* **题目：括弧匹配问题**
* **班级：2021211312 姓名：王昊钰 学号：2021211282** 分工： **1,2**
* **班级：2021211312 姓名：杨天宇 学号：2021211281** 分工： **3,4**
* **班级：2021211312 姓名：李志茂 学号：2021211289** 分工： **5,6**

**一、需求分析**

程序用来补全缺失的左括号，将加了括号的表达式视作一个数字，最终使栈中只剩下一个数字，并将其输出。

输入的形式：缺失左括号的表达式

输出形式：补全括号之后的中序表达式

正确的输入：输出补充好括号之后的表达式

错误的输入：输出invalid input

2.概要设计

思路：在遇到右括号时，出栈两个数和一个符号，并在最左边补齐左括号，最后将其当做一个数字入栈。不断循环，直到数字栈中只剩下一个数。

数据结构：stack <string> num,ch;

String a,b,c,d;

主程序的流程：读一个字符，如果是数字入num栈，如果是符号入ch栈，如果是右括号开始处理，不断循环直至num栈中只剩下一个。

调用关系：无

二、概要设计

解决思路：遇到右括号即出栈两个数字与一个符号，在其左端加上一个左括号后，将其作为一个数字重新入数字栈

数据结构类型：stack <string> num,ch;

主程序流程：

1.将所给字符串入栈。数字入num栈，符号入ch栈。

2. 遇到右括号即出栈两个数字与一个符号，在其左端加上一个左括号后，将其作为一个数字重新入数字栈

3.重复1，2步，直到读完字符串

4.输出相应表达式

1. **详细设计**

本实验只需一个函数就可实现全部的操作，核心操作为数字栈和字符栈的使用

（1）栈的声明：

    stack <string> num,ch; //使用string类型的栈完成操作，其中一个为数字栈，另一个为字符栈

1. 数字和运算符入栈：

if(in[i]>='0'&&in[i]<='9') //若此字符为数字

{

a+=in[i];

while(in[i+1]>='0'&&in[i+1]<='9')//若之后若干位字符仍为数字，则所有数字位共同组成一个数

{

a+=in[i+1];

i++;

}

if(in[i]<'0'||in[i]>'9')

i--;

num.push(a);

a.clear(); //清空字符串a

}

else if(in[i]=='+'||in[i]=='-'||in[i]=='\*'||in[i]=='/') //若此字符为加减乘除运算符

{

if(!num.empty()) //若数字栈不为空，则将运算符存入字符栈中，否则视为非法输入

{

c+=in[i];

ch.push(c);

c.clear();

}

else

{

cout << "invalid input";

judge=0;

break;

}

}

1. 右括号入栈（核心操作）：

else if(in[i]==')') //若此字符为右括号，则形成一个表达式

{

if(!num.empty()&&!ch.empty())

{

b+=')';

b=num.top()+b;

num.pop();

b=ch.top()+b;

ch.pop();

if(!num.empty())

{

b=num.top()+b;

num.pop();

b='('+b;

num.push(b);

b.clear();

}

else

{

cout << "invalid input";

judge=0;

break;

}

}

else

{

cout << "invalid input";

judge=0;

break;

}

}

1. **调试分析报告**

遇到的问题：1.未考虑多位数问题，只能实现个位数的读入，随后使用循环语句纠正；

2.对于非法输入的问题考虑不周全，发现输出结果异常后改正。

回顾讨论：主要利用双栈结构完成程序设计

时空复杂度分析：时间复杂度：O(n)，空间复杂度：O(n)：

经验和体会：注意程序对不同输入的普适性

1. **用户使用说明**

编译运行Bracket Matching.cpp，输入一个缺失左括号的表达式，若输入合法，则程序会输出一个补全了左括号的表达式，否则会输出:invalid input

**六、测试结果**

Bracket Matching.cpp测试用例

输入用例：

1+2)\*3-4)\*5-6)))

2+3)\*4+5))

3+5)/6-7)+5))

55+77)\*532-564)-4)+5))

984\*889-4566)/554-861)+459)))

输出用例：

((1+2)\*((3-4)\*(5-6)))

((2+3)\*(4+5))

((3+5)/((6-7)+5))

((55+77)\*(((532-564)-4)+5))

(984\*((889-4566)/((554-861)+459)))