红黑树和顺序统计树

1. 实验要求

实验 1：实现红黑树的基本算法， 分别对整数 n =20、40、60、80，随机生成 n 个互异的正整数（$K_1, K_2, K_3, \cdots, K_n$ 且$1 \leq Ki \leq 150$），用前 n 个正整数作为节点的关键字，向一棵初始空的红黑树中依次插入 n 个节点，统计算法运行所需时间 ，画出时间曲线。（红黑树采用三叉链表）

实验 2：对上述生成的红黑树，找出树中第 n/4 小的节点和第 n/2 小的节点，并依次删除这两个节点，统计算法运行所需时间，画出时间曲线。

2. 实验环境

编译环境：Microsoft Visual Studio 2015

机器内存：8G

时钟主频：2.5GHz

3. 实验要求

输入输出格式:

a)实验 1 需建立根文件夹，文件夹名称为：学号-project2，在根文件夹下需包括实验报告、和 ex 子文件夹，ex 文件夹下包含 3 个子文件夹：

input 文件夹：存放输入数据

source 文件夹：源程序

output 文件夹：输出数据

b)input：

输入文件中每行一个随机数据，总行数大于等于 80。

分别读取前 20、40、60、80 个正整数进行构建红黑树,插入删除节点的试验

c) output：为每种数据规模建立一个子文件夹，分别为 size20,size40,size60,size80,其输出结果数据导出到其对应子文件下面。

preoreder.txt ：输出构建好的红黑树的前序遍历序列

inorder.txt: 输出构建好的红黑树的中序遍历序列

postorder.txt: 输出构建好的红黑树的后序遍历序列

time1.txt：运行时间效率的数据。测试插入操作构建树的花费的时间，要求每插入 10 个节点测试一下花费的时间，并记录下构建完成所花的总时间

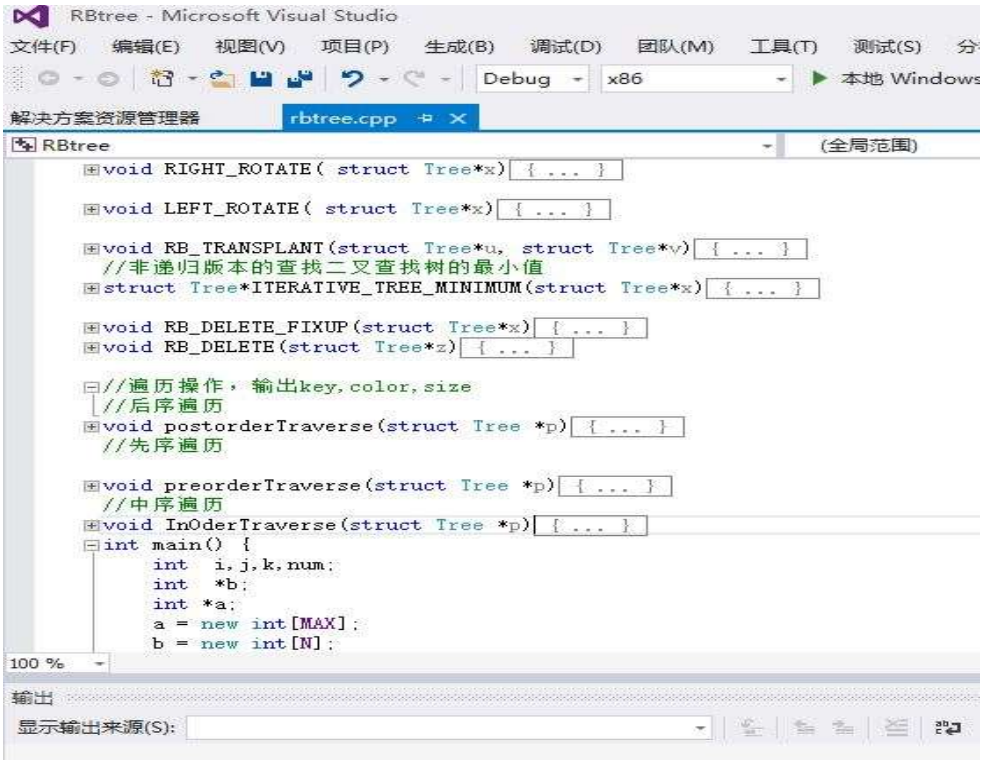
第二个实验输出结果同样是导入到相同的对应子文件夹下面。

delete_data.txt ：输出删除的数据

time2.txt:测试删除掉实验要求删除掉的两个节点所花费的时间，每删除掉一个节点测试一次

delete_inorder.txt: 输出删除后的红黑树的中序遍历子序列

4. 实验过程截图



5. 实验过程

1)使用随机交换法生成 n 个互异的随机数

```
srand((unsigned)time(NULL));
        for (i = 0; i < MAX; i++) a[i] = i+1;//将数组 a 中数字置 1~36
        for (i = 0; i < MAX; i++) {
            j= rand() % MAX ;          //随机选取 j
            num = a[j];  //将 a[i]与 a[j]交换
            a[j] = a[i];
            a[i] = num;
        }
```

2)红黑树结点结构

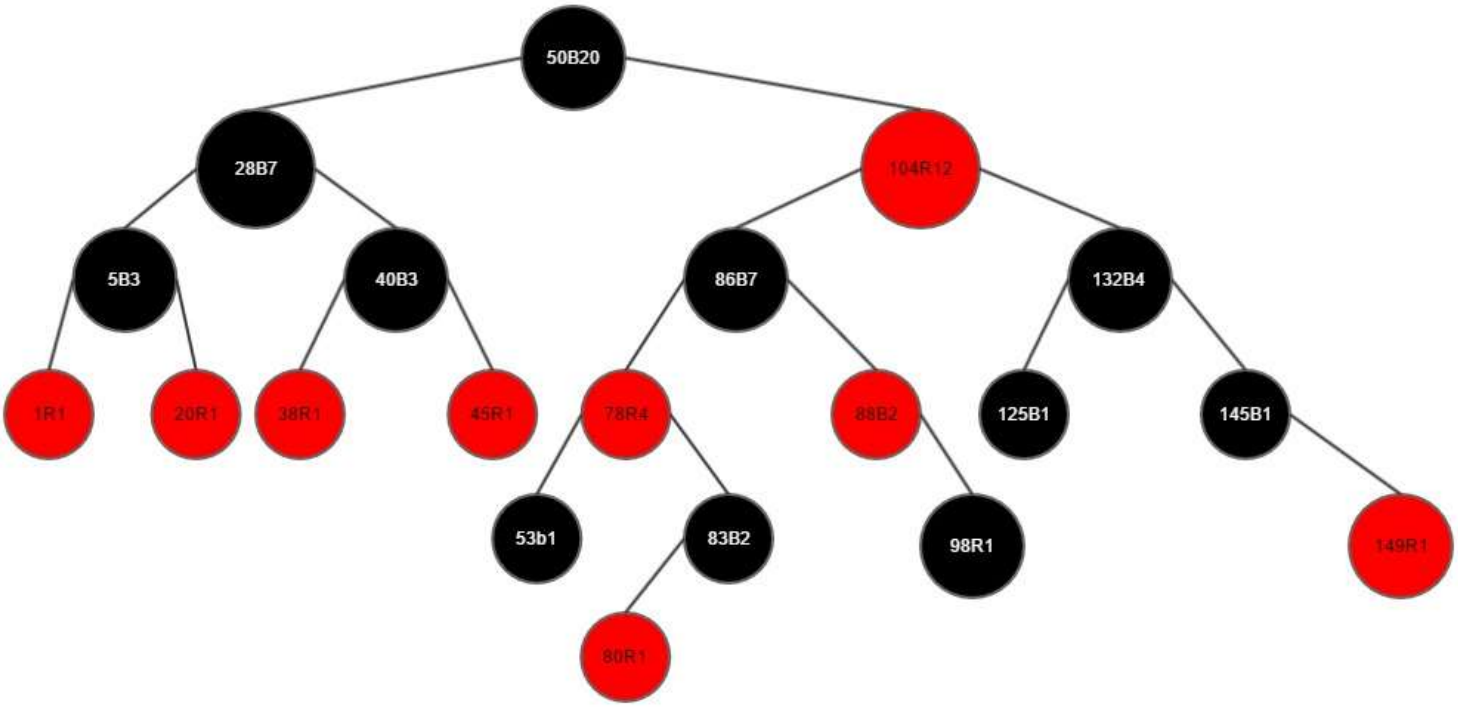
```
    struct Tree{
       int key;  //储存数值
       int color;//储存结点颜色，black 为 1，red 为 0
       int size;//储存结点的秩
       struct Tree *parent;
       struct  Tree *left;
       struct  Tree *right;
    };
    struct Tree*ROOT[4];
    struct Tree*root;
struct Tree*nil = NULL;
 nil->color = BLACK;
  nil->size = 0;//nil 结点颜色为黑，秩为 0
```

3)红黑树左旋，右旋，插入、删除、遍历等操作的实现见 source 文件夹中 rbtree.cpp

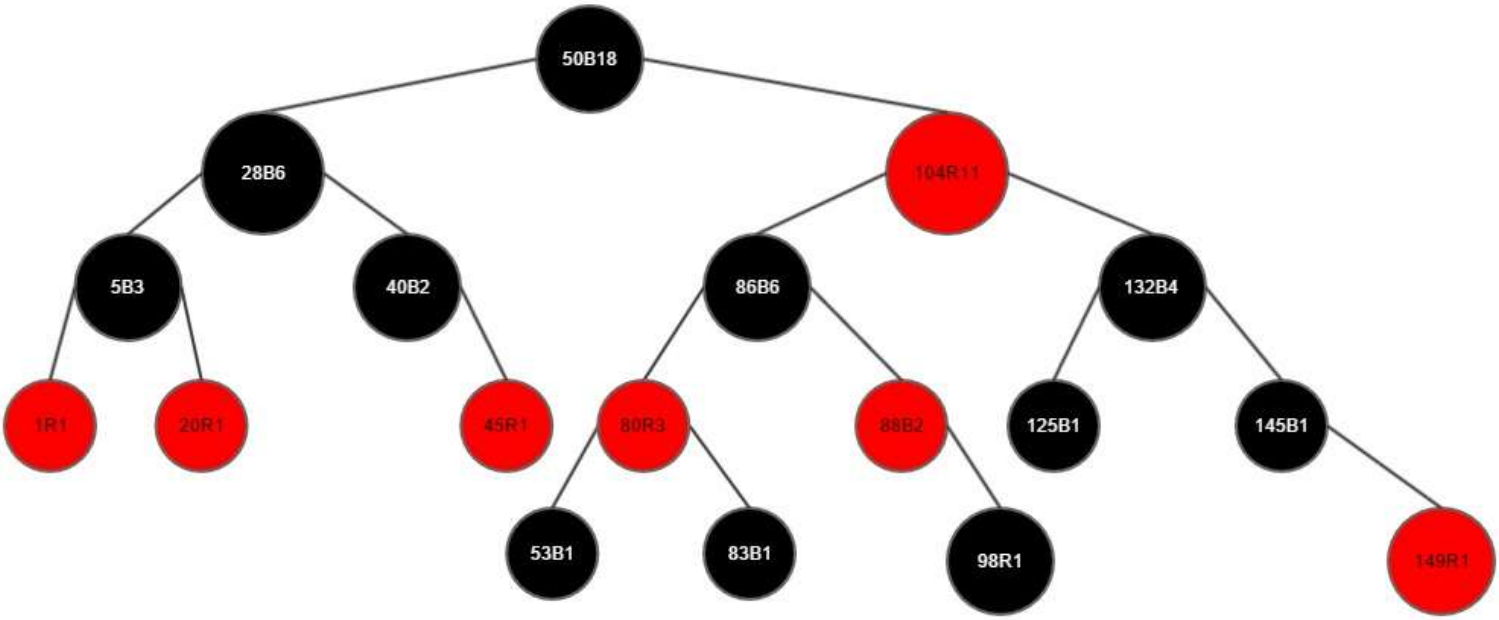
为实现查找任意结点，添加 size 域构成顺序统计树，并添加 size 调整操作。

6. 实验结果与分析

1)20 结点的红黑树结构：结点内容：key color size



2）删除两个结点后的红黑树



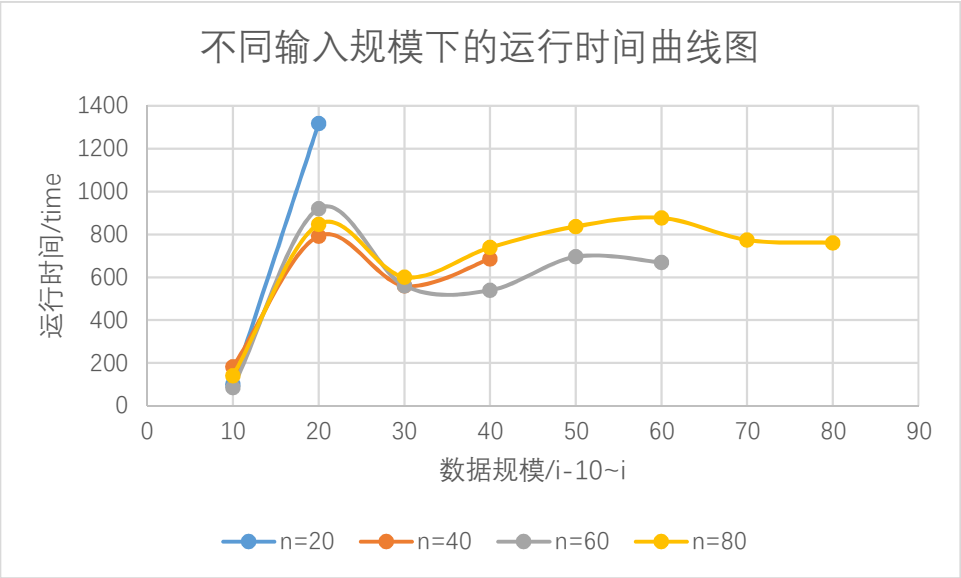
3）算法性能测试

红黑树插入需要 O(lg(n))次，对插入结点后的调整所做的旋转操作不会超过 2 次。删除结点后的调整所做的旋转操作不会操作 3 次，沿树回溯至多 O(lg(n))次。总而言之，红黑树的插入和删除的时间复杂度均为 O(lg(n))。

不同规模下的插入、删除所用时间表格：

	每插入10个结点所用的时间/us								总时间/us	删除结点所用的时间/us)	
数据规模/n	10	20	30	40	50	60	70	80		delete(n/4)	delete(n/2)
20	101	1318							1419	22	18
40	182	792	560	686					2248	42	39
60	85	920	561	539	696	669			3508	57	55
80	141	847	601	739	837	877	774	761	5632	69	103

不同输入规模下的运行时间曲线图：



由于数据规模较小以及单次实验的偶然性导致结果数据误差较大。从图中可以看出，n<=20 时，插入时间变化较大；n>=30 后开始缓慢增长。在 n 较小时，数据分布和黑红树结构对插入时间的影响更大。

7. 实验心得

红黑树是具有着色性质的二叉查找树。由性质节点到 nil 的每一条路径必须包含相同数目的黑色节点可知，只要在一个节点的一侧子树尽可能多的使用红色节点，而另一侧尽可能少甚至不使用红色节点，就可以拉开左右子树的高度差。由 20 结点的红黑树图可以看出，红黑树放弃了追求完全的平衡，而去追求大致上的平衡，在与平衡二叉树的时间复杂度相差不大的情况下，保证每次插入最多只需要三次旋转就能达到平衡，实现起来也更为简单。

对于数据量较小的实验，应可以采取多次实验取均值的方法减少实验误差。