**树实验报告**

Class: 计科20-1 Experiment Date: 2021/12/6

Student ID 1: 20402080240 Name 1: 王夏禹

Student ID 2：20402070113 Name 2：瞿子椋

Student ID3: 20401010121 Name 3: 徐志伟

**实验目的**

1、使用树或二叉树的基本操作进行线性列表操作。

2、处理文件操作。

3、加深对树和二叉树的理解，逐步培养解决实际问题的编程能力。

1. **实验环境**

配备Visual C6.0/CFree的计算机。

试验期4小时

1. **实验内容**

设计一个“自动计算器”如下:

(1)需要计算的表达式存储在TXT文本中的文本文件中;

(2)文本中的每一行都是一个表达式;

(3)表达式包括操作数、加、减、乘、除等操作符和括号;

例如:(34 - 72.3)\* 54.7 - -82.4

（4）“自动计算器”根据输入文件名计算文本文件中的每个表达式，并将结果的每个表达式写入\_out.txt中的原始文件名中，保存记录时应该使用覆盖和方法。每一行的格式为:

表达式=结果。

例如:原始文件为:A1.txt

输出的文件是:A1\_out.txt

A1\_out.txt中的文本格式为:

               ( 34 - 72.3) \* 54.7 - -82.4 = -2177.41

对于所有的计算结果，如果是小数点，在小数点后保留4位。

(5)计算后生成统计文件，其内容为:

执行时间:xxxx-xx-xx hh: mm: ss

表达式总数为:XXX

正确表达式个数为:XXX

错误表达式的个数为:XXX

文件名命名规则:原始文件名:\_log.txt，用追加写方法写入文件。

例如:A1.txt，对应统计文件:A1\_log.txt

特别提醒:★计算过程中需要先将中缀表达式转换为后缀表达式，再将后缀表达式转换为表达式树。最后通过计算得到结果。

1. **重要数据结构**

void InitStack() { //初始化栈

Top = -1;

}

int IsEmpty() { //判断是否为空

return Top == -1;

}

int Push(T X) { //入栈

if (Top + 1 == MaxSize)

return -1;

Data[++Top] = X;

return 0;

}

T Pop() { //出栈

return Data[Top--];

}

T Peek() { //返回栈顶元素

return Data[Top];

}

int GetTop() //返回栈顶元素的值;

{

return Top;

}

T ReturnData(int Top)

{

return Data[Top];

}

private:

int Top;

T Data[MaxSize];

};

//中缀改后缀

Stack<Node> InfixToPost(string Expression)

//后缀表达式计算

double Calculate(TreeNode \*T)

1. **实现思路分析**
2. 读取文件
3. .入栈
4. 形成中缀表达式
5. 变成后缀表达式
6. 计算结果
7. 输出到文件
8. **程序调试问题分析**

无法将文件中的表达式成功的读取到栈中，无法统计时间问题等

1. **实验总结**

在这一次的作业中，我们更加熟悉了栈以及树的使用，回忆了文件的读取和输出的操作，成功的写出了一个简单计算器。

1. **组员分工**

|  |  |  |
| --- | --- | --- |
| **Group division** | | |
| **Member name** | **Work done** | **Completion situation** |
| **王夏禹** | **代码设计** | **已完成** |
| **瞿子椋** | **实验报告** | **已完成** |
| **徐志伟** | **任务分配** | **已完成** |