Project2 实验报告

21307035 邓栩瀛

1、程序功能简要说明。

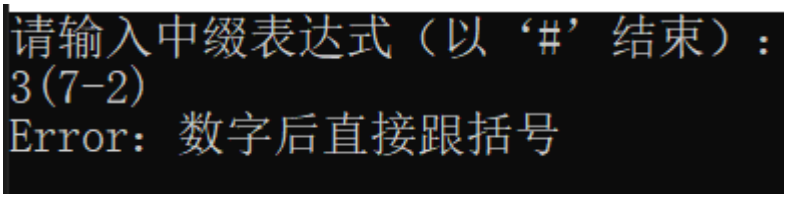
字符序列的形式从终端输入语法正确的、不含变量的整数表达式，实现对算术混合运算表达式的求值。

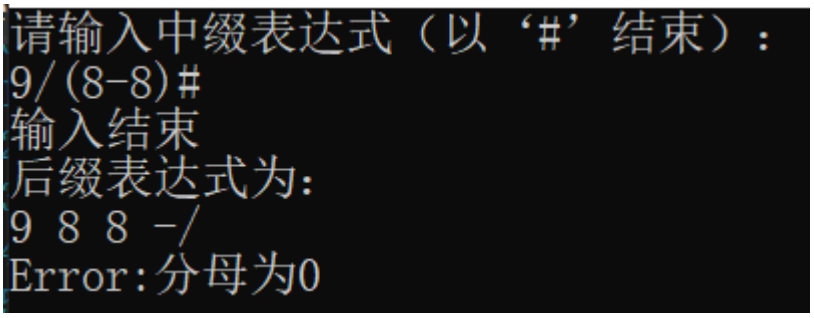
2、程序运行截图，包括计算功能演示、部分实际运行结果展示、命令行或交互式、界面效果等。

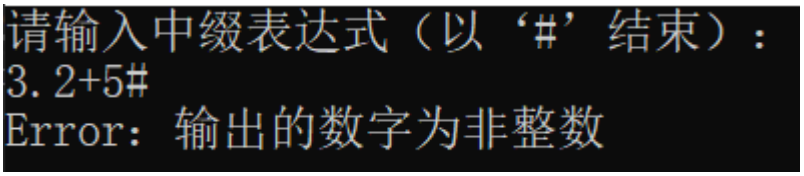
```
请输入中缀表达式（以‘#’结束）：
2*(6+2*(3+6*(6+6)))#
输入结束
后缀表达式为:
2 6 2 3 6 6 6 +*+*+*

运算符栈: #  运算数栈: 2
运算符栈: # *  运算数栈: 2
运算符栈: # *  运算数栈: 2
运算符栈: # * (  运算数栈: 2
运算符栈: # * (  运算数栈: 2 6
运算符栈: # * ( +  运算数栈: 2 6
运算符栈: # * ( +  运算数栈: 2 6 2
运算符栈: # * ( + *  运算数栈: 2 6 2
运算符栈: # * ( + *  运算数栈: 2 6 2
运算符栈: # * ( + * (  运算数栈: 2 6 2
运算符栈: # * ( + * (  运算数栈: 2 6 2 3
运算符栈: # * ( + * ( +  运算数栈: 2 6 2 3
运算符栈: # * ( + * ( +  运算数栈: 2 6 2 3 6
运算符栈: # * ( + * ( + *  运算数栈: 2 6 2 3 6
运算符栈: # * ( + * ( + *  运算数栈: 2 6 2 3 6
运算符栈: # * ( + * ( + * (  运算数栈: 2 6 2 3 6
运算符栈: # * ( + * ( + * (  运算数栈: 2 6 2 3 6 6
运算符栈: # * ( + * ( + * ( +  运算数栈: 2 6 2 3 6 6
运算符栈: # * ( + * ( + * ( +  运算数栈: 2 6 2 3 6 6 6
运算符栈: # * ( + * ( + * (  运算数栈: 2 6 2 3 6 12
运算符栈: # * ( + * ( + * (  运算数栈: 2 6 2 3 6 12
运算符栈: # * ( + * ( + *  运算数栈: 2 6 2 3 6 12
运算符栈: # * ( + * ( + *  运算数栈: 2 6 2 3 6 12
运算符栈: # * ( + * ( +  运算数栈: 2 6 2 3 72
运算符栈: # * ( + * ( +  运算数栈: 2 6 2 3 72
运算符栈: # * ( + * (  运算数栈: 2 6 2 75
运算符栈: # * ( + * (  运算数栈: 2 6 2 75
运算符栈: # * ( + *  运算数栈: 2 6 2 75
运算符栈: # * ( + *  运算数栈: 2 6 2 75
运算符栈: # * ( +  运算数栈: 2 6 150
运算符栈: # * ( +  运算数栈: 2 6 150
运算符栈: # * (  运算数栈: 2 156
运算符栈: # * (  运算数栈: 2 156
运算符栈: # *  运算数栈: 2 156
运算符栈: # *  运算数栈: 2 156
运算符栈: #  运算数栈: 312
运算符栈: #  运算数栈: 312
运算符栈: #  运算数栈: 312
运算符栈: #  运算数栈: 312
计算结果为: 312
```

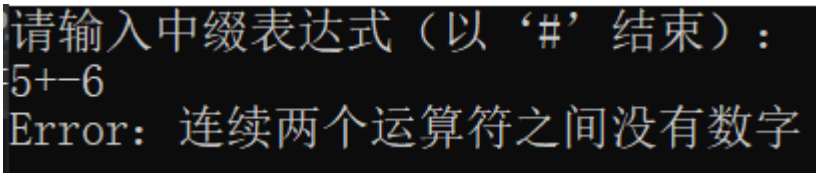
（1） 部分运行截图展示如上，用户输入表达式，直至输入“#”，停止输入

（2） 具有识别错误的功能。









3、部分关键代码及其说明。

（1） 输入输出界面实现

```
int main()
{
    char input[50], output[50];
    double ans = 0.0;
    cout << "请输入中缀表达式（以‘#’结束）：" << endl;
    cin >> input;

    while (strcmp(input, "#") != 0)
    {
        int length = strlen(input);
        tran(input, output);
        cout << "后缀表达式为：" << endl;
        cout << output << endl;
        ans = cal(output);
        outputstack(input);
        cout << endl<<"计算结果为：" <<  ans << endl;
        if (input[length - 1] == '#')
        {
            exit(0);
        }
    }
```

（2） 中缀表达式转化为后缀表达式

对字符“1”“2”“3”“4”“5”“6”“7”“8”“9”“0”“+”“-”“*”“/”“#”“（”“）”进行分情况讨论，如为数字则存入数组 output 中，如为运算符，需考虑是否有出错可能，再将其放入临时栈中，经处理后再存入 output 数组中。

（3）后缀表达式的计算

```
double calculate(char* output)
{
    int i = 0, j = 0;
    int length = 0;
    int top = -1;
    double num_stack[10]={};//数字栈，存放表达式中的数字
    double temp=0;//计算数字
    char ch[10]={};//先把数字的表达式存到ch[10]中，再转化为数字存到num_stack中
    for (i = 0, j = 0; output[i] != '\0'; i++)
    {
        if (output[i]>='0' && output[i] <= '9')
        {
            j = 0;
            while (output[i] != ' ')//将output复制到ch中，直到数字结束
            {
                ch[j] = output[i];
                i++;
                j++;
            }
            ch[j] = '\0';
            for (j = 0; ch[j] != '\0'; j++)
            {
                length = j - 1;
            }
            for (j = 0, temp = 0.0; ch[j] != '\0'; j++)
            {
                temp += (double)(ch[j] - '0') * pow(10, length - j +1);
            }
            top++;
            num_stack[top] = temp;
        }
```

```
        else//如果是运算符
        {
            temp = num_stack[top--];
            switch (output[i])
            {
            case'+':num_stack[top] += temp;
                break;
            case'-':num_stack[top] -= temp;
                break;
            case'*':num_stack[top] *= temp;
                break;
            case'/':
                if (temp == 0)
                {
                    cout << "Error:分母为0" << endl;
                    exit(0);
                }
                num_stack[top] /= temp;
                break;
            }
        }
    }
return num_stack[top];
```

计算结果存入数组尾端，并将值返回。为减少出错概率，计算过程中，数字运算均用 double 类型变量存储。

4、程序运行方式简要说明。

用户输入合法的中缀表达式，可以得到相应的后缀表达式及其结果，以及

计算过程中运算符栈和运算数栈的变化情况；对于非法的表达式，可以给出相应的错误类型。