表达式计算

#### 软件工程20级2班 2053300 胡锦晖

# 一、项目概述

## 项目背景

### 前缀表达式

前缀表达式又称波兰表达式，指的是不包含括号，运算符放在两个运算对象的前面，严格从右向左进行（不再考虑运算符的优先规则），所有的计算按运算符出现的顺序进行。

### 中缀表达式

中缀表达式是一个通用的算术或逻辑公式表示方法， 操作符是以中缀形式处于操作数的中间，中缀表达式是人们常用的算术表示方法。与前缀或后缀记法不同的是，中缀记法中括号是必需的。计算过程中必须用括号将操作符和对应的操作数括起来，用于指示运算的次序。

### 后缀表达式

后缀表达式又称逆波兰表达式，指的是不包含括号，运算符放在两个运算对象的后面，与前缀表达式类似，不需要括号指示运算的次序，所有计算严格从左向右进行。值得一提的是，由于大多数计算机采取的栈式内存结构，执行先进后出的顺序，所以后缀表达式是计算机解析算数表达式时常用的方式。

## 项目目标

获取一个中缀算数表达式的输入(包括括号)并创建对应二叉树，并对二叉树进行前序遍历，中序遍历，后续遍历，输出对应的前缀表达式（波兰表达式），中缀表达式和后缀表达式（逆波兰表达式）。

# 二、实现思路

## 2.1 数据的存储结构

本项目使用了两个辅助栈（数据栈和符号栈）来辅助建立表达式二叉树，考虑到问题的规模（输入表达式的长度）无法确定，所以采用链栈实现，每个栈结点结构如下：

文本

描述已自动生成对于要建立的表达式二叉树文本

描述已自动生成，我们采用二叉链表存储，树结点结构如下：

这里为了方便在后续进行二叉树遍历输出，重载了 << 运算符。

此外，还需要两个string类型的变量，一个用来存储输入的中缀表达式，另一个用来存储正在处理的表达式部分。

## 2.2 类的设计及关系

## 2.3 基本思路

本项目的重点是如何将所输入的中缀算数表达式构建为一棵二叉树。联系表达式的计算过程，我们使用两个辅助栈来解决这一问题：

一个数据栈和一个符号栈，数据栈存储数据的类型为一个指向二叉树结点的指针（BiTreeNode<string>\*），指向的是一个运算数结点（例如：“3”）或是一个能够通过运算得到一个运算数的结点（例如：“+”，其中该结点的左右孩子分别指向两个运算数的结点）。符号栈存储数据的类型为string，存放表达式中的双目运算符。

在上述设定下，对输入的表达式从左到右进行分析，数据压入数据栈，运算符压入符号栈，当要入栈的运算符优先级大于等于栈顶运算符时，则以该运算符为根结点，取数据栈栈顶的数据以及下一个要处理的数据为左右孩子建树，并将其压入数据栈。重复执行上述操作，直到整个表达式处理完毕。

最后数据栈和符号栈交替出栈建树，并将结果重新压入数据栈，直到数据栈中只有一个结点且符号栈为空，则表达式二叉树建立完毕，数据栈中的结点即为根结点。

## 2.4 功能实现

### 2.4.1 Stack相关方法的实现

图形用户界面, 文本

描述已自动生成1.入栈操作

文本

描述已自动生成2.出栈操作

文本

描述已自动生成图形用户界面, 文本

描述已自动生成3.empty和top方法

### 2.4.2 对于运算符的处理

这里对于+、-、\*、/的处理与之前【2.3基本思路】中所提到的一致，而对于“(”、“)”的处理来说：

1.遇到“(”直接压入运算符栈，并且为了方便在输出中缀表达式的时候能够正确的输出括号，同时将“(”加入到正在处理的表达式变量中去，这样在下一次进行表达式的数据处理时，就可以将相应的数据带着“(”一起进行后续操作。

文本

描述已自动生成2.遇到“)”时，首先将“）”加入到正在处理的表达式变量中去，如下图所示：

这样在下一次进行表达式的数据处理时，就可以将相应的数据带着“(”一起进行后续操作。符号栈和数据栈交替出栈建立局部表达式二叉树并压入数据栈中，直到符号栈中有一个与之匹配的“(”出栈为止。

图形用户界面, 文本

描述已自动生成还需要说明的是，为了将表达式中每个char类型的运算符转换为string类型，这里采用了string构造函数如下图所示的重载版本：

### 2.4.3 建立局部表达式二叉树

图形用户界面, 文本, 应用程序

描述已自动生成建立局部表达式二叉树的方法即为【2.4.2对于运算符的处理】中多次使用的CreateTree方法，即将数据栈栈顶的两个数据作为左右孩子，符号栈栈顶的运算符作为根结点建树，具体实现如下：

### 2.4.4 输出表达式

在完成表达式二叉树的建立之后，输出只需要对其进行前序、中序和后序遍历即可完成波文本

中度可信度描述已自动生成兰表达式、中缀表达式以及逆波兰表达式的输出：

文本

描述已自动生成需要说明的是，由于处理表达式转换问题的ExpConvert类是继承自二叉树BiTree类的，根据继承关系，BiTree在设计时的遍历算法并不清楚在遍历时需要对结点进行怎样的操作，所以将操作封装为了一个函数指针func进行调用，以前序遍历为例如下图所示：

文本

描述已自动生成而在ExpConvert中我们遍历时对于每个结点进行的操作为格式化输出结点中所存的值，且为了匹配中缀表达式和其他两种表达式的括号问题，所以就有了下面两个回调函数，分别为输出括号和不输出括号的版本：

需要说明的是：这里的函数指针与回调函数借助了C++的function类与Lambda表达式实现，在编译时需要使用std=c++11及以上的编译选项。

### 2.4.5 错误处理

由于本项目的输入是一个string类型的表达式，故在输入部分没有进行错误处理。

在处理表达式并建树的过程中，若检测到表达式中含有除0 - 9、.、+、-、\*、/、(、)、外的其他字符时，均会提示“Create Tree Error！”。此外，在处理数据栈和符号栈时，若最终无法达到数据栈只有一个结点，符号栈为空的状态，则证明输入的表达式不合法，会提示“Stack Empty, top() Error”。

文本

描述已自动生成示例如下：

处理策略是在每次进行弹栈（即top方法）时都对栈是否为空进行检测，若栈为空则抛出一个const char\*类型的异常，之后外部函数在捕获到异常时则输出错误提示并退出程序.

抛出异常：

文本

描述已自动生成捕获异常文本

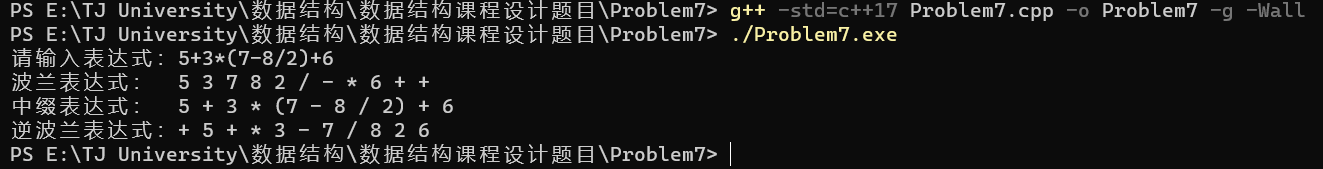
中度可信度描述已自动生成：

# 四、测试结果

## 4.1 Windows平台

操作系统：Windows11 专业版 21H2

编译器：g++ (GCC) 8.1.0

C++标准：c++17

## 4.2 Linux平台

操作系统：CentOS Linux release 8.5.2111

编译器：g++ (GCC) 8.5.0-4

C++标准：c++17

## 4.3 Mac平台

操作系统：macOS Monterey 12.0.1

编译器：clang++ 13.0.0

C++标准：c++17

电脑屏幕的照片上有字

中度可信度描述已自动生成