**作业5 查找结构与排序方法**

**1． 设计**BST**的左右链存储结构，并实现**BST**插入、删除、查找和排序算法。**

**首先定义**BST**树结点的数据结构，包含数据域以及左右儿子指针域，具体如下：**

图形用户界面

中度可信度描述已自动生成

而后BST的插入、删除、查找和排序均可以采用**递归**的思想

1. BST插入

判断新节点在当前结点的左子树还是右子树上，递归至空时申请新节点。

文本

描述已自动生成

1. BST建立

读取文件数据，重复调用插入函数即可

图形用户界面, 文本

描述已自动生成

1. BST查找

将待查数据与当前指针数据进行比较，从而决定向左或向右遍历，如若数据匹配或查找至空域（查找失败）即返回当前指针。引入count对查找长度进行记录。

文本

描述已自动生成

1. BST删除

遍历至BST树的最左节点将其删除，并将其右子树继承至其父节点上。

文本

描述已自动生成

1. BST排序

由于BST树的性质，其中序遍历序列即为其排序序列。

文本

中度可信度描述已自动生成

**2． 实现折半查找算法。**

与BST 的思想类似，对有序序列按照待查找值与当前中值比较后递归比较前半部分或者后半部分。

文本

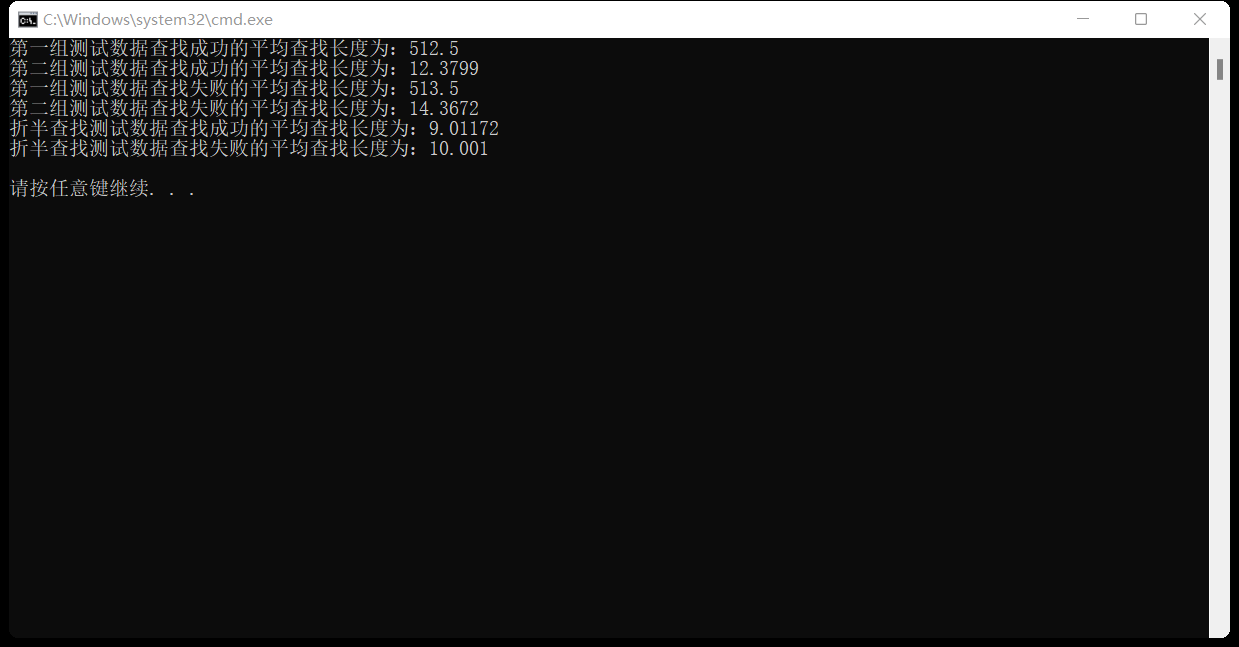
描述已自动生成

**3． 实验比较：设计并产生实验测试数据，考察比较两种查找方法的时间性能，并与理论结果进行比较。**

第一组数据：0至2048之间的已排序的奇数序列

第二组数据：第1组测试数据的随机序列。

实验结果具体如下：



理论情况下，第一组数据存入BST树时退化为单链表，但由于其有序性，可得查找成功的平均查找长度为，查找失败的平均查找长度为，与实验结果相吻合；第二组数据由于其随机性，其平均成功查找长度取决于二叉排序树的实际形态，取值范围为，实验结果也表明随机性越高，BST的查找效率越高。

而折半查找的平均成功查找长度为，查找失败的平均查找长度约为树的高度，实验结果显然也符合理论情形。

以上实验并不能完全说明就平均性能而言，BST的查找与折半查找差不多，原因是BST的查找严重依赖于数据的随机性，在有序序列的情形下，折半查找是远远优于BST查找方式的，而在随机性较强的情况下，二者的查找性能差距较小，但折半查找需要满足有序性，因而很难说两种查找方式谁更优秀，实际情况实际分析，但就维护表的有序性而言，BST的方式更为有效。