项目说明文档

数据结构课程设计

——表达式转换

作 者 姓 名： 王加炜

学 号： 2150265

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

# 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形式。日常使用的算术表达式是采用中缀表示法，即二元运算符位于两个运算数中间。请设计程序将中缀表达式转换成为后缀表达式。

## 1.2功能分析

1.输入说明：输入在一行中给出以空格分隔不同对象的中缀表达式，可包含 +, -, \*, /, -, \*, /以及左右括号，表达式不超过20个字符（不包括空格）。

2.输出说明：在一行中输出转换后的后缀表达式，要求不同对象（运算数， 运算符号）之间以空格分隔，但是结尾不得有多余空格。

# 设计

## 2.1数据结构设计

如上功能分析所述，由于中缀表达式转换为后缀表达式需要涉及到数据成员的比较，而这种比较符合数据先进后出的特点，因此想到采用栈的方式来实现。

## 2.2类结构设计

由于本题要频繁进行数据的插入，删除和比较，所以决定采用链表的存储结构对栈中数据进行存储。链式栈中每一个数据用结构体存储，该结构体包含了数据本身和指向下一个链表的指针。

## 2.3成员与操作设计

### 2.3.1全局变量

typedef map<char, int>mcp;

mcp isp;//栈内优先级

mcp icp;//栈外优先级

### 2.3.2linknode结构体

struct linknode//链式栈的结点

{

char data;

linknode\* link;

linknode(int x, linknode\* ptr = NULL) :link(ptr), data(x) { ; }

};

### 2.3.3linkedstack类

class linkedstack//链式栈

{

protected:

linknode\* top;

public:

linkedstack(linknode\* p = NULL) :top(p) { ; }//构造函数

~linkedstack()//析构函数

{

empty();

}

bool isempty()const;//栈是否为空

void push(char x);//插入栈中

bool pop(char& x);//弹出栈中

void empty();//清空栈中元素

bool gettop(char& x)const;//获取栈顶元素

void calculate();//计算

};

## 2.4系统设计

用户输入一个中缀表达式，将该表达式做calculate运算，根据中缀表达式中的元素和栈内元素优先级的比较而执行进栈或退栈的操作，处理完毕后将后缀表达式进行输出。

# **实现**

## 3.1中缀表达式输入

### 3.1.1文字说明

将中缀表达式输入到formula数组中。

### 3.1.2代码实现

cout << "请输入一个中缀表达式：";

push('#');//先把终止符压入栈中

char formula[22];//公式存储在里面

char ch;

int p = 0;

while ((ch = getchar()) != '\n')

{

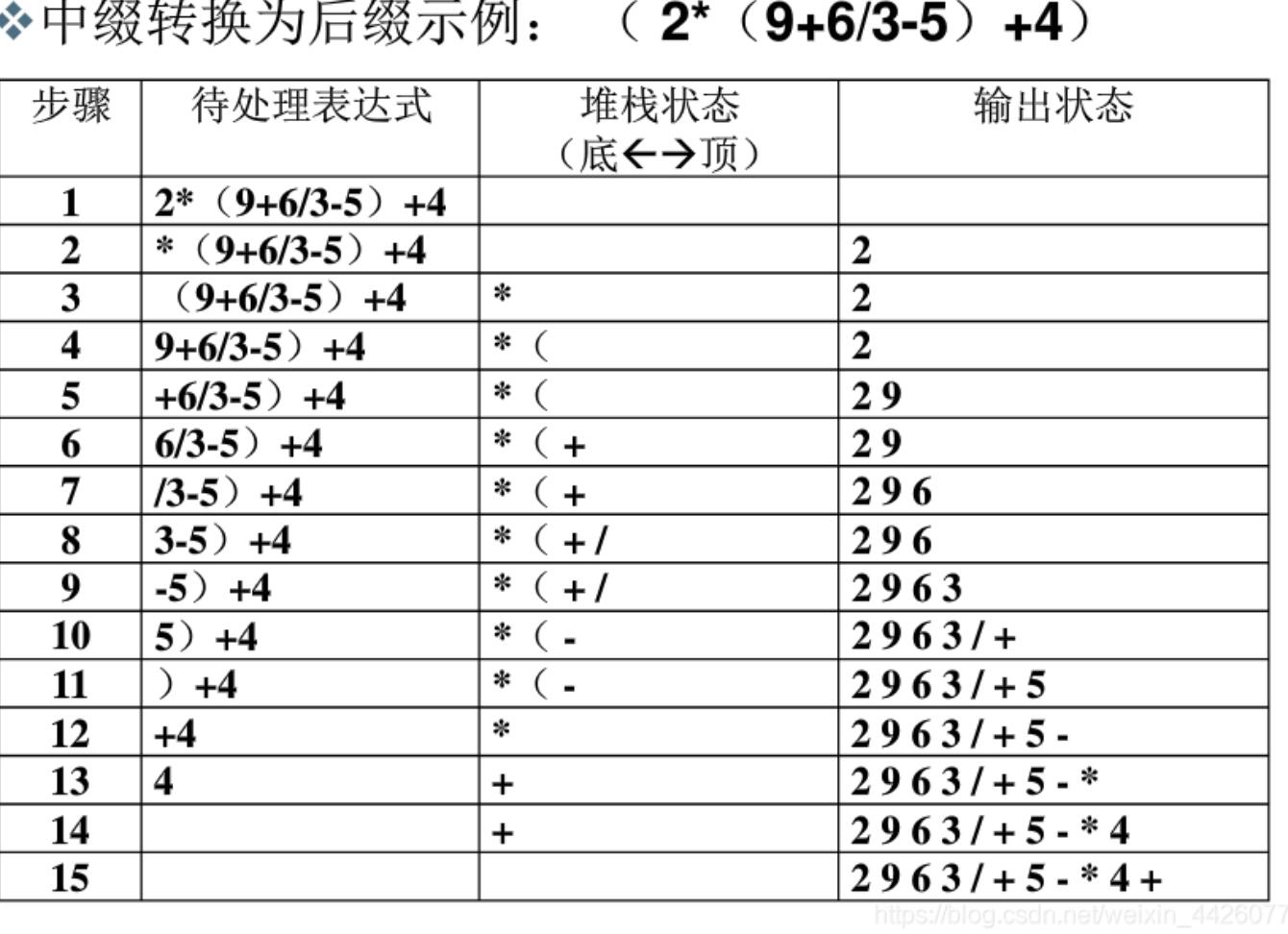
if (ch != 32)//排除空格的影响

formula[p++] = ch;

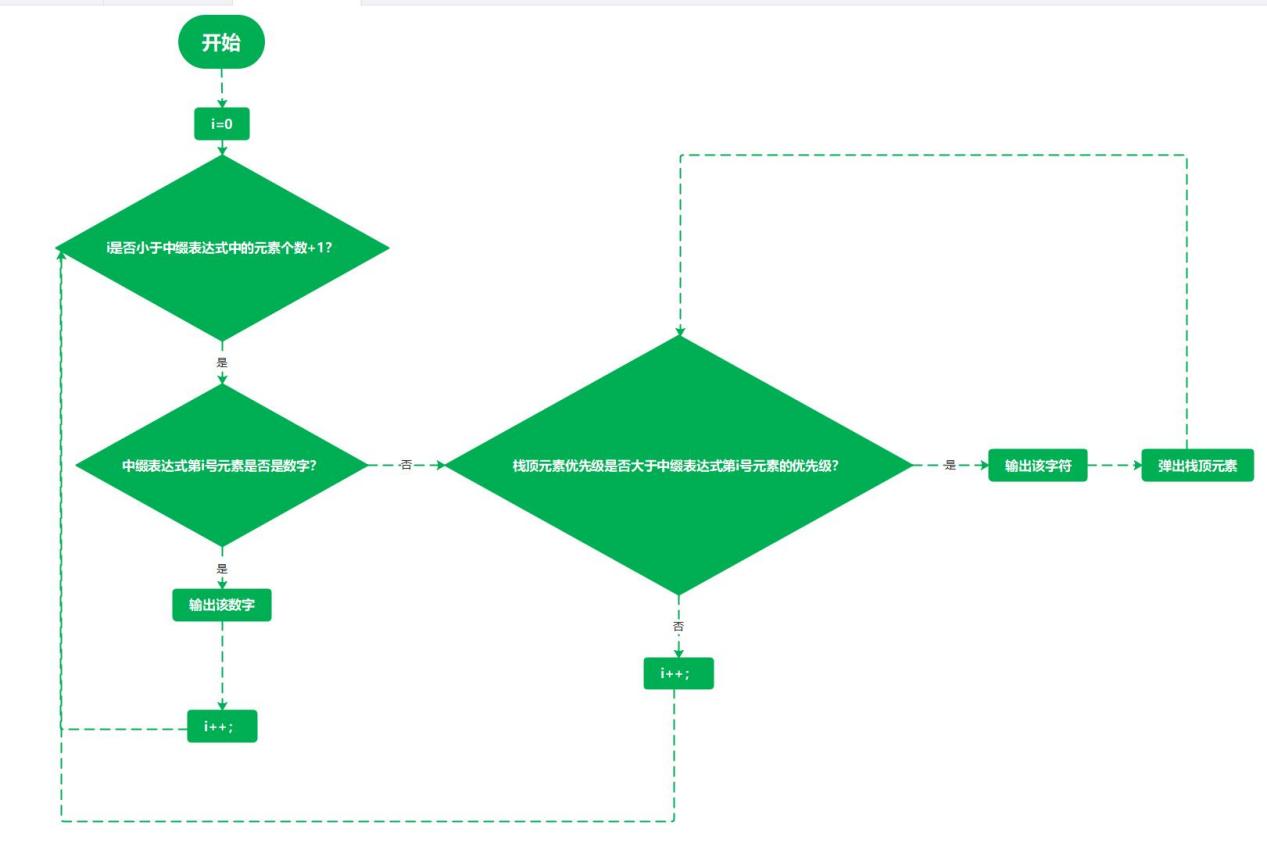
}

## **3.2通过链式栈对中缀表达式进行处理并输出**

### 3.2.1原理示例



### 3.2.2流程图



### 3.2.3文字说明

该函数是实现该项目功能的核心函数。本质上中缀表达式转换成后缀表达式的过程和运算符优先级相关。根据各个运算符号在栈里和栈外的优先级关系，同时借助栈这样一个先进后出的数据结构，可以较为形象且直观地解决这个问题。

由于输入函数要求含空格，所以不能采用cin输入的方式，而要对输入键盘的字符是否是空格进行单独判断。输入之后要在结尾添加一个#表示输入终止符。将该字符串共同存放在名为formula的字符数组当中。

之后开始对于输入进行处理，首先考虑特殊情况。第一种特殊情况是如果运算数包含正负号，那么正负号就不表示运算符号而是正负的意思。这时候要对此做一个特判。如果正负号出现在公式开头或者该符号之前的符号是左括号，都说明该符号作为正负号而非运算符来处理。

从formula数组中的第一个元素开始进行遍历。如果当前搜索的元素不是数字，若当前搜索的元素优先级小于等于栈顶元素的优先级，则持续令栈顶元素退栈，每退一次栈都输出该元素，直到当前栈顶元素的优先级小于当前搜索的元素。最后令当前搜索的元素进栈；若当前搜索到是数字，则输出该数字。上述考虑的是一般情况。不难想到，当左括号在栈中而正好搜索到右括号时，此时只需要将左右括号互相消去即可，并不需要将左右括号进行输出，因此需要对这种情况做一个特殊处理。

若当前搜索到元素是”#“并且栈顶元素是”#“时，说明搜索已经达到尽头，所以终止该循环。

### 3.2.4代码实现

void linkedstack::calculate()

{

cout << "请输入一个中缀表达式：";

push('#');//先把终止符压入栈中

char formula[22];//公式存储在里面

char ch;

int p = 0;

while ((ch = getchar()) != '\n')

{

if (ch != 32)//排除空格的影响

formula[p++] = ch;

}

formula[p] = '#';//公式末尾加一个#表示终止符

cout << "转换后的后缀表达式是：";

for ( int i = 0; i < p + 1; i++)//遍历公式所有的元素

{

int end = 0;

//加号和减号作为符号而非运算符的特判

if (i == 0 && formula[i] == '-')//减号出现在开头

{

cout << "-";

}

else if (i == 0 && formula[i] == '+')//加号出现在开头

{

;

}

else if (formula[i] == '-' && formula[i - 1] == '(')//减号作为符号而非运算 符

{

cout << "-";

}

else if (formula[i] == '+' && formula[i - 1] == '(')//加号作为符号而非运算符

{

;

}

else if (formula[i] == '+' || formula[i] == '-' || formula[i] == '\*' || formula[i] == '/' || formula[i] == '(' || formula[i] == ')' || formula[i] == '#')//遍 历运算符

{

char t;

gettop(t);//取出栈顶元素

while (isp[t] >= icp[formula[i]])//如果当前搜索的元素的优先级小于等于栈顶的 元素，则持续令栈顶元素退栈

{

if (isp[t] == icp[formula[i]] && t == '(')//如果要退栈元素是（，退栈但 不输出，重新读入下一个字符

{

end = 1;

pop(t);

break;//跳出该循环，从下一个字符开始

}

else if (isp[t] == icp[formula[i]] && t == '#')//若到达尾部，则终止循 环

{

break;

}

else

{

//这里做一个if-else分支的目的是让结尾之后没有空格

if (formula[i] == '#' && top->link->data == '#')//判断是否达到输出 的结尾

{

cout << t;

}

else

{

cout << t << " ";

}

pop(t);//弹出栈顶元素

gettop(t);//重新取一次栈顶元素

}

}

//出了循环，表示栈顶元素优先级小于该元素优先级,进栈

if (end == 0)//排除左右括号的情况

push(formula[i]);

}

else//遍历整数和小数，小数可以加小数点

{

if (i == p - 1 || formula[i + 1] == '\*' || formula[i + 1] == '/' || formula[i + 1] == '(' || formula[i + 1] == ')' || formula[i] == '#' || formula[i + 1] == '+' || formula[i + 1] == '-')//判断数字是否连续，若是连续数字， 中间不需要加空格

{

cout << formula[i] << " ";

}

else

{

cout << formula[i];

}

}

}

}

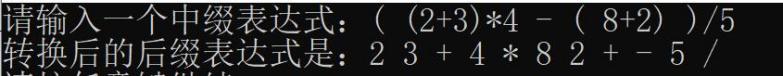
# **测试**

## 4.1功能测试

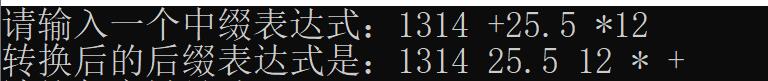
### 4.1.1正常情况：

1

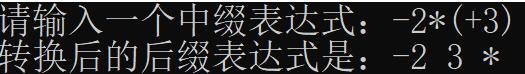
### 4.1.2嵌套括号：



### 4.1.3运算数超过1位整数且有非整数出现：



### 4.1.4运算数有正或负号:



### 4.1.5只有1个数字:

