar == '(') && (ch == '+' || ch == '-'))

            {

                opnd.push(0);

                priorChar = '0';

            }

            if (optrTop == ')' && ch == '(' || optrTop == '(' && ch == '=' || optrTop == '=' && ch == ')')throw "表达式错误!";

            else if (optrTop == '(' && ch == ')')

            {

                if (optr.empty())throw "表达式错误!";

                optrTop = optr.top();

                optr.pop();

                ch = Getchar();

                priorChar = ')';

            }

            else if (ch == '(' || operPrior(optrTop) < operPrior(ch))

            {

                optr.push(ch);

                priorChar = ch;

                ch = Getchar();

            }

            else

            {

                if (optr.empty())

                    throw "表达式错误!";

                op = optr.top();

                optr.pop();

                T left, right;

                get2Operands(left, right);

                opnd.push(operate(left, op, right));

            }

        }

        if (optr.empty())throw "表达式错误!";

        optrTop = optr.top();

    }

    if (opnd.empty())throw "表达式错误!";

    operand = opnd.top();

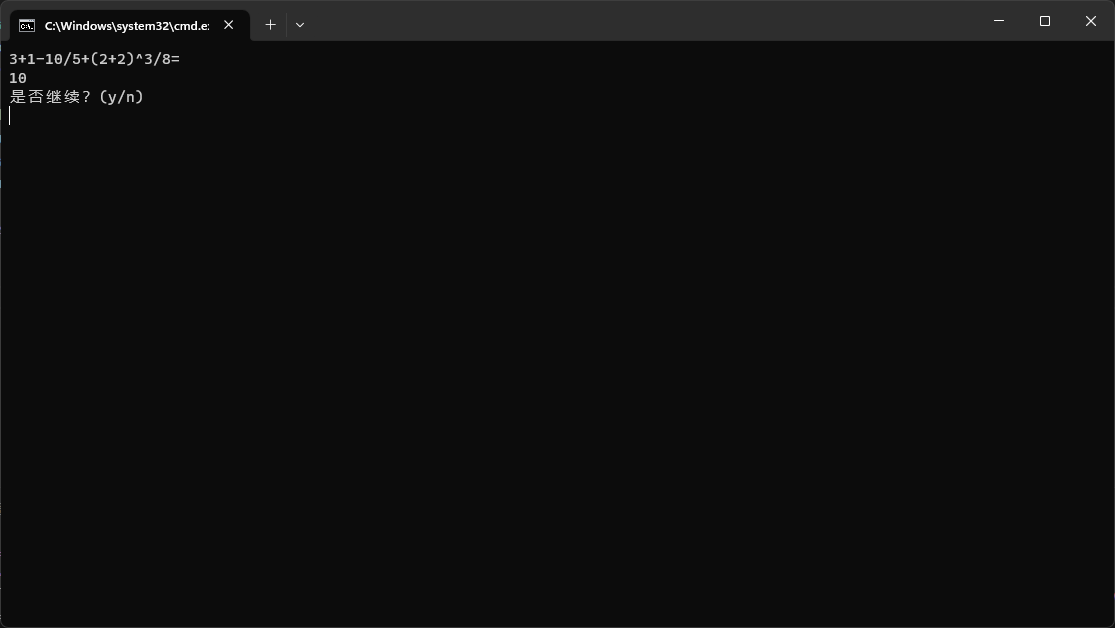
    cout << operand << endl;

}

## 5. 测试与结论

使用 C++ 类模板实现计算器类 Calculator，包含操作数栈、操作符栈、辅助函数等。

Run() 函数实现表达式求值算法。



## 6. 实验总结

测试覆盖各种情况，确保程序满足基本要求。