

浙江大学

Data Structure & Database Technique
Project5
Project Report



2020~2021 春夏学期 2021 年 4 月 25 日

Categories

CHAPTER 1: INTRODUCTION	3
CHAPTER 2: ALGORITHM SPECIFICATION.....	3
<i>计算:递归实现计算</i>	<i>4</i>
CHAPTER3 EXPERIMENT PROCESS.....	4
出现的问题:	5
<i>问题 1.....</i>	<i>5</i>
<i>问题 2:.....</i>	<i>5</i>
<i>问题 3:.....</i>	<i>5</i>
<i>问题 4:.....</i>	<i>6</i>
<i>问题 5:.....</i>	<i>6</i>
<i>问题 6:.....</i>	<i>6</i>
CHAPTER 4: TESTING RESULTS	6

Project5

Chapter 1: Introduction

Background and Our goals :

使用二叉树完成表达式的存储和计算，要求

- 能够实现含有 $()$ 、 $+$ 、 $-$ 、 $*$ 、 $/$ 等运算符的实数表达式计算功能
- 能够处理小数和负数
- 能够处理求余运算符 $(\%)$
- 能够处理乘方 $(^)$
- 能够处理 $\exp, \sin, \cos, \tan, \cotan$ 等常见函数(我觉得这非常难, \tan 存父节点, 就一个子结点?单目运算又不一样非常麻烦)
- 能够打印包含括号的中缀表达式
- 实现变量功能,使表达式树叶节点既可以是数字也可以是变量,在计算前给变量赋值 (选做)

参考课件: [数据结构与算法 \(zju.edu.cn\)](http://zju.edu.cn)

Chapter 2: Algorithm Specification

课件上的方法:

- 处理时，首先找到运算级别最低的运算符“+”或者“-”作为根结点
- 继而确定该根结点的左、右子树结点在表达式串中的范围，例如上面表达式中是 a 和 $(b-c)/d$
- 再在对应的范围内寻找运算级别最低的运算符作为子树的根结点，直到范围内无运算符，则剩余的变量或数为表达式树的叶子

当遇到左括号的时候把左括号入栈，之后跳过所有括号内的符号或者数字，直到遇到第一个右括号，在扫描字符，如果遇到第一个加号或者减号，把该符号作为头结点。然后再把整个字符串以第一个加号或者减号作为中间点，分裂到左右子节点，再重复以上的过程，不断分裂，直到递归的字符串中所有的字符都为数字或者小数点，再通过 `numProcess` 函数判断字符串为 `int` 类型还是 `float` 类型，进行转换，放入子节点中。

对于 `sin`、`cos`、`tan` 等函数，需要通过字符匹配的函数，找到这个函数，在调用 `math.h` 库里面对应的函数，将结果存到子节点中。对于括号的问题，就是当整个字符串的两端是括号的时候，在头结点处设置一个 `flag`，设为 1，如果到时候中缀遍历的时候遇到了节点里的 `flag` 是 1，在左节点输出之前先要输出一个左括号，然后再右节点输出之后，要输出右括号，这样就做到打印包含括号的中缀表达式。对于中缀表达式的输出，只要对整棵树进行中序遍历，就可以得到完整的中缀表达式

对于 `sin`, `con`, `exp` 这些,用 `strcmp` 来比较,一样就返回 0. 所以用与非逻辑, 有 0 出 1

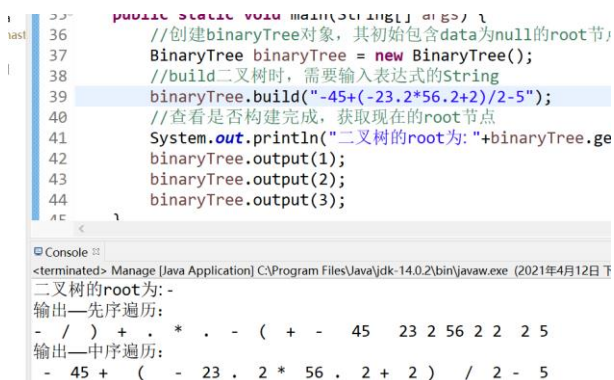
计算:递归实现计算

```
int cal(int root)
{
    int ans = 0;
    int ch = op[root];
    if (!is_alpha2[root]) return ch;
    switch (ch)
    {
        case '+': ans = cal(lch[root]) + cal(rch[root]); break;
        case '-': ans = cal(lch[root]) - cal(rch[root]); break;
        case '*': ans = cal(lch[root]) * cal(rch[root]); break;
        case '/': ans = cal(lch[root]) / cal(rch[root]); break;
    }
    return ans;
}
```

Chapter3 experiment process

测试先行, 先把测试和函数接口写好.

完成能够打印包含括号的中缀表达式. 括号, 小数和负数似乎可以输出但是还不能计算



```
public static void main(String[] args) {
    //创建binaryTree对象, 其初始包含data为null的root节点
    BinaryTree binaryTree = new BinaryTree();
    //build二叉树时, 需要输入表达式的String
    binaryTree.build("-45+(-23.2*56.2+2)/2-5");
    //查看是否构建完成, 获取现在的root节点
    System.out.println("二叉树的root为: "+binaryTree.getRoot());
    binaryTree.output(1);
    binaryTree.output(2);
    binaryTree.output(3);
}
```

Console:

```
<terminated> Manage [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (2021年4月12日 下)
二叉树的root为: -
输出—先序遍历:
- / ) + . * . - ( + - 45 23 2 56 2 2 2 5
输出—中序遍历:
- 45 + ( - 23 . 2 * 56 . 2 + 2 ) / 2 - 5
```

问题: 我感觉这样有点错误, 括号不应该存进去, 或许应该中序遍历的时候加进去, 这也太难了.

负号, 既可以做单目运算符, 也可以做双目运算符. 我目前没有想到什么好的处理办法, 就是每次扫描的时候如果扫描到的是运算符, 就把 flag 置位 1, 如果下一个字符串也是运算符的话而且是 '-' 的话, 就说明后面的字符串表示的是负号的形式.

2.对于括号的处理

括号里面的内容可以看做一个整体, 所以每次遇到左括号的时候, 就把左括号入栈, 括号内的符号跳过, 直到遇到右括号. 再将左括号出栈, 继续扫描整个字符串.

3.对于变量在计算前的提前声明:

感觉这个点实现难度有点大，如果实现了就可以像编译器一样对变量进行定义赋值计算。大概的思路是对 `int float` 这类的字符串进行匹配，然后要找到等号，等号左侧是变量的名称，右侧的所有就是要赋值的变量了。而且要以分号作为语句分格，就可以对语句进行清晰的分割。

解决方法:

- 1.遇到右括号，然后还是照样操作，操作到左括号就停,我没想到怎么处理.
- 2.在里面不存括号，中序遍历如果有左子树，说明不是叶节点，应该输出一个左括号，如果有右子树，说明不是叶节点，应该输出一个右括号,比如下面这样

```
if(isOper(p->right->oper)&&getOperPri(p->right->oper)<=getOperPri(p->oper))
{
    cout<<"(";
    inOrderTraverse(p->right);
    cout<<")";
}
```

出现的问题:

问题 1

Unresolved compilation problem:

你的项目的编译器版本比运行环境 jre 的版本低，会造成这个异常.但是我好像不是这个问题. 我是因为有中文汉字出现.

问题 2:

fatal error C1071: 在注释中遇到意外的文件结束

解决:

现是中文注释的问题，在使用 `/* 中文注释 */` 进行中文注释时，中文前后没有加空格，由于编乱码的问题，导致了错误。

加了空格后不再显示错误.

问题 3:

怎么输入 `char*`?

将指针转换为 `char const *` 类型

```
char const* pch = "abc";
```

文献里说是最为正确，避免错误的方法

但是我们这里要修改 `char*` 所以需要强制转换.

将字符串强制转换为 `char*` 类型

```
char *pch = (char*)"abc"
```

文献翻译为：由于 C++11 并没有取消强制类型转换，这样也可以正常编译。但这样和隐式类型转换

本质上是相同的，依旧有可能发生所谓错误。建议使用例一的方法。

问题 4:

error LNK2019: 无法解析的外部符号 "char * __cdecl left(char *,char *,int)" (?left@@YAPADPAD0H@Z) ，该符号在函数 "int __cdecl isNumber(char *)" (?isNumber@@YAHPAD@Z) 中被引用

可能的原因:

c 语言运行时找不到适当的程序入口函数.

一般，若定义在 .h 头文件里的函数，如果不是 static 类型，那么就会遇到这个无法解析的链接错误，改成 static 后，就行了；

但是，改成 static 可能又报错，说未定义头文件 myself 中的函数，这时候，直接 include myself.cpp ，不要头文件了，就行了

是因为同一个 cpp 文件中函数顺序不对。换一下顺序就对了，不知道为啥 h 文件声明了还是会顺序不对.

问题 5:

warning C4715: “calculate”：不是所有的控件路径都返回值

产生原因：带返回值的函数在最后没有 return x。 条件分支没有覆盖所有选项.

问题 6:

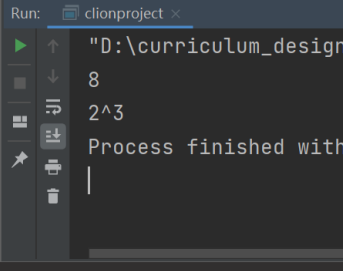
本地函数定义是非法的，此行有一个“{”没有匹配项

原因：有可能是前面少了一个大括号，导致你后面定义函数时被前面的函数包了进去！

刚好出错的两个函数定义的上方有大量用/*。。。*/框起来的注释就猜会不会和/*。。。*/对应的字符什么有关，就把程序中乱七八糟的用/*。。。*/框起来的注释全都删掉，然后就好了。

Chapter 4: Testing Results

test case	Correct answer	Actual behavior of my program
-----------	----------------	-------------------------------

Sin(0.5233) sin 测试	$\sin(\pi/6) = 1/2$	<pre> struct Node* root; root = (struct Node*) malloc(sizeof(struct Node)); char *s = (char*)"sin(0.5233333)"; //char *s = (char*)"e"; //char *s = (char*)"2^"; //char *s = (char*)"1%2-2+3*(4-5)"; //char *s = (char*)"1"; </pre> 
Exp(2) exp 测试	7.38906	<pre> root = (struct Node*) malloc(sizeof(struct Node)); //char *s = (char*)"sin0.2"; char *s = (char*)"exp(2)"; //char *s = (char*)"1%2-2+3*(4-5)"; STRProcess(s,root); double result =calculate(root); cout << result <<endl; displayInOrder(root); </pre> 
1%2-2+3*(4-5) = -4 %和括号测试	-4	
2^3 幂测试	8	<pre> t main() { struct Node* root; root = (struct Node*) malloc(sizeof(struct Node)); //char *s = (char*)"sin0.2"; //char *s = (char*)"exp(2)"; char *s = (char*)"2^3"; //char *s = (char*)"1%2-2+3*(4-5)"; STRProcess(s,root); double result =calculate(root); cout << result <<endl; displayInOrder(root); return 0; } </pre> 
-2+3/1.5-(10*3)+40 负数测试	10	<pre> char *s = (char*)" -2+3/1.5-(10*3)+40"; </pre> 