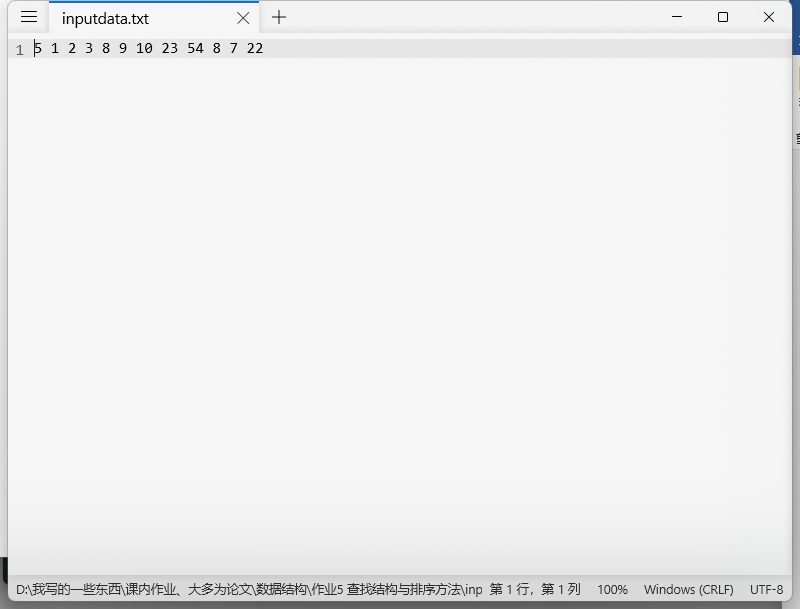
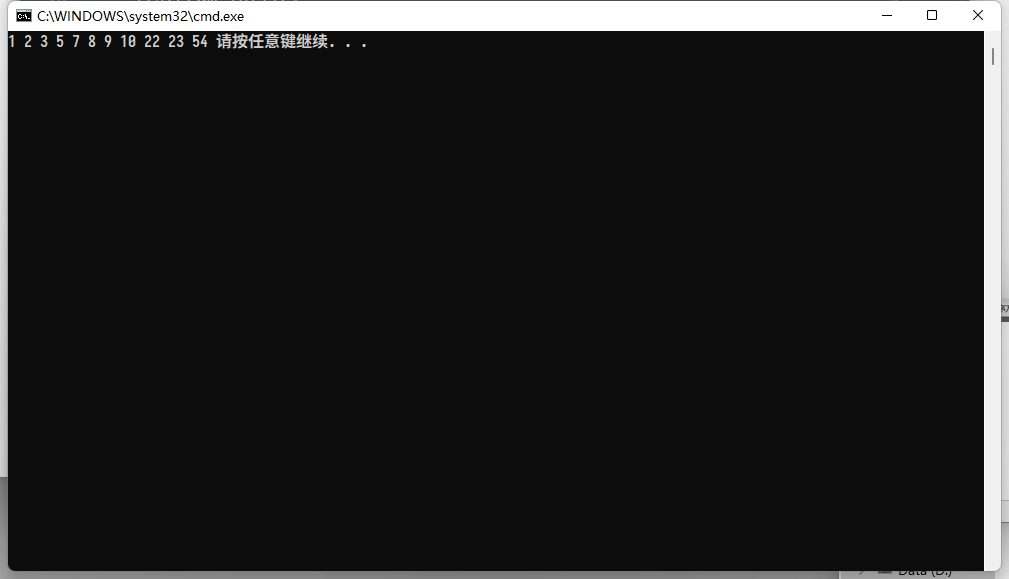
# BST相关操作

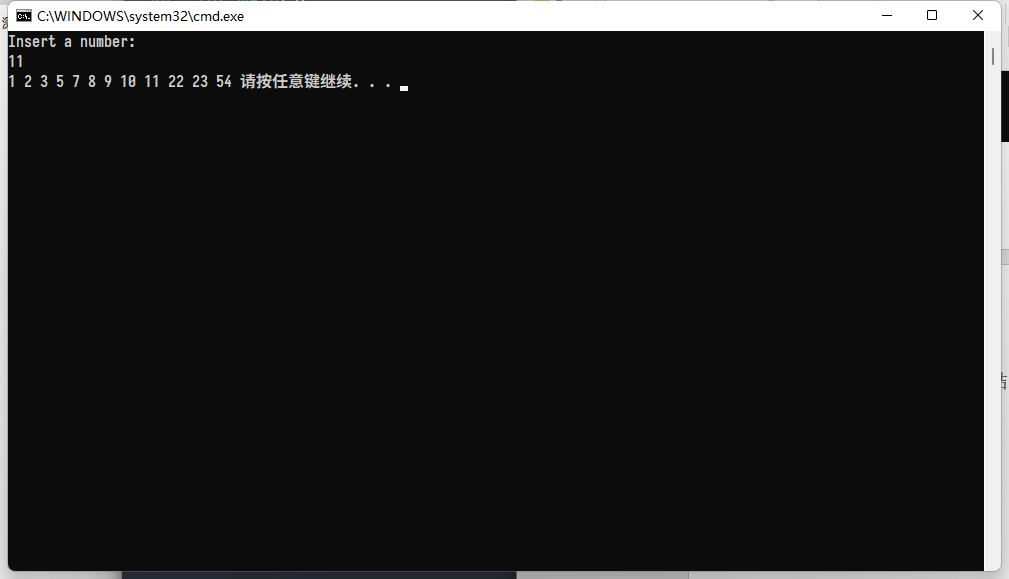
测试数据：



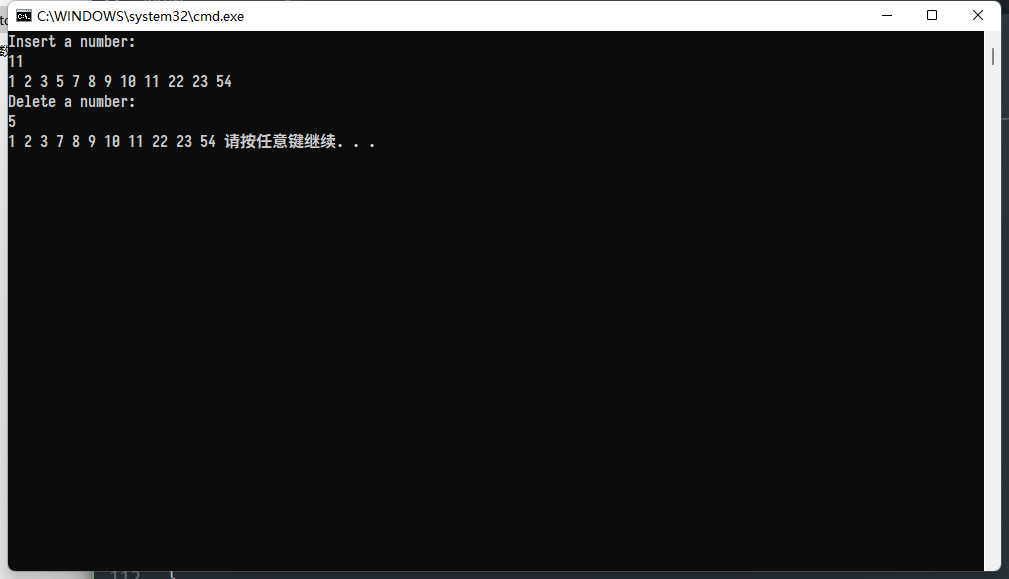
排序结果：



插入数字11 后的排序结果：

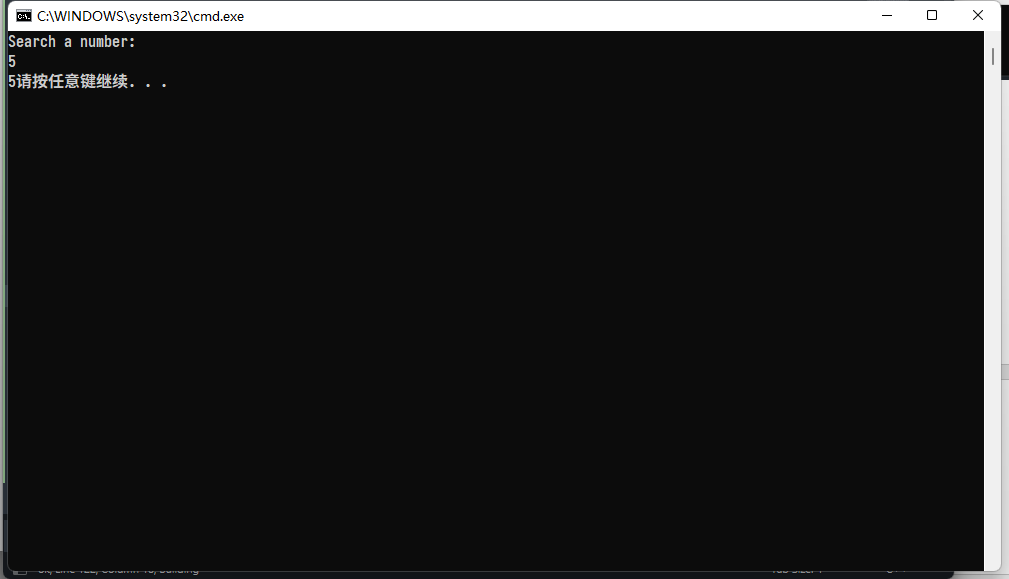


删除数字 5 后的排序结果：

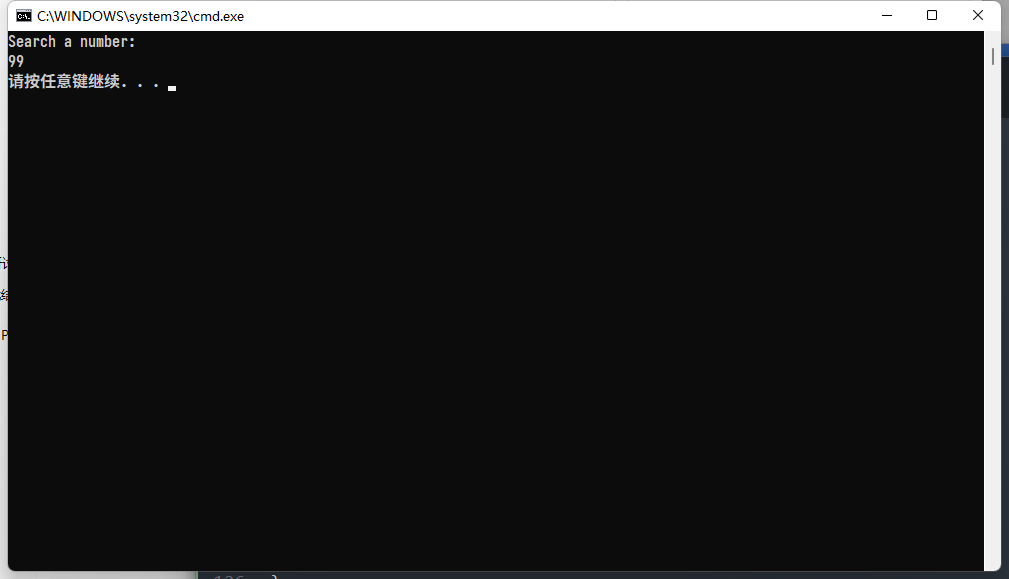


查找数字（返回结点）

查找5：

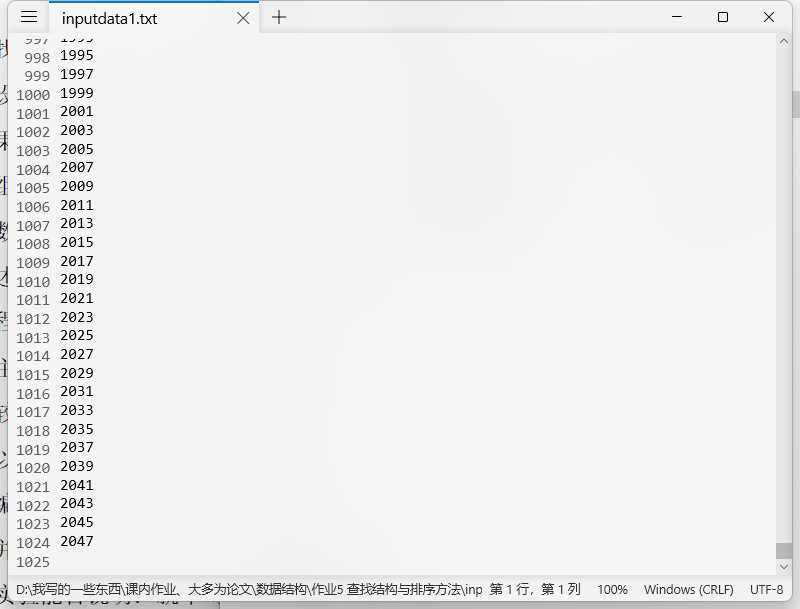


查找 99：（查找不到，返回为空）



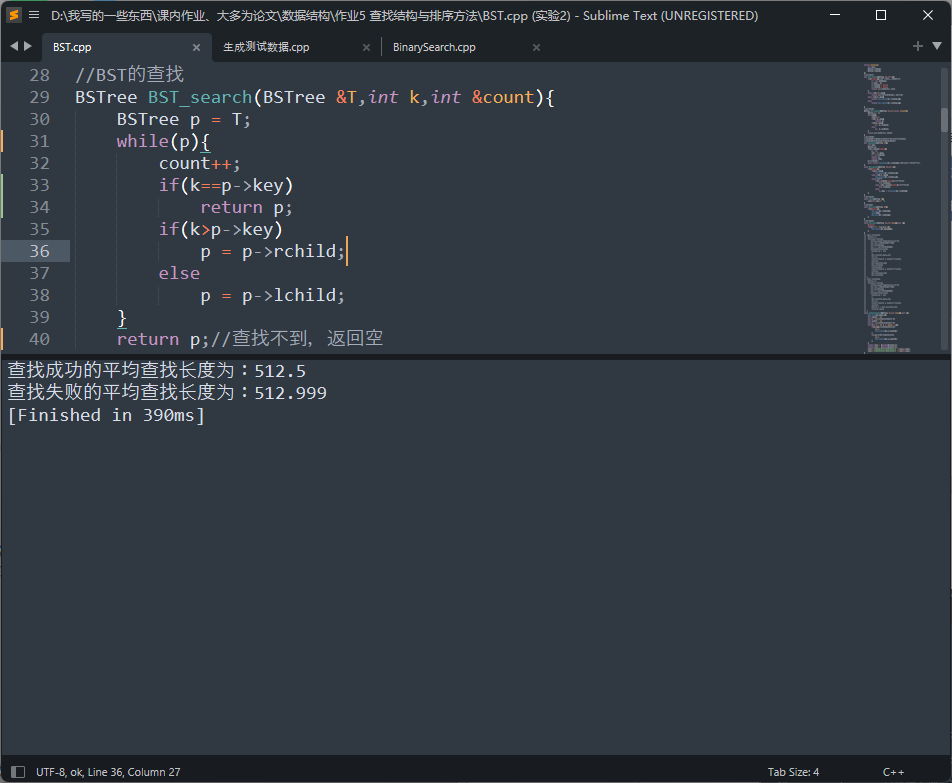
# 实验比较

## 1 第1组数据

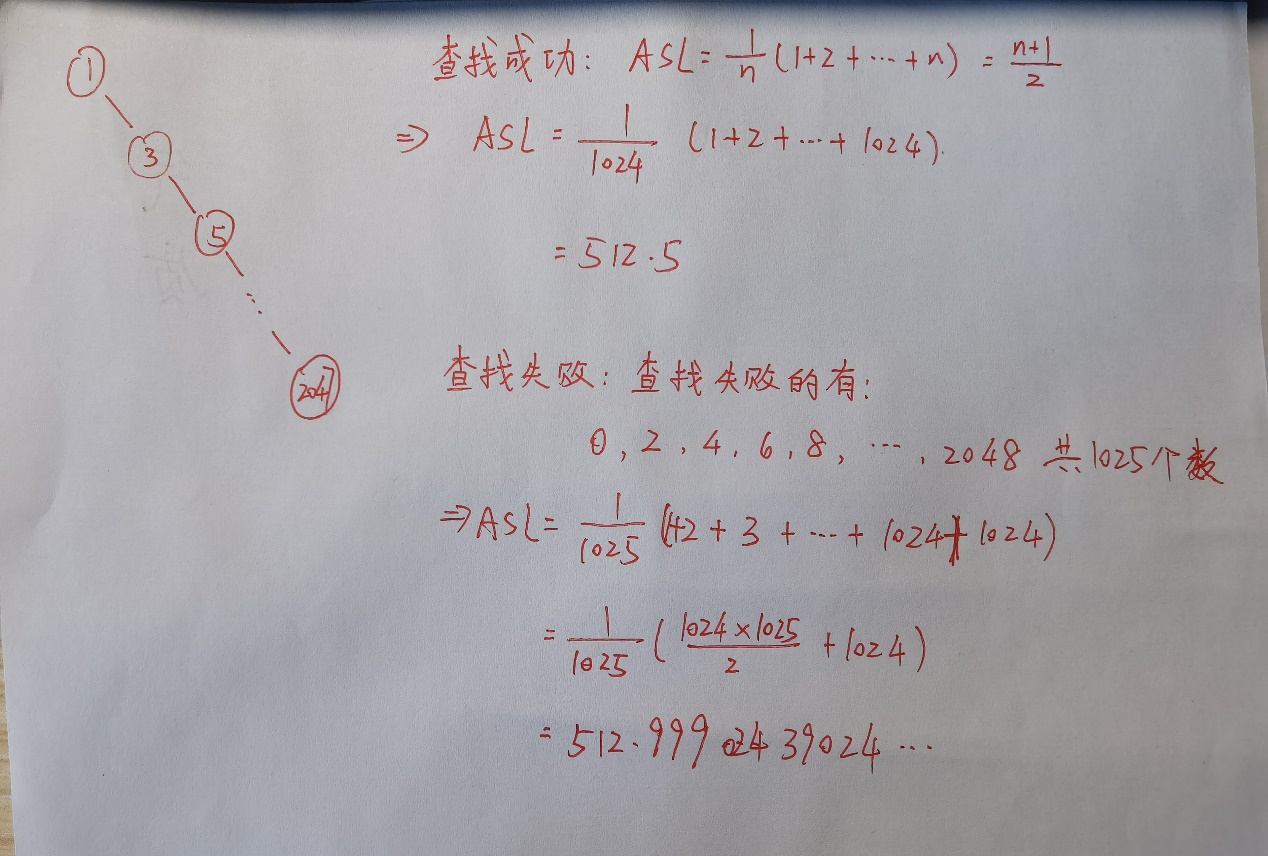


这是0至2048之间的奇数

BST的查找成功和查找失败的平均查找长度：

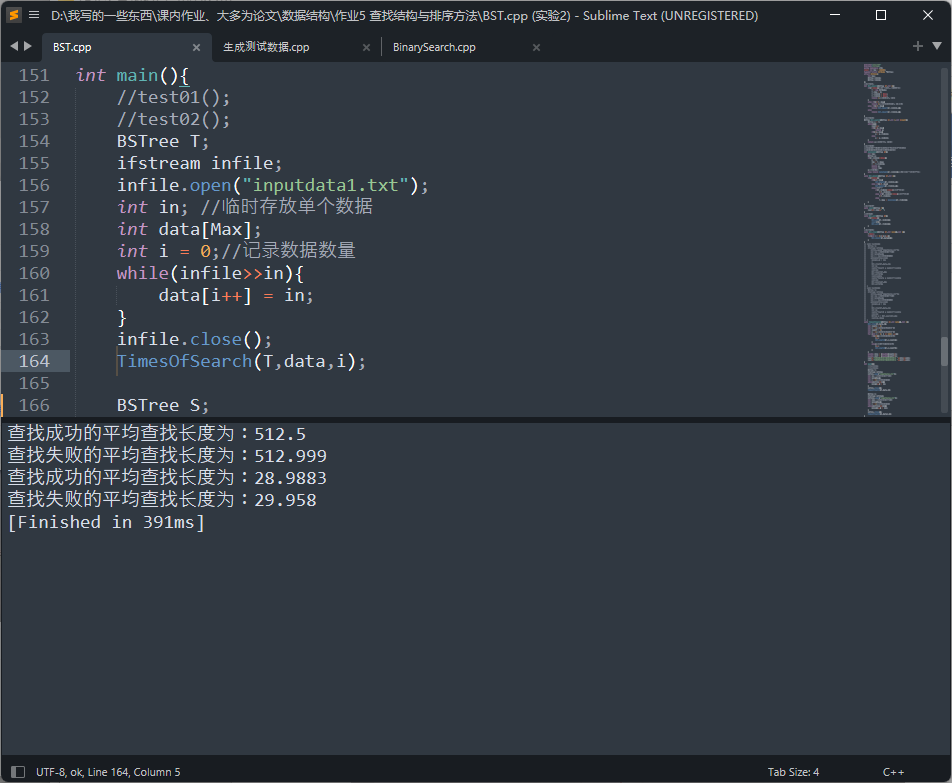


与理论值比较：



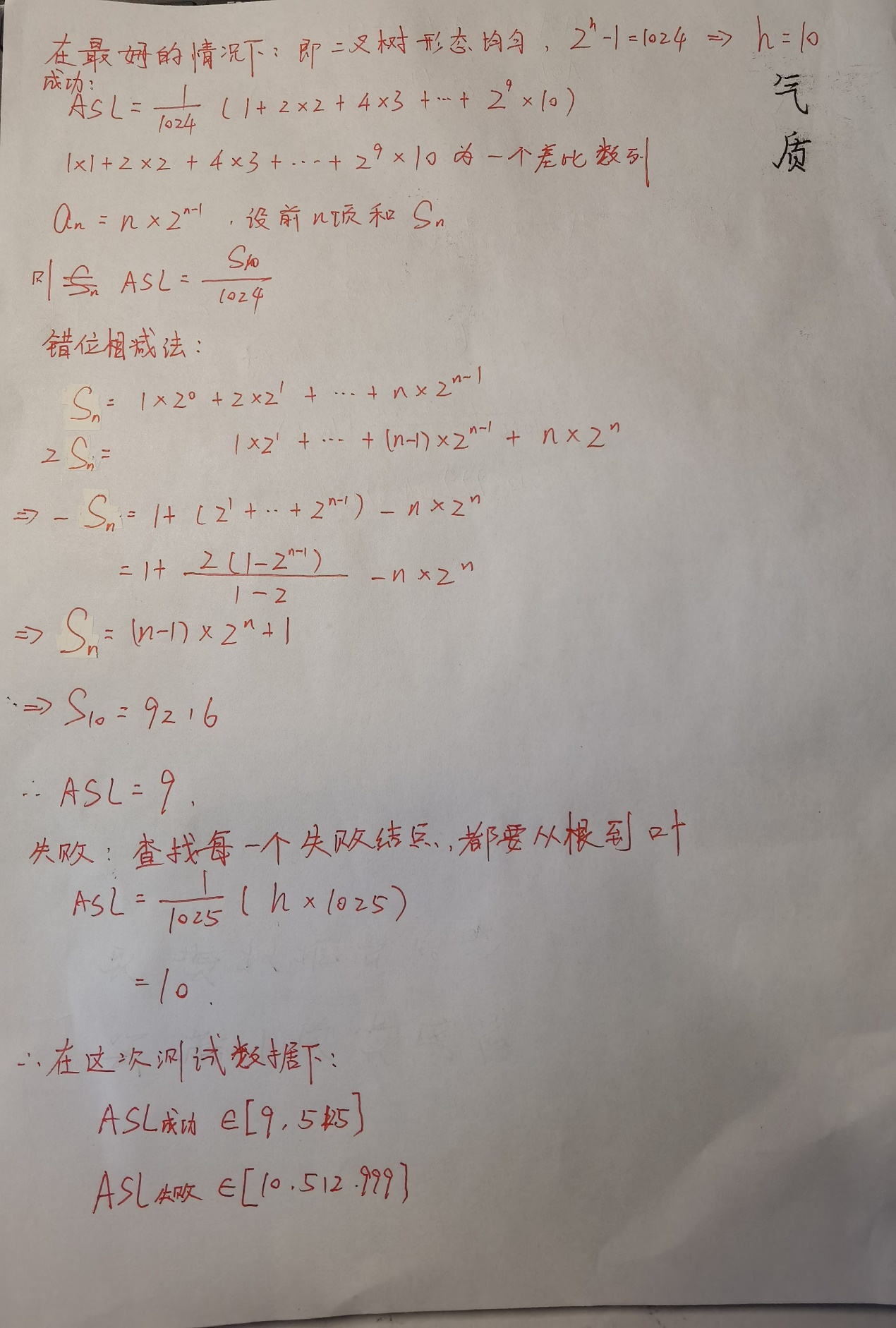
与理论值完全符合

## 2 第2组数据

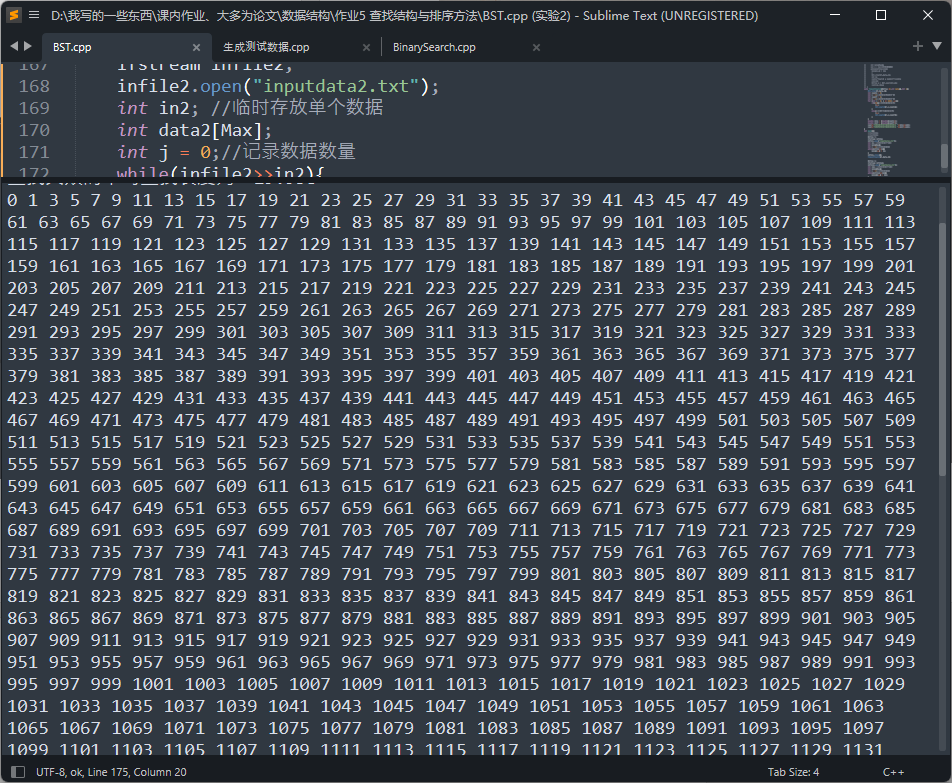


与理论值比较：

由于该测试数据数据量庞大，无法精确算出理论值具体是多少，下面算出最好情况下的结果，并分别给出一个范围

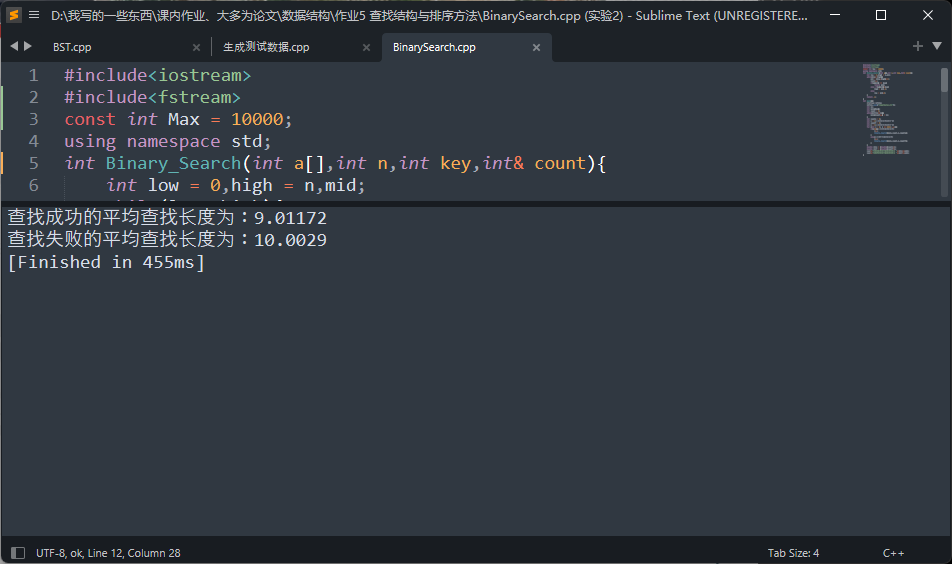


## BST查找结构的中序遍历序列



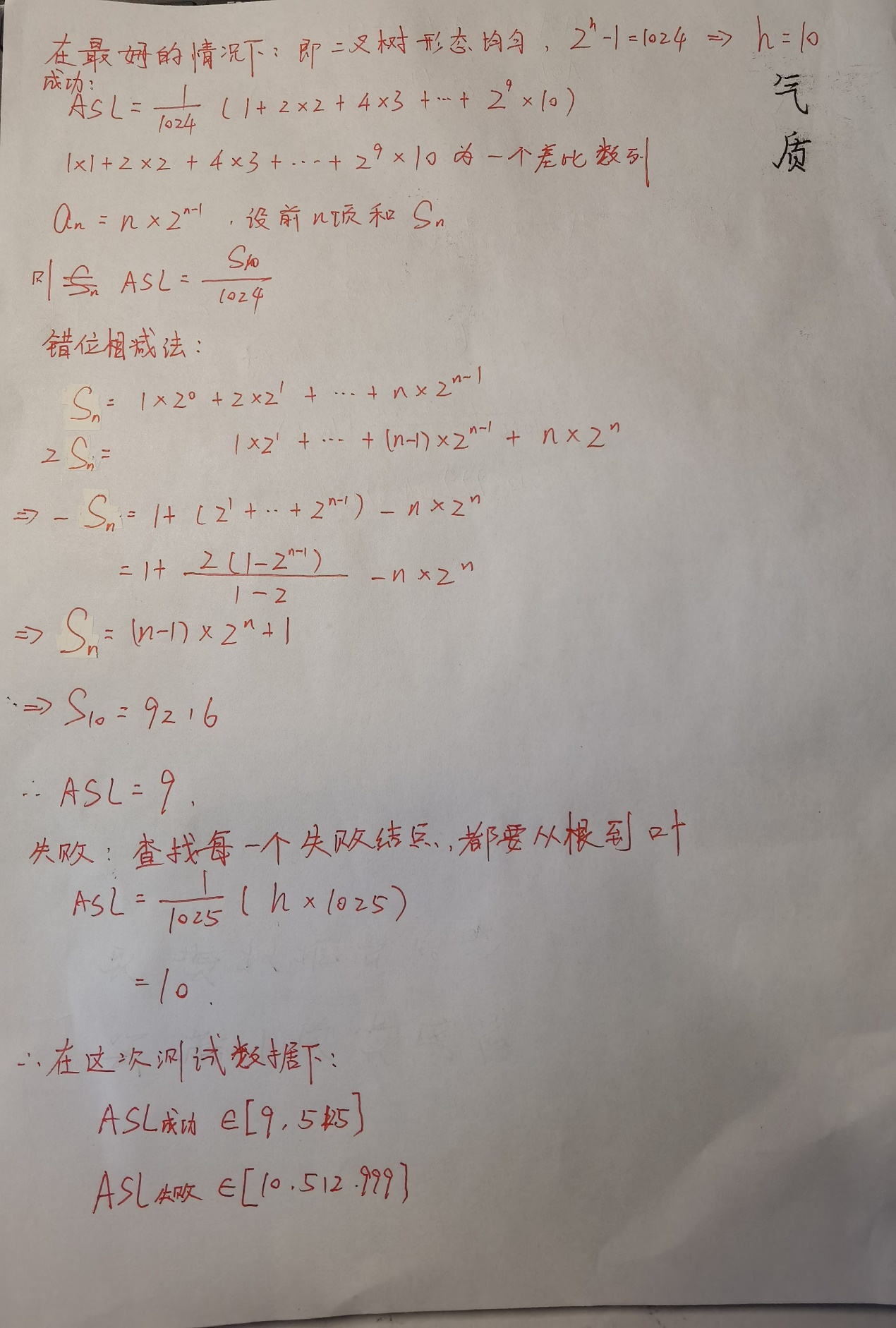
升序排列，与第一组测试数据是一样的

## 折半查找测试



折半查找在最好的情况下的平均查找长度与BST是一样的

理论推导也一样：



可见，实验结果与最好的情况下的推导结果几乎一样。

# 结论

无论输入数据如何，折半查找的判定树始终都是非常均衡的。而BST的查找树与数据排列关系非常大。

就平均而言，如果BST的查找树非常均衡，那么可以认为两者的性能是差不多的，甚至可以说时间性能完全相同。