同濟大學
TONGJI UNIVERSITY

项目说明文档

表达式转换

指导教师：张颖

1751984 王舸飞

1.分析

1.1 背景分析

算数表达式有前缀表示法、中缀表示法和后缀表示法等方式。人们在日常表达算式时最常使用的是中缀表达式，但是在计算机中，使用中缀表达式有操作符的优先顺序问题，还有可加括号改变优先级的问题，与之相比，利用栈可以方便的对后缀表达式进行计算，因此在编译程序中一般采用后缀表达式求解表达式的值。

基于此，本项目要求设计一个中缀表达式转换成后缀表达式的程序。

1.2 功能分析

本次设计的项目首先应该满足一些基本要求，对于基本的数字和测试用例应该实现正确转化，但是在测试要求中可以发现，本次设计的转换程序还应对括号、多位整数、小数和负数进行处理，且输出的结尾不能有多余空格，应在考虑到这些的前提下对程序进行改进。

2.设计

2.1 数据结构设计

要将中缀表达式转为后缀需要考虑到运算符的优先级问题，遇到数字，可以直接输出，但是遇到运算符时除了考虑出现的先后顺序外，还应考虑到它在数学中的运算优先级，因此，可以采用栈的数据结构，让先出现的符号先进栈等待，当下一个出现的符号优先级没有栈内符号高时再进行输出。

在上述数据结构的基础上，附加判断语句可简单的实现题目要求的其他功能。

2.2 类结构设计

本次课程设计的栈未采用标准库，而是采用了自己定义的链式栈WorkStack。在创建该栈时，为方便后续操作创建一个附加头节点。

2.3 成员与操作设计

链式栈类(Workstack)

受保护的成员：

StackNode *first;//链表的头指针

公有操作：

WorkStack(){first=new StackNode;}//带附加头节点的构造函数

void Push(char pc);//将值为pc押入

void Pop(char &pc);//将弹出的值赋给变量pc

bool IsEmpty(){return first->next==NULL?true:false;};//判断栈是否为空

char GetTop(){return first->next->c;};//返回获取头节点的值

链表结点类(StackNode)

受保护的成员：

char c;//用于存储字符

StackNode *next;//指针域

其他相关函数：

bool issdigit(char pc) //将系统的isdigit函数更新的新判断函数，可将小数点纳入范围内

int isp(char c) //栈内优先级函数，根据字符不同返回优先级，它的赋值参考了课本99页的内容

int icp(char c) //栈外优先级函数，根据字符不同返回优先级，它的赋值参考了课本99页的内容

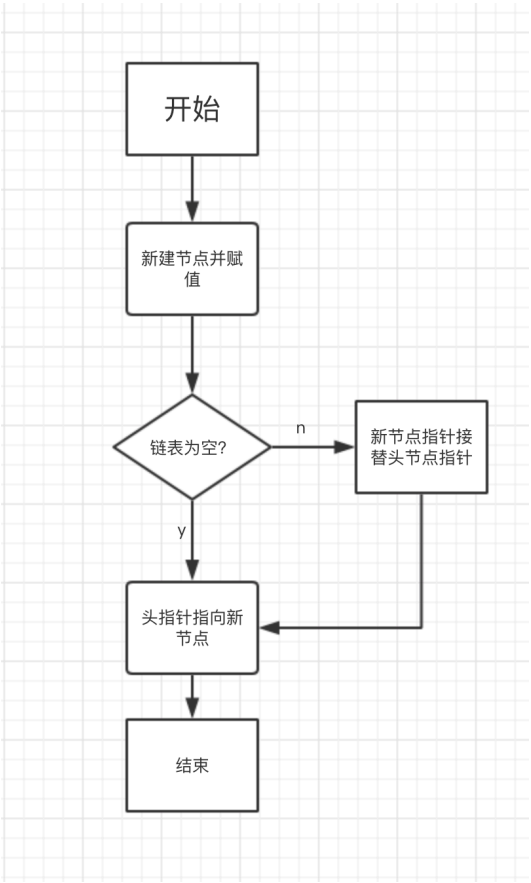
2.4 系统设计

程序运行之后，系统会立刻创建一个链式工作栈类s，之后将代表开始和结束的'#'押入栈中，随后对用户的输入进行逐级检查和读取，并利用栈的运算输出结果表达式。

3.实现

3.1 押入功能的实现

3.1.1 押入功能流程图

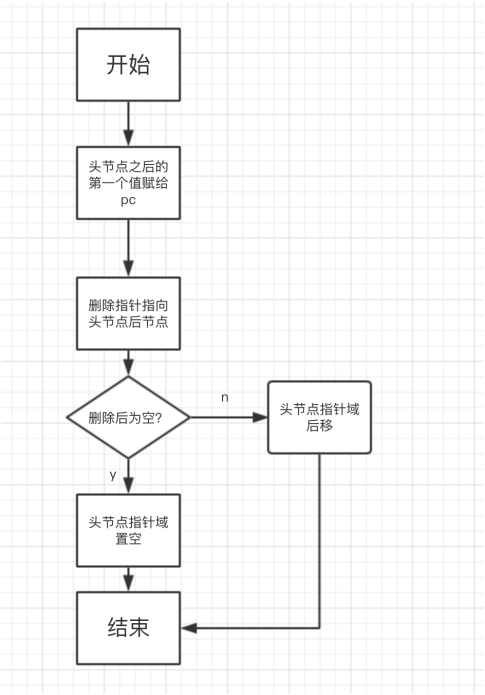


3.1.2 押入功能核心代码

```
void WorkStack::Push(char pc){
    StackNode *newnode=new StackNode;
    newnode->c=pc;
    if(!IsEmpty()){newnode->next=first->next;}
    first->next=newnode;
}
```

3.2 弹出功能的实现

3.1.1 弹出功能流程图

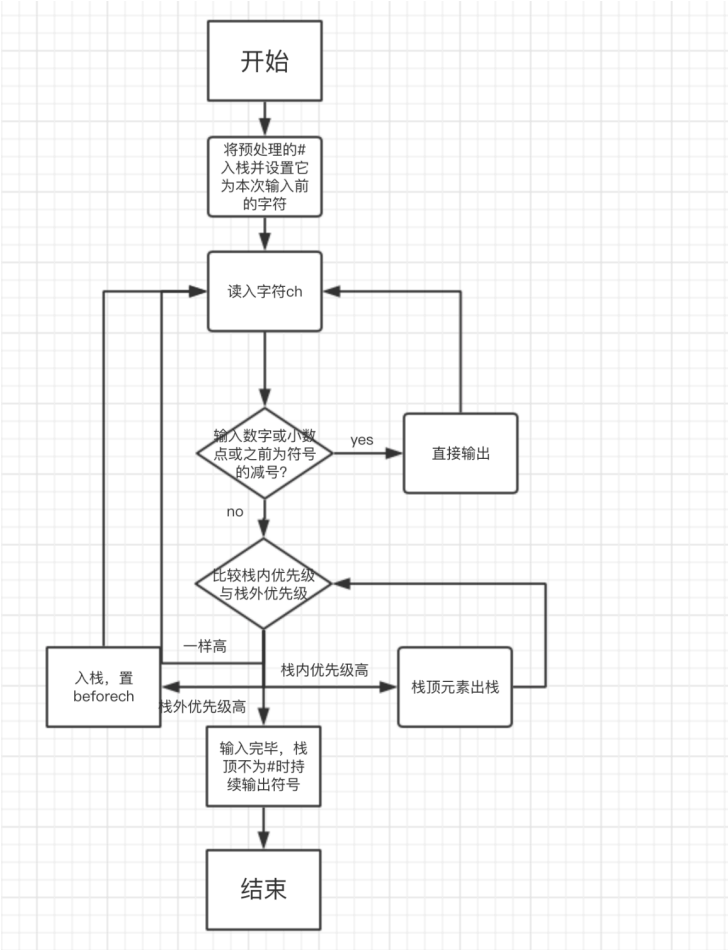


3.1.2 弹出功能核心代码

```
void WorkStack::Pop(char &pc){
    if(!IsEmpty()){pc=first->next->c;
        StackNode *del=first->next;
        if(first->next->next!=NULL)first->next=first->next->next;
        else first->next=NULL;
        delete del;
    }
}
```

3.3 表达式转换功能的实现

3.3.1 表达式转换功能流程图



3.3.2 主程序核心代码

```
int main(){WorkStack s;
    char ch,op,beforech='#';
    s.Push('#');
    cin.get(ch);

    while(!s.IsEmpty()&&ch!='\n')
    {if( issdigit(ch)||(ch=='-'&&!isdigit(beforech))) {cout<<ch;beforech=ch;cin.get(ch);if( !issdigit(ch))cout<<" ";}
    else{
        if(isp(s.GetTop())<icp(ch)){s.Push(ch);beforech=ch;cin.get(ch);}
        else if(isp(s.GetTop())>icp(ch)){s.Pop(op);cout<<op<<" ";}
        else{s.Pop(op);
            if(op=='(')beforech=ch;cin.get(ch);
        }


    }
    }
    while(s.GetTop()!='#'){s.Pop(op);cout<<op<<" ";}
```

4.测试

4.1 常规测试

4.1.1 正常测试六种运算符

测试用例：2+3*(7-4)+8/4

预期结果：2 3 7 4 - * + 8 4 / +

实验结果：

```
2+3*(7-4)+8/4
2 3 7 4 - * + 8 4 / + Program ended with exit code: 0
```

4.1.2 嵌套括号

测试用例：((2+3)*4-(8+2))/5

预期结果：2 3 + 4 * 8 2 + - 5 /

实验结果：

```
((2+3)*4-(8+2))/5
2 3 + 4 * 8 2 + - 5 / Program ended with exit code:
```

4.1.3 运算数超过一位整数且有非整数

测试用例：1314+25.5*12

预期结果：1314 25.5 12 * +

实验结果：

```
1314+25.5*12
1314 25.5 12 * + Program ended with exit code: 0
```

4.1.4 运算数有正或负号

测试用例：-2*(3)

预期结果：-2 3 *

实验结果：

```
-2*(3)
-2 3 * Program ended with exit code: 0
```

4.1.5 只有一个数字

测试用例：123

预期结果：123

实验结果：

```
123
123 Program ended with exit code: 0
```