**实验三**

1. **实验要求**

1)实现红黑树左旋操作LEFT-ROTATE(T, x) ,实现右旋操作RIGHTROTATE(T, x)；

2)实现红黑树插入节点的操作 RB-INSERT(T, z)以及插入之后修正为红黑树的算法 RB-INSERT-FIXUP(T, x)；

3)实现红黑树删除节点的操作 RB-DELETE(T, z)以及插入之后修正为红黑书的的算法 RB-DELETE-FIXUP(T, x)；

4)实现按要求数据构建顺序统计树的操作；

5)实现遍历输出构建好的红黑树的操作；

6)实现查找顺序统计树的第i小关键字的操作OS-SELECT(T.root,i)；

7)为了验证第二个实验的正确性，要求编写一个检测程序，使用中位数一章的线性时间的选择算法Select(a,p,r,i)在输入数据找到找到第n/3小的节点和第n/4小节点，与OS-SELECT(T.root,i)的结果delete\_data进行对比检查

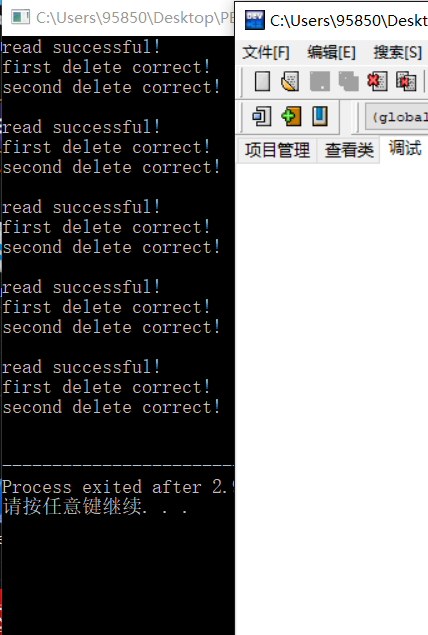
1. **实验环境**

编译环境：Dev C++ 5.11

机器内存：16G

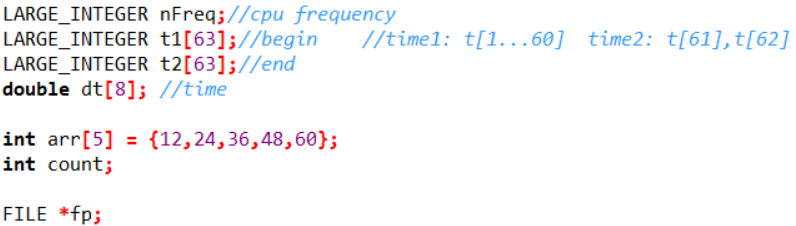
时钟主频：2.20GHz

1. **实验过程**

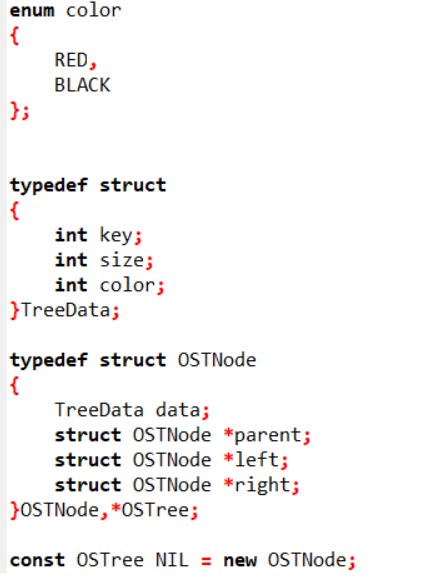


1. **实验关键代码截图（结合文字说明）**

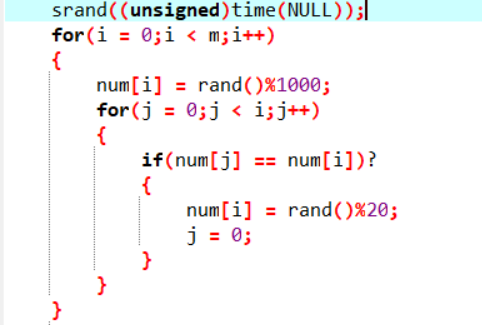
1.重要的全局变量：dt[1...6]为每插入十个结点的运行时间，dt[7]为插入总时间。



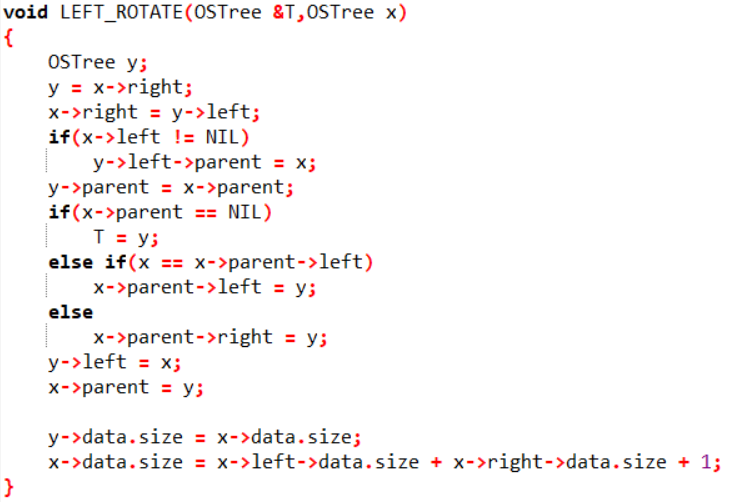
2.顺序统计树结构：



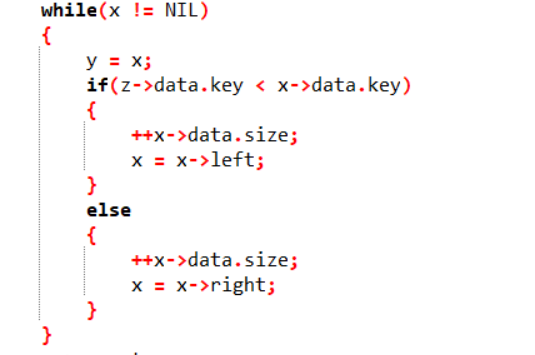
3.随机生成函数



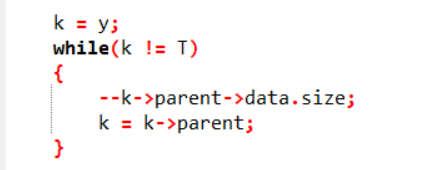
4.左旋，注意维护size即可。



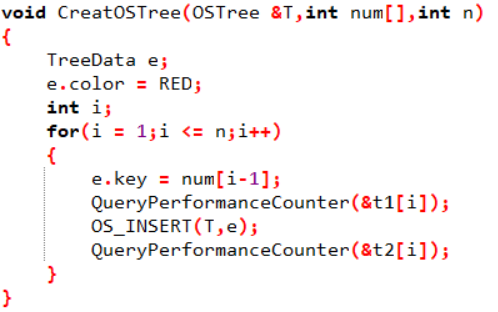
1. INSERT中对size的维护



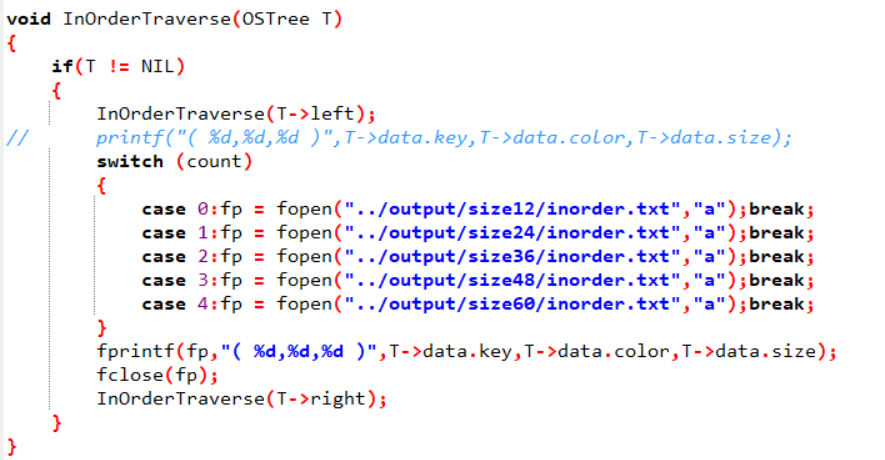
1. DELETE中对SIZE的维护



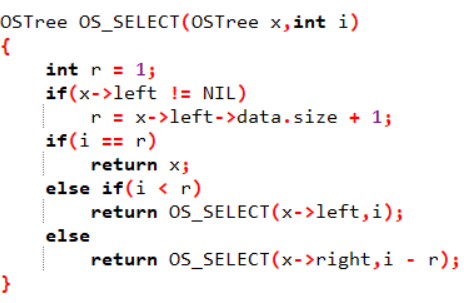
7.建立顺序统计树以及记录插入时间



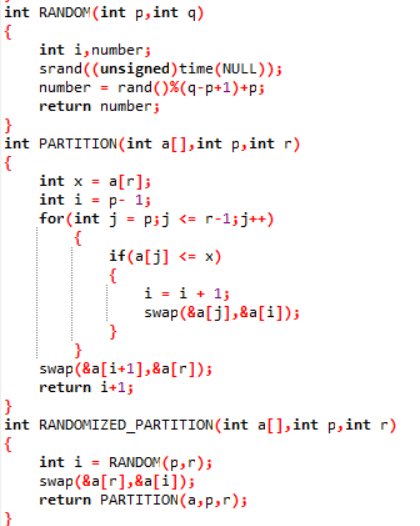
8.中序遍历，另外两个遍历此处略

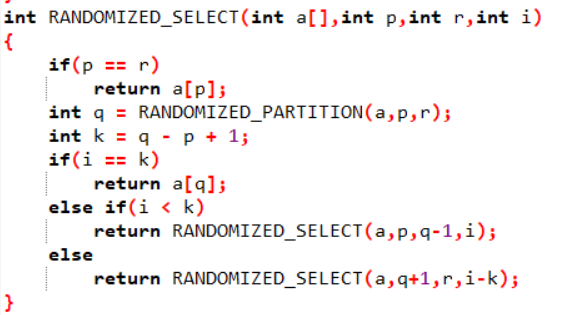


9.顺序统计树中查找第i小的结点

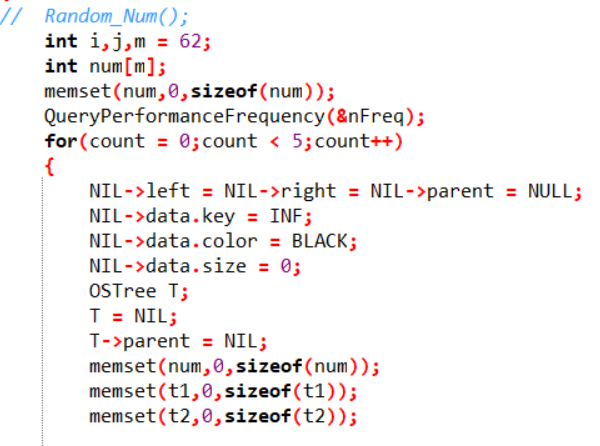


10.线性时间查找第i小数的随机算法

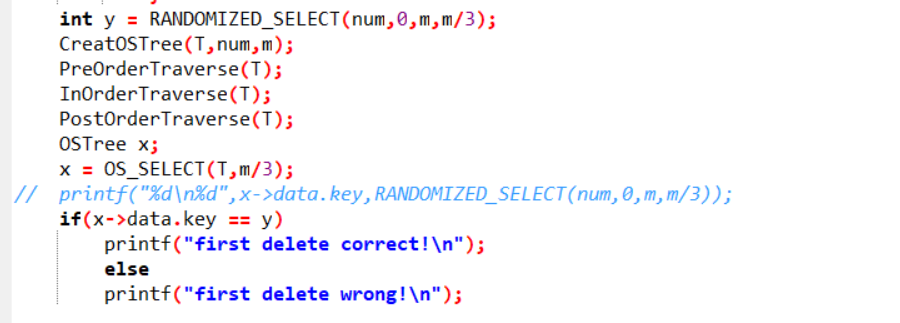




11.变量及顺序统计树初始化



12.判断删除结点是否正确



1. **实验结果、分析（结合相关数据图表分析）**

插入操作：（n已取log2对数）

对原始数据的观察可以发现，无论n的大小，前十个结点插入的运行时间明显高于后面的结点，这应该与空间分配的消耗有关。另外，由于数据规模较小以及测量工具的局限，因此得到的测量数据难以支持算法导论中插入操作O(lgn)的论断。

删除操作：（n已取log2对数）

由于本实验删除操作的数据规模极小，单次删除操作运行时间与待删除结点在树中的位置有较大关系，因此得到的测量数据难以支持算法导论中删除操作O(lgn)的论断。

1. **实验心得**

通过对红黑树/顺序统计树各种操作算法的实现，我较清楚地理解了红黑树/顺序统计树的基本原理，了解了红黑树/顺序统计树的优缺点，了解了红黑树的应用方向。进一步学习了怎样测量程序运行时间以及图表的制作。