西安电子科技大学
XIDIAN UNIVERSITY

面向对象程序设计

Object Oriented Programming

实验报告

Experimental Report

学号	20009200713	**姓名**	曾凡浩
班级	**2003052**	**任课教师**	**张淑平**
实验名称	第 4 次实验		
实验学期	**2021 – 2022 学年第 2 学期**		
实验日期	2022 年 5 月 9 日	**实验地点**	
报告成绩			

西安电子科技大学 计算机科学与技术学院

1、 实验目的

使用"类"相关机制来设计并实现一些程序，以熟悉 C++提供的面向对象基本概念和机制，掌握数据抽象的基本手段，用类型上的操作来封装数据结构，为面向对象程序设计奠定基础。

2、 实验环境

操作系统：Window 10

开发工具：Visual Studio2020

3、 实验内容

3.1 题目 1 定义表示二叉树及其结点的类型

Rewrite Tnode from §7.10[7] as a class with constructors, destructors, etc.
Define a tree of Tnodes as a class with constructors, destructors, etc.

3.2 题目 2 定义算术表达式类型

Define a class for analyzing, storing, evaluating, and printing simple arithmetic expressions consisting of integer constants and the operators +, -, *, and /. The public interface should look like this:

```
class Expr {
// …
public:
    Expr(char*);
    double eval();
    void print();
};
```

The string argument for the constructor Expr::Expr() is the expression. The function Expr::eval() returns the value of the expression, and Expr::print() prints a representation of the expression on cout. A program might look like this:

```
Expr x("123.2/(4+123)*43.8");
cout << "x = " << x.eval() << "\ n ";
x.print();
```

Experiment with different ways of printing the expression: fully parenthesized, postfix notation, prefix notation, etc.

4、 数据结构与算法说明

4.1 定义表示二叉树及其结点的类型

模块结构及文件组织设计：

模块 1：主控模块，仅包括文件 main.cpp ，定义了 main()函数。

模块 2：树操作模块，包括以下两个文件：

tree.hpp　　定义了树的/类型，以及操作接口的声明；

tree.cpp　　实现了树相关的操作。

模块 3：结点操作模块，包括以下两个文件：

Tnode.hpp　定义了结点的数据结构/类型，以及操作接口的声明；

Tnode.cpp　实现了对结点的相关操作。

关键数据结构设计：

数据结构 1：定义了树的结点类型 Tnode

```
Class Tnode{
  string  word;
  int  count;
  Tnode* left;
  Tnode* right;
};
```

算法 1.1　int main()

作　用：主控函数，也实现对题目所需其他内容的测试。

参　数：无参数。

返回值：总是返回 0。

计算过程：

(1) 循环读入所有字符串。对于每个字符串，用算法 regist_word 将其登记到树中；

(2)调用算法 print_tree_byorder，采用中序遍历方式，打印树；

(3)调用算法 print_tree，采用先序遍历方式，打印树；

(4)析构函数销毁整个树。

算法 1.2　`void regist_word(Tree *, string&);`

作　用：将输入元素记录到树中

参　数：参数 Tree——传入生成树的根节点。

　　参数 string——所需记录的元素

返回值：无

计算过程：

（1）判断传入数是否为空树，若是空树则传入元素为根节点元素，若不是空树则进行判断；

（2）将当前根结点元素与传入元素进行比较，按不同情况进行递归调用

算法 1.3 **void print_tree(Tnode*);**

作用：将树按照先序打印

参数： Tnode*——需打印树的根节点

返回值：无

计算过程：先序递归打印

算法 1.4 void print_tree_byorder(Tnode*);

作用：将树按照中序打印

参数： Tnode*——需打印树的根节点

返回值：无

计算过程：中序递归打印

算法 1.5 void setword(string);

作用：设置结点的元素值

参数：string——结点元素值

返回值：无

算法：void setLeft(Tnode*);void setRight(Tnode*);同理类似

4.2 定义算术表达式类型

模块结构及文件组织设计：

模块 1：主控模块，仅包括文件 main.cpp ，定义了 main()函数。

模块 2：错误处理模块，包括以下两个文件：

Expr.hpp 定义了语法，词法错误分析操作的接口声明

Expr.cpp 实现了语法，词法错误分析操作的接口

关键数据结构设计：

数据结构 1：定义了枚举类型 token_type {

NUMBER,

PLUS = '+',

MINUS = '-',

MUL = '*',

DIV = '/',

```
    LP = '(',
    RP = ')',
    PRINT = ';',
    ASS = '=',
    ERR_TOKEN,
    NAME,
    END
};
```

算法 2.1　int main

作　用：主控函数，也实现对题目所需其他内容的测试。

参　数：无

返回值：总是返回 0。

计算过程：

通过判断 argc 是否为 1 来确定由键盘输入还是命令行文件输入；

调用算法 get_token（），获取到第一个字符代表什么；

循环调用算法 expr（flase），计算各式的结果；

算法 2.2　token_type get_token()

作　用：对于输入的字符变量，获取对应的实际含义。

参　数：void

返回值：然后一个枚举类型的 token_type.

计算过程：

获取到输入流转化成 char 类型的 ch。

运用 switch--case 来对 ch 进行分类判断返回实际含义；

算法 2.3 double expr(bool)

作　用：在计算过程中完成加减法操作；

参　数：bool

返回值：double

计算过程：

通过递归调用 term 来保留之前计算过的结果；

运用 switch--case 分别进行加减操作；

算法 2.4　double term(bool)

作　用：在计算过程中完成乘除法操作；

参　数：bool

返回值：double

计算过程：

通过递归调用 prim 来保留之前计算过的结果；

通过一个无限循环以及 switch--case 的嵌套进行乘除操作；

算法 2.5　double prim(bool)

作　用：在计算过程中返回数值信息，以及括号等高优先级，有赋值等特殊运算；

参　数：bool

返回值：double

计算过程：

首先判断功能为读取字符信息，还是返回数值信息等操作；

若为前者则调用 get_token（）获取下一字符信息；

若为后者则通过传入的 token_type 类型的变量判断进行什么操作；

算法 2.6　double error(string)

作　用：在计算过程出现错误时进行报错操作；

参　数：string

返回值：double

计算过程：

对传入的 string 信息判断发生什么错误并输出；

5、 测试用例与测试结果

5.1 定义二叉树 Tnode 及其操作

序号	测试数据	先序打印结果	中序打印结果
1	e a f c b	e a b c f	b c a e f
2	a c d z t g	a c d z t g	a c d g t z
3	hjh sd fdf ad z	hjh fdf ad sd z	ad fdf hjh sd z

5.2 定义算术表达式类型

序号	测试数据	输出结果
1	4*3+4-1*4-(2+3)	7
2	A=23+3*4-2	33
3	A+24*2	表达式有误

6、 实验总结

理解如何将项目进行切分，对类的相关机制有了更好的理解，掌握数据抽象的基本手段，用类型上的操作来封装数据结构。