姚皓天 (2013011515)

t本原3

概述

原理

操作说明

性序设1

概要设计

详细设计

Student 4

RBTree 类

HashTable 3

收获

特色

附: 红黑树高 度的证明

数据结构大作业报告 红黑树

姚皓天 (2013011515)

2014年12月

Student 奕 RBTree 类 HashTable

设计心得 收获

附: 红黑树? 度的证明

基本原理

概述

本程序是由 Microsoft Visual Studio 2012 创建,目标框架为.NET Framework 4.5。程序实现了红黑树数据结构的可视化,实现了对学生成绩信息的输入输出以及检索的功能。此外,实现了 Hash 表,用于管理学生的姓名和学号数据。

原理

首先实现了将学生封装为 Student 类,接着封装为 Node 类,并在基础上创建红黑树,RBTree 类。接着创建了 HashTable<T> 模板,实现了 Hash 表的功能。最后,在.NET 框架下,完成图形界面的部署。

姚皓天 (2013011515)

概述 原理 界面

操作说明程序设计

概要设计详细设计

Student 类 RBTree 类

RBTree 类 HashTable 3

设计心

特色 7/4 // 102

附: 红黑树高 度的证明

界面

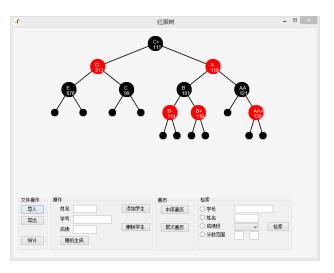


Figure: 界面

姚皓天 (2013011515)

本原野

概述 原理

操作说明

程序设计

Income has a second

概要以

Student

RBTree 类

HashTable

设计心

収获 特色

附: 红黑树? 度的证明

操作说明

文件操作 导入,导出按钮,可以从文本文件导入数据,或 者导出到文本文件。统计信息按键可以显示实 时的统计信息。

添加删除 输入学生的信息,就可以添加学生的信息,同时, 上方的界面将同步绘制响应的红黑树可视化界 面。输入学生的学号,就可以删除对应的学生。

可视化 点击节点,可以在弹出的对话框中查看节点的 信息,同时在对话框中选中的条目信息可以回 填到主界面中。

遍历可以实现中序遍历和层次遍历两种方式的遍历。

检索 可以选择通过,姓名,学号,成绩段和分数区间 四种不同的方式来检索学生成绩信息。

姚皓天 (2013011515)

基本原理

程序设计

概要设计 详细设计

Student 类 RBTree 类

设计心 收获

附: 红黑树高

程序设计

概要设计

首先实现了将学生封装为 Student 类,接着封装为 Node 类,并在基础上创建红黑树,RBTree 类。接着创建了 Hash Table < T > 模板,实现了 Hash 表的功能。最后,在.NET 框架下,完成图形界面的部署。



Figure: 类图

姚皓天 (2013011515)

ま本原モ

概述 原理 以而

**** \ \ \

程序设

概要设计

Student 类

RBTree 类

HashTable

收获

附: 红黑树和

Student 类



Figure: Student 类

枚举型 TypeRate 描述成绩段。 Student 类封装了学生的属性,并重载了 operator < 和 opeartor > 通过分数段来实现对学生的比较,便于在红黑树 中进行操作。

数据结构大作业报告 姚皓天 (2013011515)

原理 界面 操作说明

程序设计

详细设计 Student 类

RBTree 类

设计心

收获 特色

附: 红黑树高 度的证明

RBTree 类

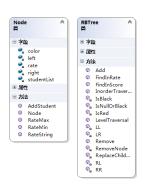


Figure: 红黑树的实现

Node 为红黑树中的每一个节点,包含的字段有颜色,关键码,左右子树,以及一个线性表用于保存数据。

RBTree 实现了红黑树的插入,删除,查找,遍历等方法。其中树的旋转,换底作为私有方法,用于红黑树的维护。

姚皓天 (2013011515)

基本原理

概述 原理

7H ->-> 1H

相要设计

详细设计

Student

HashTable 巻

.....

收获

特色

附: 红黑树高 度的证明



Figure: HashTable 的实现

Hash Table 用泛型编写,实现添加,删除以及查找的功能。 Hash 函数的实现,查找匹配的方法使用委托编写,在实例化 时实现。 姚皓天 (2013011515)

本原理

概述 原理 界面 操作iou

程序设计

概要设计 详细设计 Student 类 RBTree 类

RBTree 类 HashTable

収択化收获

特色

附: 红黑树 度的证明

设计心得

收获

这次大作业抛弃了年代久远的 MFC 框架,在全新的.NET Framework 4.5 框架下完成了此处程序的编写。主要收获是熟悉了.NET Framework 以及编程语言 C‡。

特色

- 程序界面简洁。
- 文本输入正则匹配:

```
Regex regex = new Regex(@"^[\dX]+\z");
if (!regex.IsMatch(id))
{
    throw(new Exception("ID输入不正确"));
}
```

姚皓天 (2013011515)

操作说明

概要设计

RBTree 类

特色

版本库

https://github.com/yht1995/RBTree.git



姚皓天 (2013011515)

概要设计

RBTree 类

附: 红黑树高 度的证明

附: 红黑树高度的证明

求证:包含 n 个内部节点的红黑树的高度是 O(log(n))。 定义: $h(v) = 以节点 v 为根的子树的高度。 <math>bh(v) = \mathcal{K} v$ 到 子树中任何叶子的黑色节点的数目 (如果 v 是黑色则不计数 它)(也叫做黑色高度)。

附: 红黑树高 度的证明 引理: 以节点 v 为根的子树有至少 $2^{bh(v)}$ -1 个内部节点。 归纳基础: h(v)=0 时,bh(v)-0,没有内部节点, $2^0-1=0$,结论成立。

归纳假设: 若 h(v) = k 的 v 有 $2^{bh(v)} - 1$ 个内部节点,则 h(v') = k + 1 的 v' 有 $2^{bh(v')} - 1$ 个内部节点。

因为 v' 有 h(v') > 0 所以它是个内部节点。同样的它有黑色高度要么是 bh(v') 要么是 bh(v')-1 的两个儿子。它的内部节点数为

$$2^{bh(v')-1}-1+2^{bh(v')-1}-1+1=2^{bh(v')}-1$$

结论证明: 因为在从根到叶子的任何路径上至少有一半的节点是黑色 (根据红黑树属性 4),根的黑色高度至少是 h(root)/2。通过引理我们得到:

$$\textit{n} \geq 2^{\frac{\textit{h(root)}}{2}} - 1 \Leftrightarrow \textit{log}(\textit{n} + 1) \geq \frac{\textit{h(root)}}{2} \Leftrightarrow \textit{h(root)} \leq 2\textit{log}(\textit{n} + 1)$$