

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

课程类型：必修

实验项目名称：查找与排序

实验题目：二叉查找树、快排优化、  
线性排序与比较

班级：1603006

学号：1160800724

姓名：池嘉洋

## 二叉搜索树

### 一、实验目的：

实现二叉查找树的插入查找和删除操作，了解二叉查找树的工作原理。

### 二、实验要求及实验环境：

#### 实验要求：

- 1、将用户输入的数据建立成二叉查找树
- 2、使用过程中，用户可以任意查找、插入和删除相应数据
- 3、数据可以有重复出现

#### 实验环境：

寝室+机房+Codeblocks

### 三、设计思想

#### 数据类型定义：

```
typedef struct celltype{
    int data;
    celltype *lchild,*rchild;
    int counts;
}* BST;
```

#### 程序用到的函数及其功能：

```
BST Search(int k,BST F)
//查找二叉搜索树中元素 k 的位置，若没有返回 NULL
void Insert(int R,BST &F)
//将元素 R 插入到二叉搜索树的相应位置
int delete_min(BST &F)
//删除二叉搜索树中的最小元素的节点位置，并返回最小元素
void Delete(int k,BST &F)
//删除二叉搜索树中的元素为 k 的节点
BST creat_BST()
//建立二叉树
void menu()
//函数功能菜单
```

#### 调用关系：

```
BST creat_BST()调用 void Insert(int R,BST &F)
void Delete(int k,BST &F)调用 int delete_min(BST &F)
主函数调用 BST Search(int k,BST F)、void Insert(int R,BST &F)、void
Delete(int k,BST &F)、BST creat_BST()
```

#### 程序功能实现的思想：

查找、插入和删除的思想和课本一致。其中，建立二叉树时，每次从用户输入的数据中读取数据，读取之后调用插入函数，将数据插入，数据读取完毕时，二叉树建立完成。对于元素值重复的情况，在构建节点结构体是增加了一个 count 域，每次

有重复数据进入是，将 count+1，删除时，如果 count 小于 1，则将节点删除，否则将 count 值剪 1。

测试结果：

请输入您选择的功能：

- 1、建立二叉搜索树
- 2、删除节点
- 3、插入节点
- 4、搜索节点
- 0、结束运行

1

请输入您的搜索二叉树数据，输入 0 结束输入

1

2

3

1

0

二叉搜索树建立完成

请输入您选择的功能：

- 1、建立二叉搜索树
- 2、删除节点
- 3、插入节点
- 4、搜索节点
- 0、结束运行

4

请输入要搜索的节点 data 值

1

节点存在于搜索二叉树中其值为：1

请输入您选择的功能：

- 1、建立二叉搜索树
- 2、删除节点
- 3、插入节点
- 4、搜索节点
- 0、结束运行

2

请输入要删除节点的 data 值

1

节点删除完毕

请输入您选择的功能：

- 1、建立二叉搜索树
- 2、删除节点
- 3、插入节点

4、搜索节点

0、结束运行

4

请输入要搜索的节点 data 值

1

节点存在于搜索二叉树中其值为: 1

请输入您选择的功能:

1、建立二叉搜索树

2、删除节点

3、插入节点

4、搜索节点

0、结束运行

2

请输入要删除节点的 data 值

1

节点删除完毕

请输入您选择的功能:

1、建立二叉搜索树

2、删除节点

3、插入节点

4、搜索节点

0、结束运行

4

请输入要搜索的节点 data 值

1

搜索二叉树中不存在此节点

请输入您选择的功能:

1、建立二叉搜索树

2、删除节点

3、插入节点

4、搜索节点

0、结束运行

4

请输入要搜索的节点 data 值

9

搜索二叉树中不存在此节点

请输入您选择的功能:

1、建立二叉搜索树

2、删除节点

3、插入节点

4、搜索节点

0、结束运行

3

请输入要插入的节点 data 值

9

节点插入完毕

请输入您选择的功能：

1、建立二叉搜索树

2、删除节点

3、插入节点

4、搜索节点

0、结束运行

4

请输入要搜索的节点 data 值

9

节点存在于搜索二叉树中其值为：9

请输入您选择的功能：

1、建立二叉搜索树

2、删除节点

3、插入节点

4、搜索节点

0、结束运行

经验体会：

1、在写程序时，尽量提前把需要实现的功能以及需要注意的细节想周到，放置程序写完之后再添加其他功能

2、写一个程序时，首相尽量想好其大体结构，想好结构之后再去写相应的功能会比较简单。

六、附录：源代码

见文件

## 快排优化

### 一、实验目的：

了解快排算法的实现，考虑快排的优化算法，提高快排效率

### 二、实验环境：

寝室+机房+Codeblocks

### 三、设计思想

程序用到的函数及其功能：

```
void Swap( int &i, int &j)
```

```
//交换两个数的值
```

```
int Partition(int list[], int low, int high)
```

```
//快排分块
```

```
void InsertSort(int list[],int low,int high)
```

```
//插入排序法
```

```
void myQuickSort(int list[], int low, int high,int k)
```

```
//优化的快排算法
```

优化的思想：

当数据有一定的规律的时候，每次都用数组的第一个元素作为基准，有可能分块不均匀，造成算法运行速度慢，所以每次选取基准的时候都采用随机数法选取，提高排序的效率。

当数据的数量小于 16 的时候，插入排序的速度远高于其他算法的速度，所以当快排分块的元素小于 16 个时，采用插入排序对数组进行排序。

### 四、测试结果：

```
E:\codeblocks\omice cnjiayang\快排优化\bin\debug\快排优化.exe
随机数的数量为10000000
时间为: 7.372 s.
Process returned 0 (0x0) execution time : 7.562 s
Press any key to continue.
```

### 五：经验体会

1、对于程序的优化算法有很多，应该根据程序数据存在的可能的不同特点选择最优的结果

2、编写优化算法，最根本的要非常熟悉原来算法的结构和原理

### 五、附录：源代码

见目录

## 线性排序的比较

### 一、实验目的：

通过比较不同的线性排序方法，掌握不同线性排序方法的原理，发起其各自的优缺点，及最佳的适应环境

### 二、实验环境

寝室+机房+Codeblocks

### 三、比较的线性排序算法的种类

#### 1、计数排序

#### 2、桶排序

#### 3、基数排序

### 四、各个函数的优缺点

#### 1、计数排序：

计数排序需要占用大量空间，它仅适用于数据比较集中的情况。比如  $[0 \sim 100]$ ， $[10000 \sim 19999]$  这样的数据。

计数排序的基本思想是：对每一个输入的元素  $arr[i]$ ，确定小于  $arr[i]$  的元素个数。

所以可以直接把  $arr[i]$  放到它输出数组中的位置上。假设有 5 个数小于  $arr[i]$ ，所以  $arr[i]$  应该放在数组的第 6 个位置上。

#### 2、桶排序：

桶排序可用于最大最小值相差较大的数据情况，比如

$[9012, 19702, 39867, 68957, 83556, 102456]$ 。

但桶排序要求数据的分布必须均匀，否则可能导致数据都集中到一个桶中。比如  $[104, 150, 123, 132, 20000]$ ，这种数据会导致前 4 个数都集中到同一个桶中。导致桶排序失效。

桶排序的基本思想是：把数组  $arr$  划分为  $n$  个大小相同子区间（桶），每个子区间各自排序，最后合并。

1. 找出待排序数组中的最大值  $max$ 、最小值  $min$

2. 我们使用 动态数组  $bucket$  作为桶，桶里放的元素也用  $bucket$  存储。桶的数量为  $(max-min)/arr.length+1$

3. 遍历数组  $a$ ，计算每个元素  $a[i]$  放的桶

4. 每个桶各自排序

5. 遍历桶数组，把排序好的元素放进输出数组

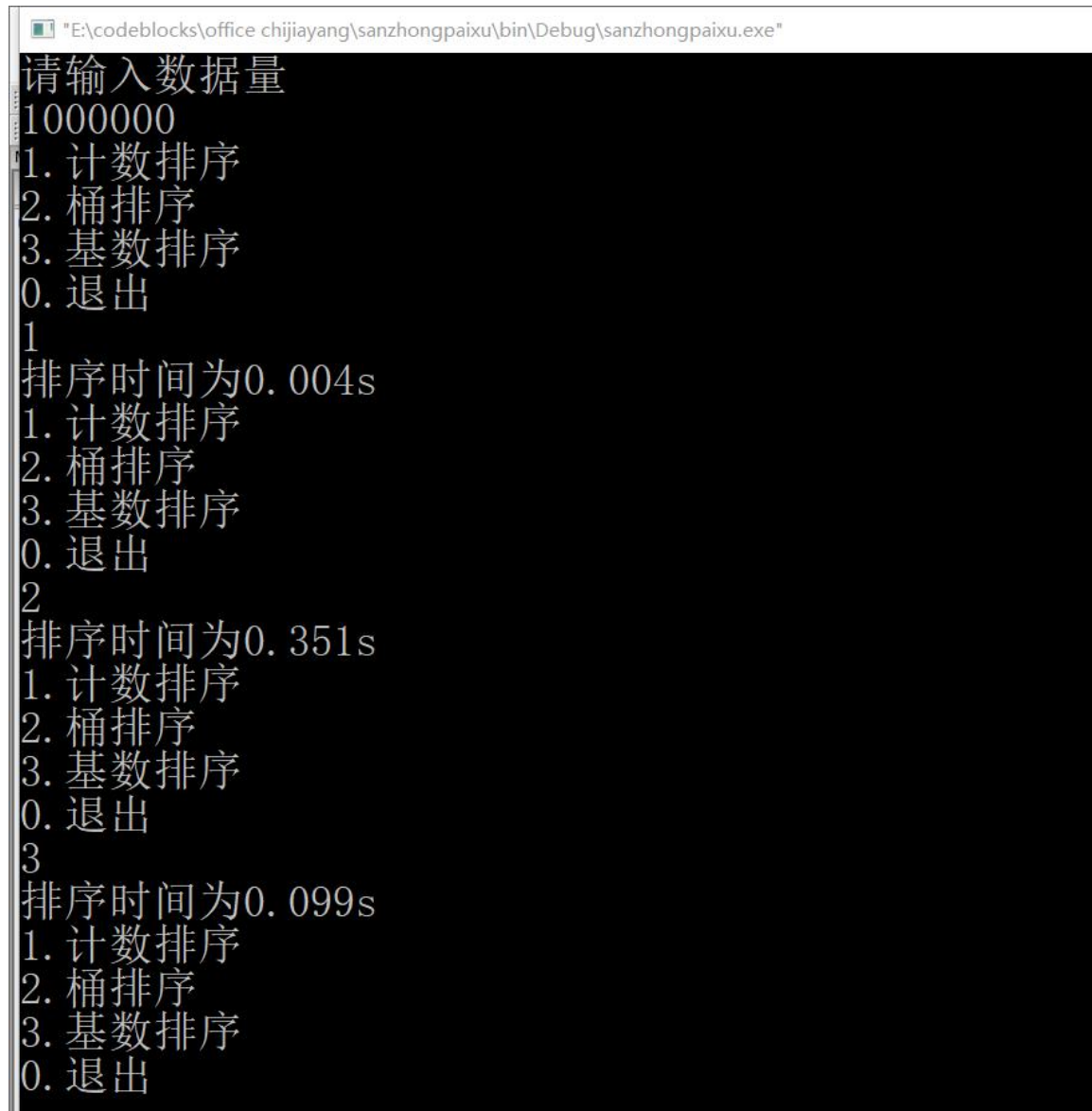
#### 3、基数排序

基数排序必须依赖于另外的排序方法。基数排序的总体思路就是将待排序数据拆分成多个关键字进行排序。

去比较每个位的数值的大小并进行排序，由小到大排序，一些数的数值位数没有达到相应要求是，认为其为 0。

基数排序方法对任一子关键字排序时必须借助于另一种排序方法，而且这种排序方法必须是稳定的。对于多关键字拆分出来的子关键字，它们一定位于 0-9 这个可枚举的范围内，这个范围不大，因此用桶式排序效率非常好。

## 五、测试结果



```
"E:\codeblocks\office chijiyang\sanzhongpaixu\bin\Debug\sanzhongpaixu.exe"
请输入数据量
1000000
1. 计数排序
2. 桶排序
3. 基数排序
0. 退出
1
排序时间为0.004s
1. 计数排序
2. 桶排序
3. 基数排序
0. 退出
2
排序时间为0.351s
1. 计数排序
2. 桶排序
3. 基数排序
0. 退出
3
排序时间为0.099s
1. 计数排序
2. 桶排序
3. 基数排序
0. 退出
```

## 六、系统不足

测试的过程中，只采用了随机数的方式进行测试，没有测试当数据具有一定规律时的情况

## 七、经验体会

每种不同的排序方法，都有其优缺点，也有其各自比较优越的适应环境，应用那种方法要根据具体情况决定。



## 八、附录：源代码

见文件