**项目说明文档**

**数据结构课程设计**

**——二叉排序树**

作 者 姓 名： 戴仁杰

学 号： 1951650

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

目 录

[1 分析 - 3 -](#_Toc91208574)

[1.1 背景分析 - 3 -](#_Toc91208575)

[1.2 功能分析 - 3 -](#_Toc91208576)

[2 设计 - 3 -](#_Toc91208577)

[2.1 数据结构设计 - 3 -](#_Toc91208578)

[2.2 类结构设计 - 3 -](#_Toc91208579)

[2.3 成员与操作设计 - 3 -](#_Toc91208580)

[2.3.1 AVLTreeNode类 - 3 -](#_Toc91208581)

[2.3.2 AVLTree类 - 4 -](#_Toc91208582)

[2.3.3 BsortTreeManagementSystem类 - 5 -](#_Toc91208583)

[2.4 系统设计 - 5 -](#_Toc91208584)

[3 实现 - 6 -](#_Toc91208585)

[3.1 建立二叉排序树功能的实现 - 6 -](#_Toc91208586)

[3.1.1 建立二叉排序树功能流程图 - 6 -](#_Toc91208587)

[3.1.2 建立二叉排序树功能核心代码 - 6 -](#_Toc91208588)

[3.1.3 建立二叉排序树功能截屏展示 - 7 -](#_Toc91208589)

[3.2 向二叉排序树中插入元素功能的实现 - 7 -](#_Toc91208590)

[3.2.1 向二叉排序树中插入元素功能流程图 - 7 -](#_Toc91208591)

[3.2.2 向二叉排序树中插入元素功能核心代码 - 7 -](#_Toc91208592)

[3.2.3 向二叉排序树中插入元素功能截屏示例 - 8 -](#_Toc91208593)

[3.3 查询元素功能的实现 - 8 -](#_Toc91208594)

[3.3.1 查询元素功能流程图 - 8 -](#_Toc91208595)

[3.3.2 查询元素功能核心代码 - 8 -](#_Toc91208596)

[3.3.3 查询元素功能截屏示例 - 9 -](#_Toc91208597)

[4 测试 - 9 -](#_Toc91208598)

[4.1 正常功能测试 - 9 -](#_Toc91208599)

[4.2边界测试 - 10 -](#_Toc91208600)

[4.2.1 仅输入0建立二叉排序树的情况 - 10 -](#_Toc91208601)

[4.2.2 输入全部重复元素建立二叉排序树的情况 - 10 -](#_Toc91208602)

[4.3 出错测试 - 11 -](#_Toc91208603)

[4.3.1 操作数输入错误 - 11 -](#_Toc91208604)

[4.3.2 输入的排序数字不合法 - 11 -](#_Toc91208605)

1 分析

1.1 背景分析

排序和查找是计算机科学中非常重要的操作，探究如何快速高效地排序并查找出指定元素具有十分重要的意义。在排序的数据结构中，二叉排序树是一种十分有效的算法，并且可以通过改进成平衡二叉树（AVL树），使其具有更好的排序、查找效率。

1.2 功能分析

二叉排序树就是指将原来已有的数据根据大小构成一棵二叉树，二叉树中的所有结点数据满足一定的大小关系，所有的左子树中的结点均比根结点小，所有的右子树的结点均比根结点大。改进后的排序树即AVL树除了具有上述特点外，还满足左右子树的高度差绝对值不超过1，保证了其高度不会太高。

二叉排序树查找是指按照二叉排序树中结点的关系进行查找，查找关键自首先同根结点进行比较，如果相等则查找成功；如果比根节点小，则在左子树中查找；如果比根结点大，则在右子树中进行查找。这种查找方法可以快速缩小查找范围，大大减少查找关键的比较次数，从而提高查找的效率。对于平衡二叉树，由于树的高度不会太高，其查找的速度会更快。

2 设计

2.1 数据结构设计

如上功能分析所述，查找元素需要用到一个已经排好序的平衡二叉树，在按顺序输入的过程中，不断向二叉树建立、插入结点，并根据树的高度进行相应翻转，保持树的平衡性。所以在此过程中需要用一棵二叉树来存储输入的数据，并提供插入查询旋转平衡的操作。

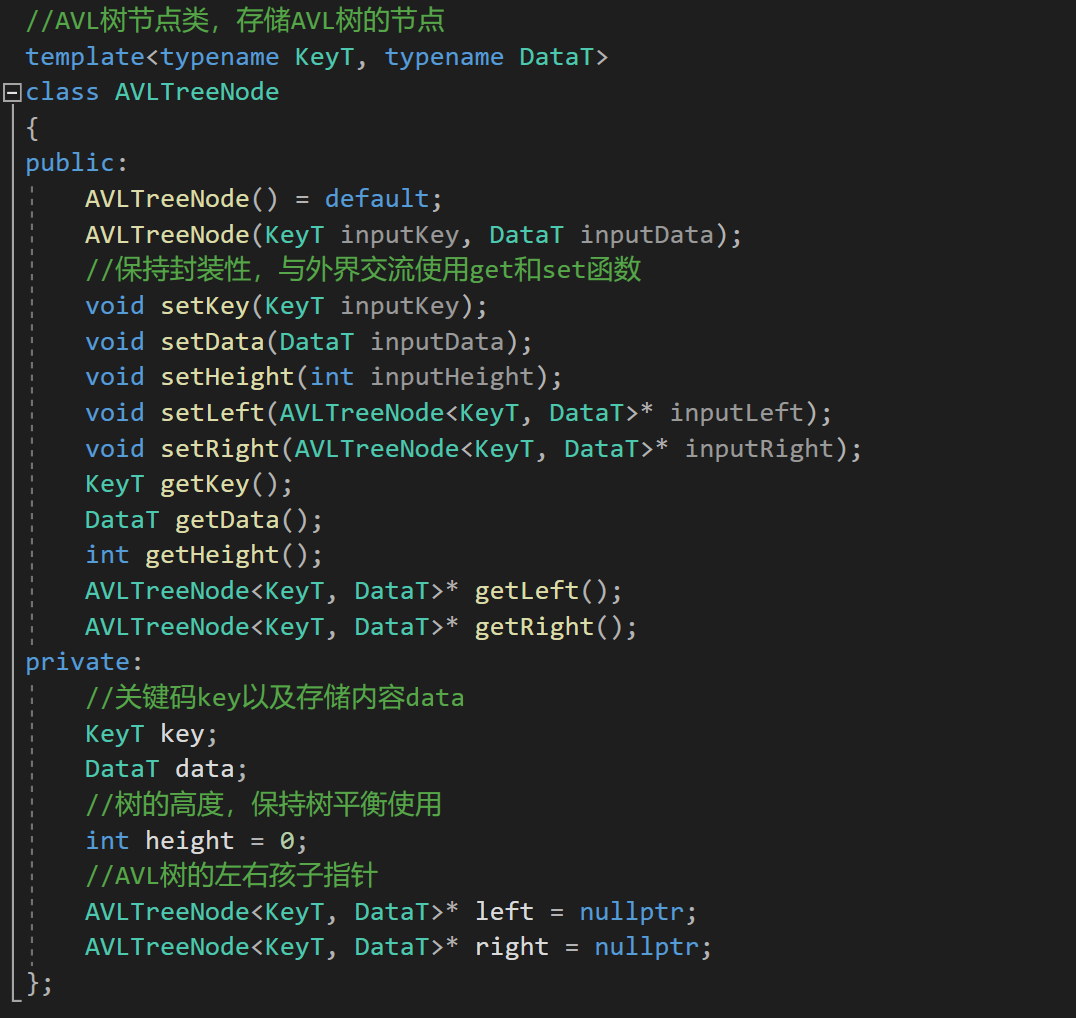
2.2 类结构设计

首先，要将数据存储在一棵二叉树中，所以要有一个二叉树类，同时为了优化其性能，将此类设置为平衡二叉树（AVLTree）类，并配有相对应的结点（AVLTreeNode）类。最后，更加方便管理，还设计了一个二叉树管理系统（BsortTreeManagementSystem）类，将关键字的插入、查询、显示菜单与结果等功能整合在一起。为了使平衡树类更具有泛用性，本程序将AVLTreeNode类和AVLTree类都设计为了模板类。

2.3 成员与操作设计

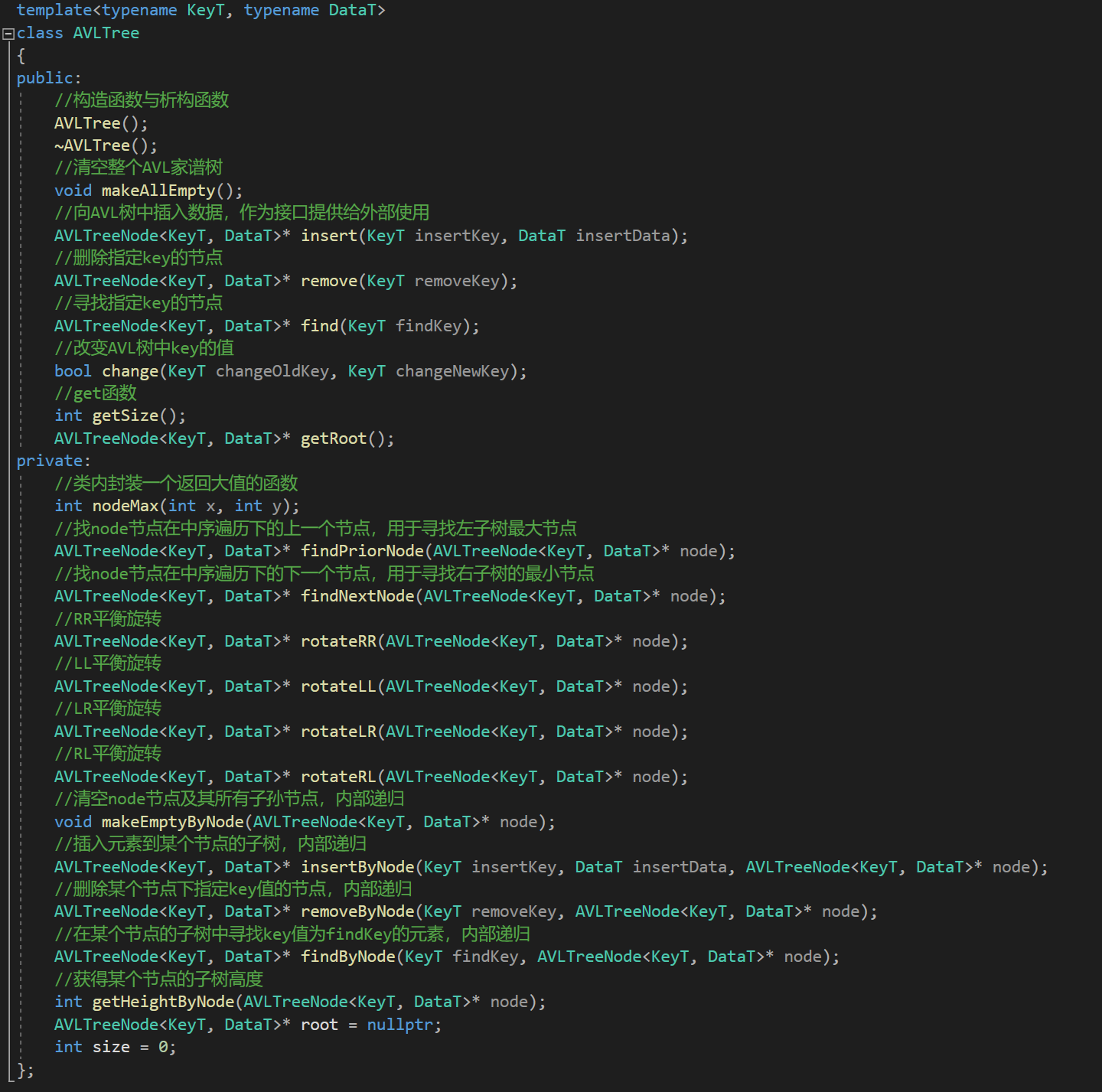
2.3.1 AVLTreeNode类

AVLTreeNode类为二叉树结点类，采用了类模板提高泛用性。提供默认构造函数、带参构造函数以及与外接数据交流的一系列get与set函数。类内数据为关键值、data数据域、左右孩子指针以及一个记录孩子高度的整数，存储了结点信息并与孩子结点建立联系。



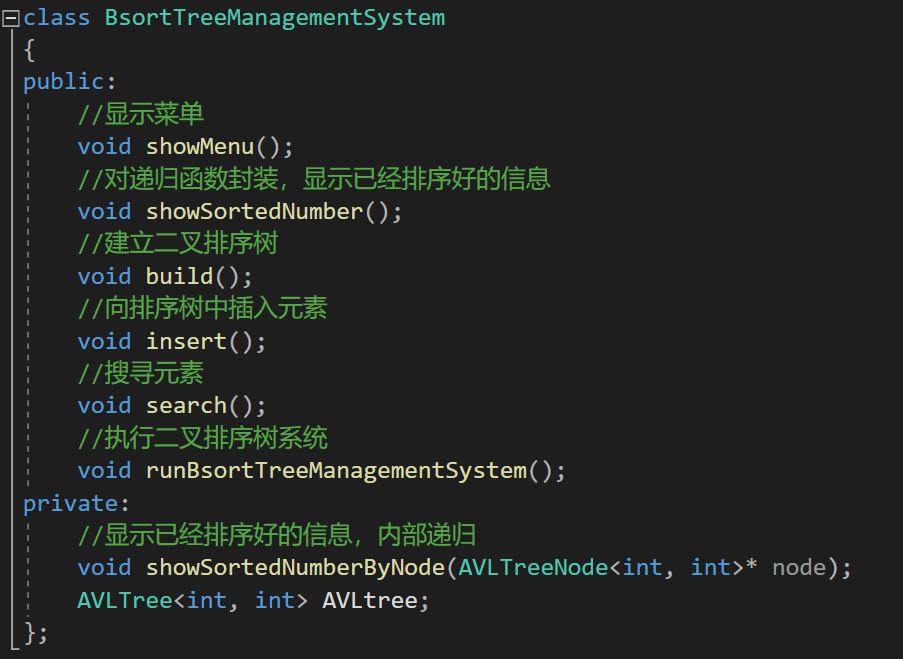
2.3.2 AVLTree类

AVLTree类为二叉树类，采用了类模板提高泛用性。提供增删查改与清空的操作，并提供了get和set系列函数与外界进行数据交流。同时内部有一系列辅助函数用于完成功能。类内数据为二叉树根节点和记录大小的数字。



2.3.3 BsortTreeManagementSystem类

BsortTreeManagementSystem类为二叉树表达式计算系统类。对AVLTree类进行封装增强，丰富了交互提示信息，增强了输入错误、边界错误处理性能。类内封装了建立二叉排序树函数、插入元素函数、查询元素函数。还提供了一个runBsortTreeManagementSystem()函数用于启动程序，方便外部调用。类内数据为一个平衡二叉树类型变量，用于保存表达式的操作符和操作数信息。



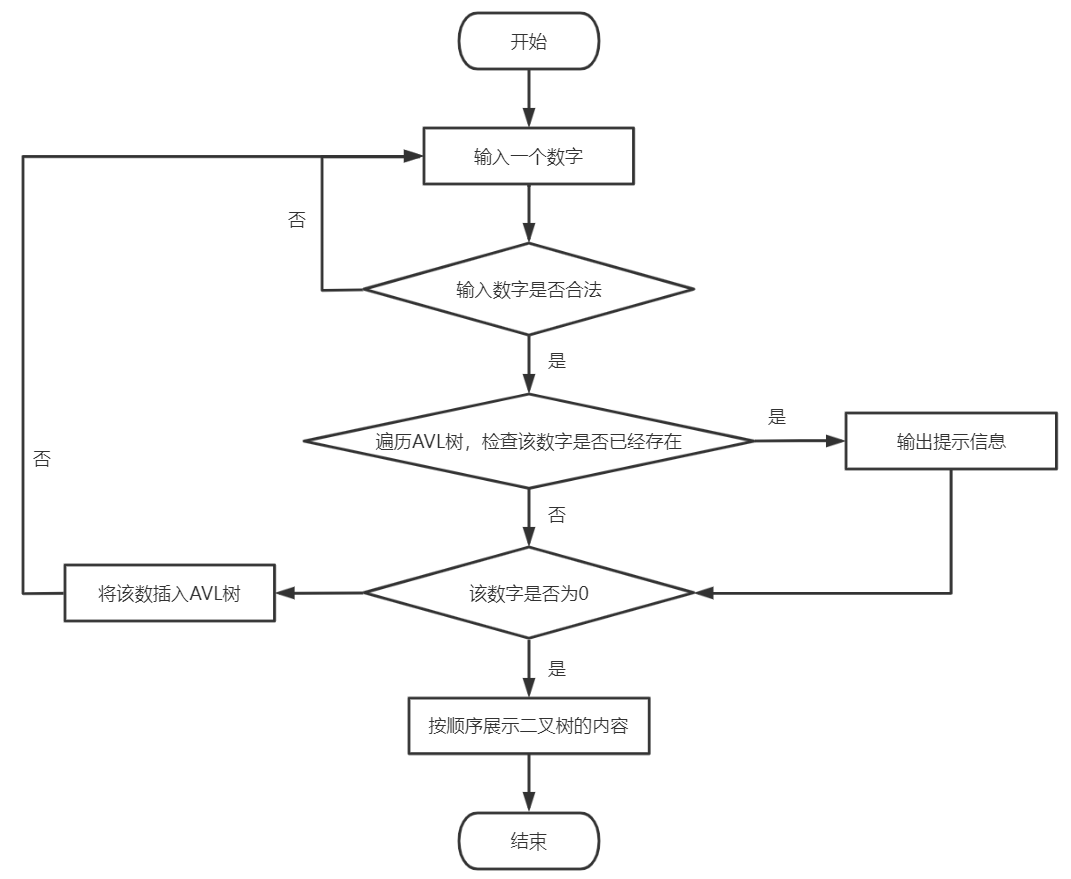
2.4 系统设计

程序在使用时，首先依次输入要排序和寻找的数据，然后对数据依次插入AVL树，并进行旋转调整二叉树的高度，建立起平衡二叉树表达式系统。如果插入失败，会将错误信息显示在控制台。查询时，同根结点进行比较，如果相等则查找成功；如果比根节点小，则在左子树中查找；如果比根结点大，则在右子树中进行查找。直至最后，如果未找到，则提示对应的信息。

# 3 实现

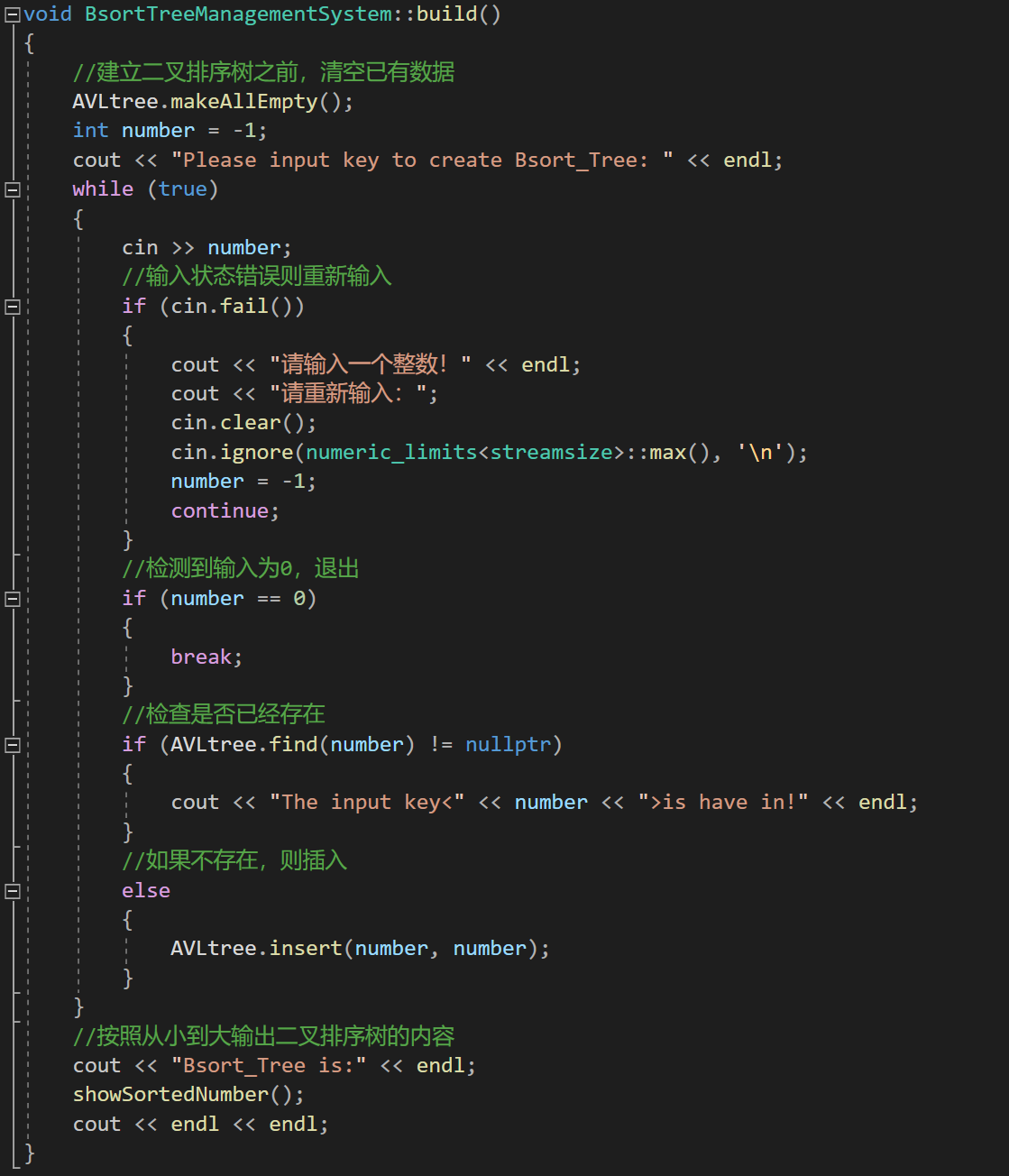
3.1 建立二叉排序树功能的实现

3.1.1 建立二叉排序树功能流程图

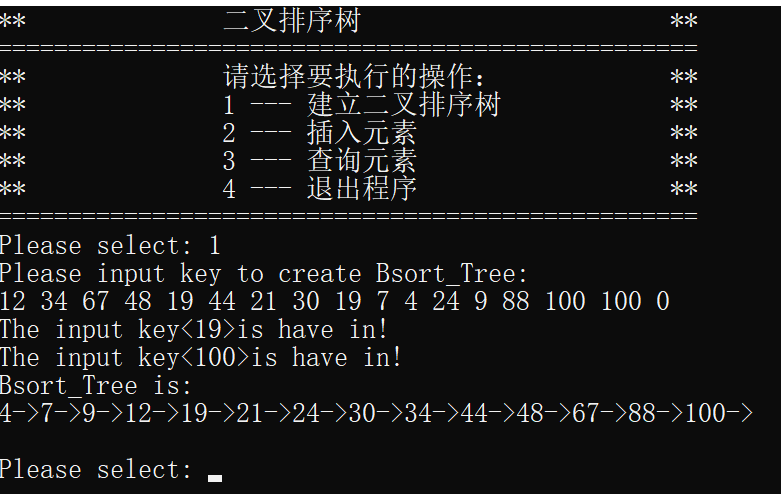


3.1.2 建立二叉排序树功能核心代码

BsortTreeManagementSystem类的build()函数

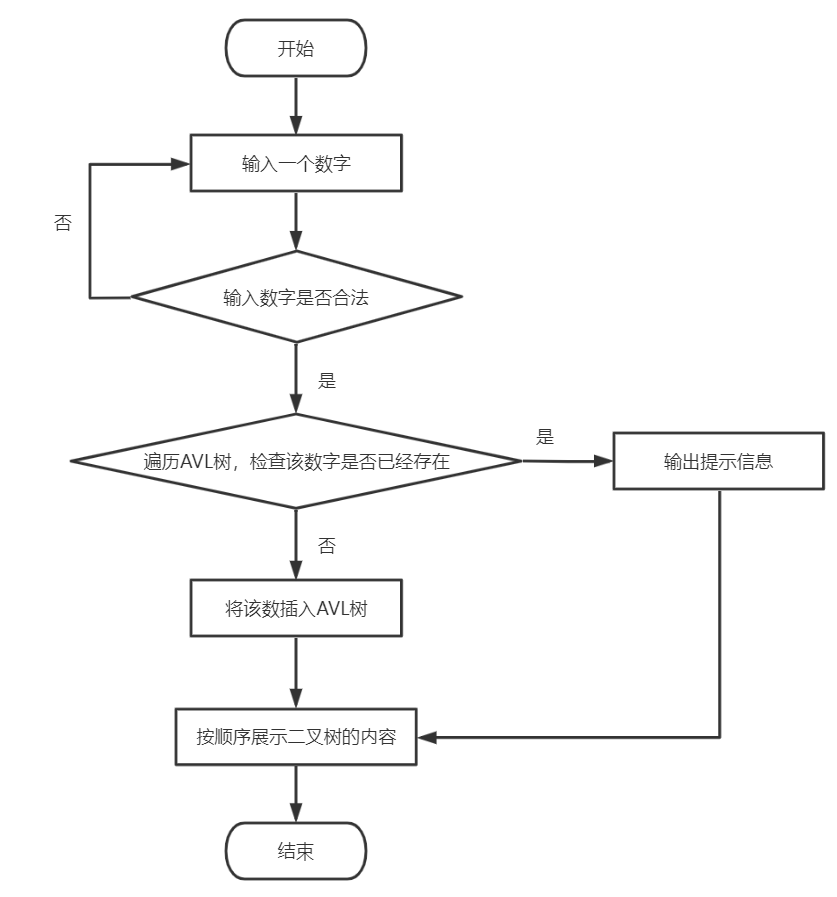


3.1.3 建立二叉排序树功能截屏展示



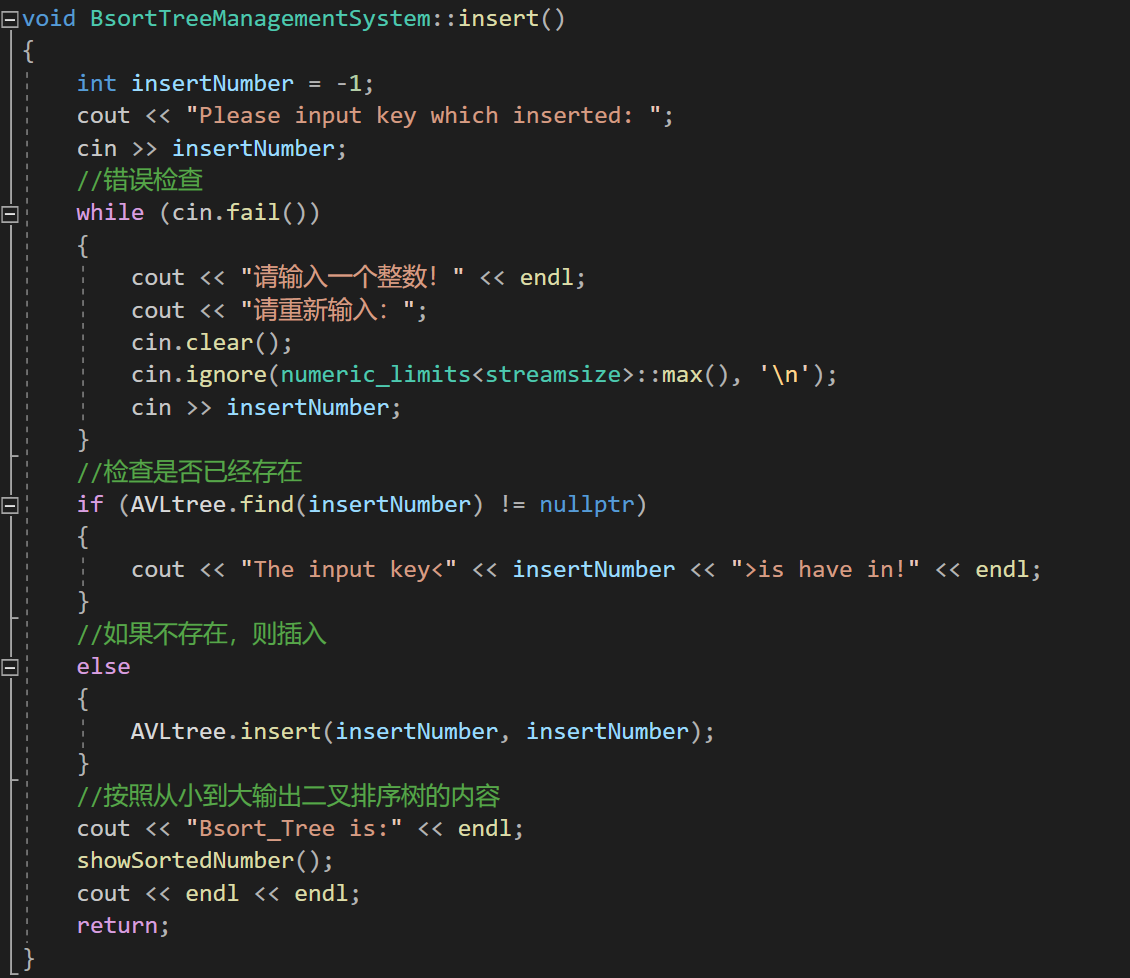
3.2 向二叉排序树中插入元素功能的实现

3.2.1 向二叉排序树中插入元素功能流程图

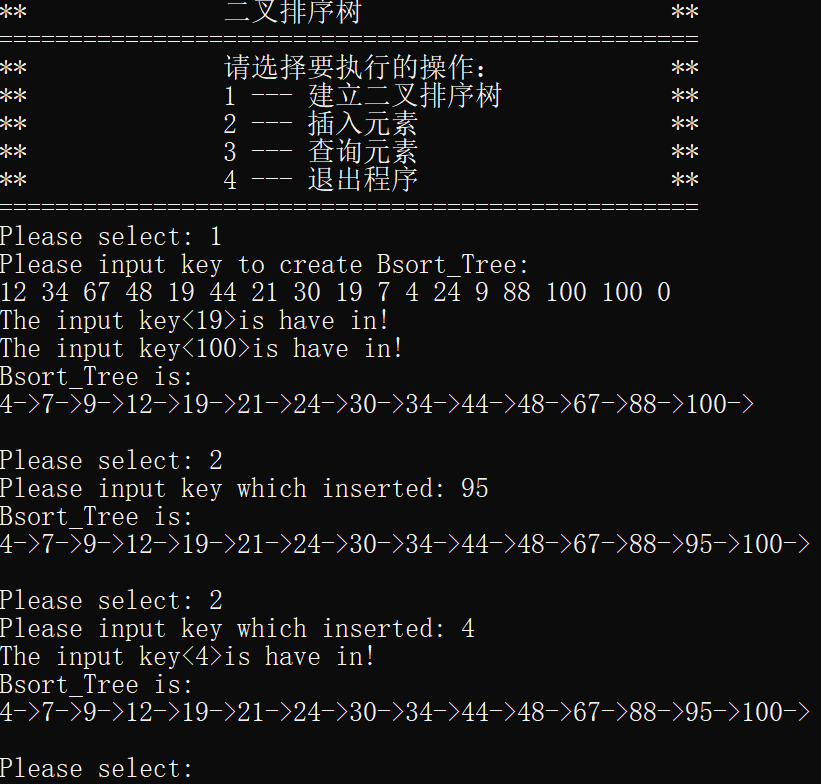


3.2.2 向二叉排序树中插入元素功能核心代码

BsortTreeManagementSystem类的insert()函数

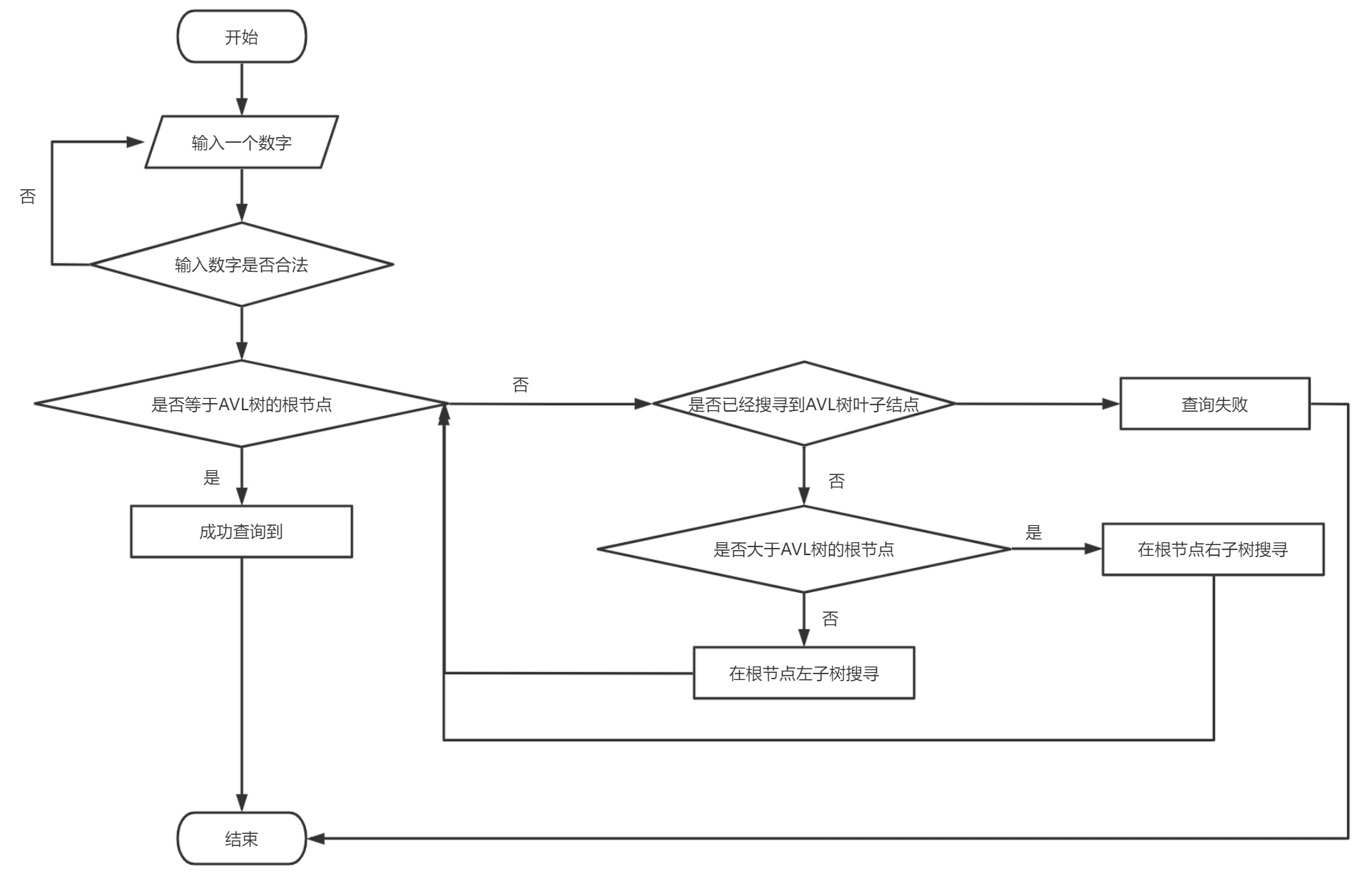


3.2.3 向二叉排序树中插入元素功能截屏示例



3.3 查询元素功能的实现

3.3.1 查询元素功能流程图

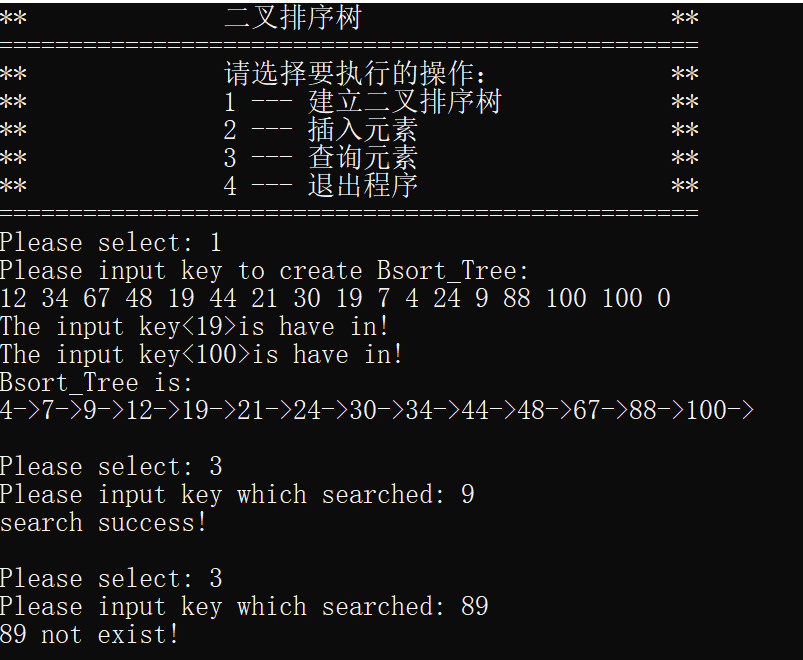


3.3.2 查询元素功能核心代码

AVLTreeNode类的findByNode()函数



3.3.3 查询元素功能截屏示例

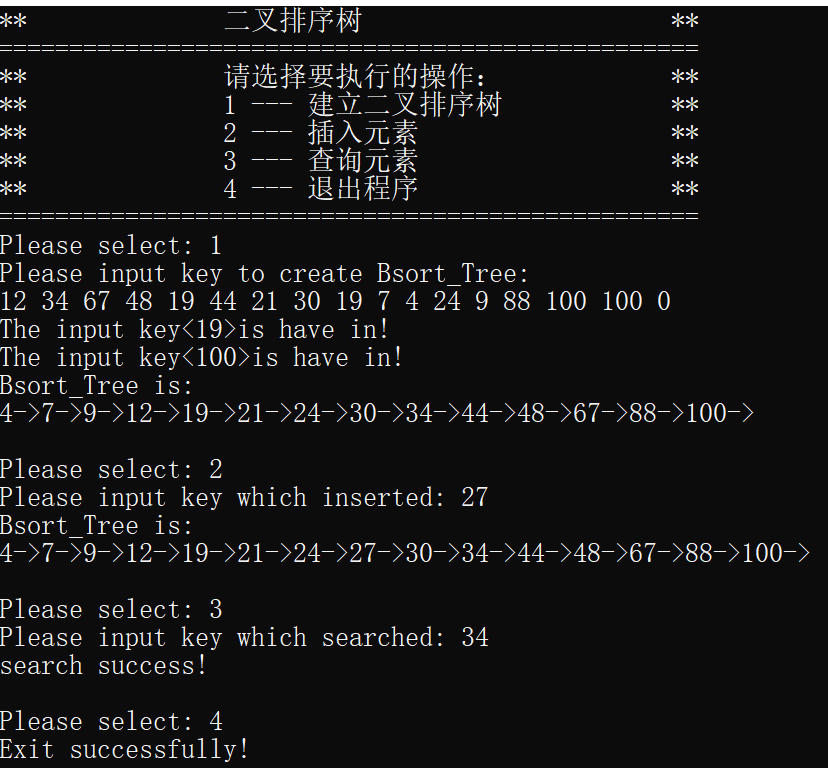


4 测试

4.1 正常功能测试

**测试用例**：一段正常的10个数字，包含重复

**实验结果：**排序顺序正确，插入正确，查询正确

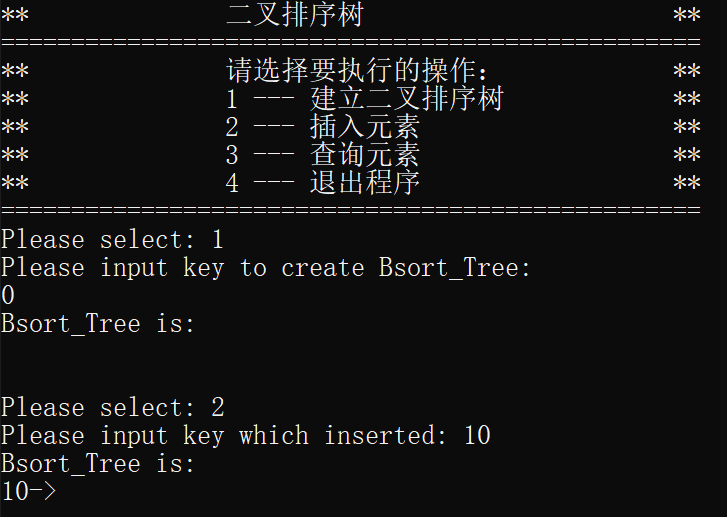


4.2边界测试

4.2.1 仅输入0建立二叉排序树的情况

**测试用例**：仅输入0建立二叉排序树

**实验结果：**程序正常运行，并且能够继续执行后续插入元素等操作



4.2.2 输入全部重复元素建立二叉排序树的情况

**测试用例**：输入全部重复元素10建立二叉排序树

**实验结果：**程序正常运行，且正确输出一个10



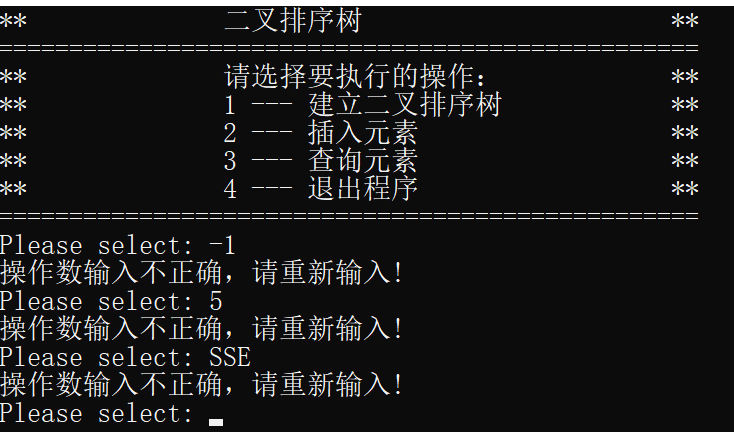
4.3 出错测试

4.3.1 操作数输入错误

**测试用例**：输入的操作数不在1-4之间，或者为非数字

**预期结果**：提示输入错误，并要求重新输入

**实验结果：**符合预期



4.3.2 输入的排序数字不合法

**测试用例**：输入的排序数为非数字

**预期结果**：提示输入错误，并要求重新输入

**实验结果：**符合预期

