哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称： 数据结构与算法设计

课程类型：必修

实验项目名称：AvlTree

实验题目： 高度平衡的二叉搜索树的应用

班级：1503101

学号：1150310116

姓名：李博

|  |  |  |
| --- | --- | --- |
| 设计成绩 | 报告成绩 | 指导老师 |
|  |  |  |

一、实验目的

本实验通过编写程序实现AVL 存储结构的建立（插入）、删除、查找算法，加深对此种数据结构的尤其是单双旋转提根的技巧的掌握。

二、实验要求及实验环境

实验要求：

1.设计AVL 的左右链存储结构；

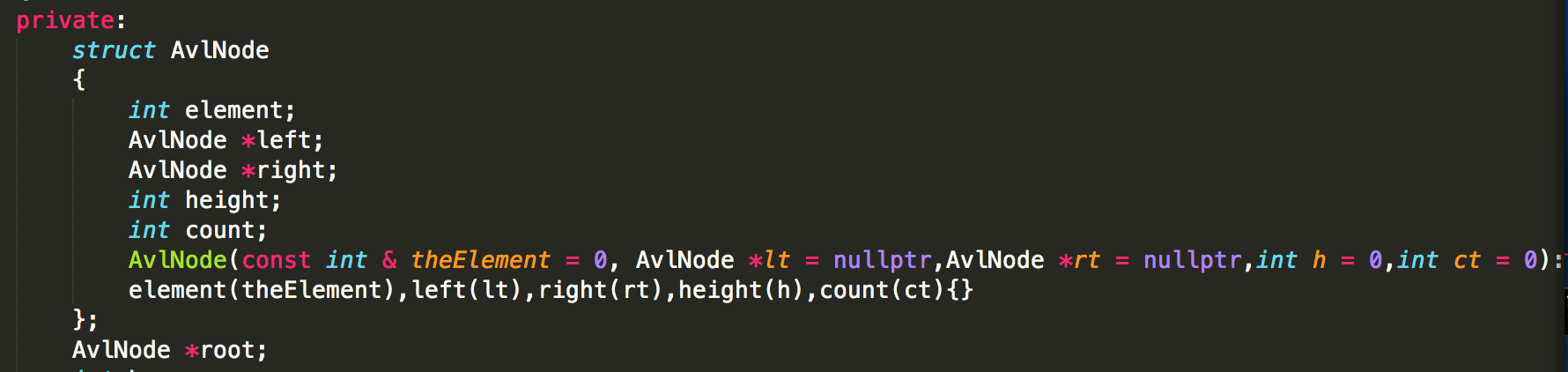
2.实现AVL 左右链存储结构上的插入（建立）、删除、查找和排序算法。

3.测试数据以文件形式保存，能反映插入和删除操作的四种旋转，并输出相应的结果。

三、设计思想（本程序中的用到的所有数据类型的定义，主程序的流程图及各程序模块之间的调用关系）

1. 逻辑设计

存储结构的定义：



在自己实现的avl类里面，结点的定义为avlnode，其中avlnode是典型的树结点的定义方式，包含元素，左右儿子指针。在avlnode中由于需要执行旋转操作，需要之前知道avlnode的高度，所以定义了int类型的height存高度，count是用于执行lazyremove所需要的，表示结点出现的次数。

1. 物理设计
   1. int height(AvlNode \*t)

返回当前结点的高度

* 1. void rotateWithLeftChild(AvlNode \* &k2)

左旋提左儿子为根节点

* 1. void rotateWithRightChild(AvlNode \* &k1)

右旋提右儿子为根节点

* 1. void doubleWithLeftChild(AvlNode \* &k1)

针对LR型树修正，先右旋拐点，在左旋头结点

* 1. void doubleWithRightChild(AvlNode \* &k1)

针对RL型树修正，先左旋拐点，在右旋头结点

* 1. void insert(const int &x,AvlNode \* &t)

对树根结点插入x元素，自动调整高度。

* 1. void FindParent(AvlNode \* &t,AvlNode \*root)

从根节点向下找到t结点的父亲。

* 1. void Remove(int x,AvlNode \* &t)

在树种删除x元素

* 1. void makeEmpty(AvlNode \* &t)

清空整颗树。

四、测试结果

1.测试输入信息

无输入，程序自带插入结点。

2.测试输出信息

\*\*\*\*\*\* Now it's a sample of serial insertion \*\*\*\*\*\*\*\*

5

1 8

7 9

\*\*\*\*\*\*\*\*\* Now it's a sample of removal \*\*\*\*\*\*\*\*\*\*

3

1 7

0 2 5 8

4 6 9

\*\*\*\*\*\*\*\*\* After removed 3 the tree is \*\*\*\*\*\*\*\*\*\*

4

1 7

0 2 5 8

6 9

\*\*\*\*\*\*\*\*\* After removed 6 the tree is \*\*\*\*\*\*\*\*\*\*

4

1 7

0 2 5 8

9

\*\*\*\*\*\* Now it's a sample of single left rotation \*\*\*\*\*\*\*\*

\*\*\*\*\*\* Aftre inserted the tree is \*\*\*\*\*\*

3

2

1

\*\*\*\*\*\* Aftre rotated the tree is \*\*\*\*\*\*\*

2

1 3

\*\*\*\*\*\* Now it's a sample of single right rotation \*\*\*\*\*\*\*\*

\*\*\*\*\*\* Aftre inserted the tree is \*\*\*\*\*\*

1

2

3

\*\*\*\*\*\* Aftre rotated the tree is \*\*\*\*\*\*\*

2

1 3

\*\*\*\*\*\* Now it's a sample of double LR rotation \*\*\*\*\*\*\*\*

\*\*\*\*\*\* Aftre inserted the tree is \*\*\*\*\*\*

4

1 8

5

6

\*\*\*\*\*\* Aftre rotated the tree is \*\*\*\*\*\*\*

4

1 6

5 8

\*\*\*\*\*\* Now it's a sample of double RL rotation \*\*\*\*\*\*\*\*

\*\*\*\*\*\* Aftre inserted the tree is \*\*\*\*\*\*

6

2 9

4

3

\*\*\*\*\*\* Aftre rotated the tree is \*\*\*\*\*\*\*

6

3 9

2 4

五、系统不足与经验体会

系统不足：这个项目完成的比较好，没有什么不足的，为了展示给老师及助教，添加了一个BST实现的mirrortree，多花了一点时间，不过也锻炼了自己。

心得体会：在完成remove这个函数的时候，由于删除的时候会将排序后最近的结点去代替被删除的结点，这样就会导致这棵树暂时存在两个相同的结点，之后在找寻待删除节点的父亲时，如果按照元素来匹配父亲，就会出现错误，所以之后改成了按照地址值，也就是说直接比较指针是否相等的方法来实现FindParent这个函数。这点bug十分的隐蔽，在当时调试的时候很不容易想到，只有自己手动展开节点指针，一步步跟随代码去debug最终才发现这个细节之处。

六、附录：源代码（带注释）

//见文件