

**本科生实验报告**

**实验课程 数据结构与程序设计**

**学院名称 核技术与自动化工程学院**

**专业名称 测控技术与仪器**

**学生姓名 追梦少年南南**

**学生学号 202006010336**

**指导教师 朱 杰**

**实验地点 6C802**

**实验成绩**

**二〇二二年二月 二〇二二年六月**

**填写说明**

1. 适用于本科生所有的实验报告（印制实验报告册除外）；
2. 专业填写为专业全称，有专业方向的用小括号标明；
3. 格式要求：
4. 用A4纸双面打印（封面双面打印）或在A4大小纸上用蓝黑色水笔书写。
5. 打印排版：正文用宋体小四号，1.5倍行距，页边距采取默认形式（上下2.54cm，左右2.54cm，页眉1.5cm，页脚1.75cm）。字符间距为默认值（缩放100%，间距：标准）；页码用小五号字底端居中。
6. 具体要求：

**题目**（二号黑体居中）；

**摘要**（“摘要”二字用小二号黑体居中，隔行书写摘要的文字部分，小4号宋体）；

**关键词**（隔行顶格书写“关键词”三字，提炼3-5个关键词，用分号隔开，小4号黑体)；

正文部分采用三级标题；

**第1章** ××(小二号黑体居中，段前0.5行)

**1.1** ×××××小三号黑体×××××（段前、段后0.5行）

**1.1.1**小四号黑体（段前、段后0.5行）

**参考文献**（黑体小二号居中，段前0.5行），参考文献用五号宋体，参照《参考文献著录规则（GB/T 7714－2005）》。

实验一 线性表的应用

1. 实验目的：
2. 掌握线性表的逻辑结构和存储结构特点；
3. 掌握线性表的基本操作，如建立、查找、插入和删除等。
4. 问题描述

智能家居系统创建一个家居环境参数表，包含“日期、时间、地点、温度、湿度”等信息。程序能够完成如下功能：

(1) 能够逐条输入信息，创建表；

(2) 能够显示表中的所有信息；

(3) 根据时间和地点进行查找，返回相关参数信息；

(4) 给定一条环境参数信息，按照日期和时间顺序插入到表中指定的位置；

(5) 删除指定日期的记录；

三、数据结构设计（选用的数据元素逻辑结构和存储结构实现形式说明）

（1）逻辑结构设计 我写了一个顺序表的结构。

（2）存储结构设计 分配了MAXSIZE=100的存储空间域

（3）存储结构形式说明

每个空间域有一个表示长度的整型变量和一个指针结构体。结构体里面有三个字符串组

分别表示日期，时间，地点；有两个整形变量分别表示温度和湿度。

四、算法设计

（1）算法列表（说明各个函数的名称，作用，完成什么操作）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 顺序表初始化 | InitList | 初始化一个顺序表 |
| 2 | 顺序表创建 | ListCreate | 可以输入信息 |
| 3 | 顺序表查找 | Locate | 按照时间和地点查找 |
| 4 | 顺序表插入 | ListInsert | 在指定位置插入顺序表 |
| 5 | 顺序表删除 | ListDelete | 删除指定位置的顺序表 |
| 6 | 主函数 | Main | 主函数 |

（2）各函数间调用关系（画出函数之间调用关系）

InitList

ListCreate

主函数

Locate

ListInsert

ListDelete

（3）关键算法描述

顺序表初始化：用c++的new分配100个空间域。

顺序表创建：提示输入的东西，按照提示创建顺序表。

顺序表查找：按照时间和地点查找，如果相同则显示查找到的内容，如果不相同则显示

“对不起，没找到”。

顺序表插入：选择要插入的位置，并且输入插入的数据，插入成功后会显示出来。

顺序表删除：选择要删除的位置，删除之后会显示出来。

五、调试记录（调试过程中遇到的主要问题，是如何解决的，对设计和编码的回顾讨论和分析；改进设想等）

第一次调试失败：是我的顺序表创建函数的参数有问题，没有用取地址符。

第二次调试失败：我的算法出现了问题。

第三次调试失败：我在删除顺序表的时候循环处初始变量用了1，导致每次循环都会删除第一个。将1改为i解决。\*

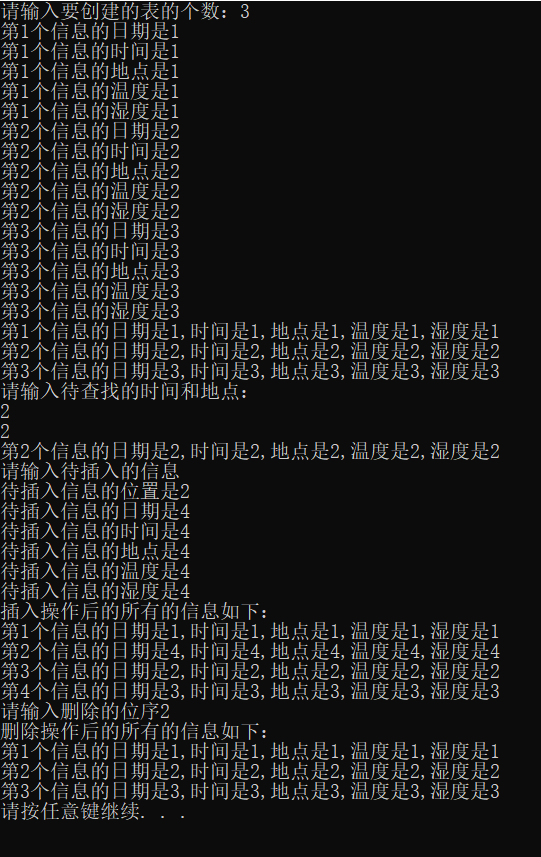
六、运行说明（列出测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于示例中所列数据）

输入：1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 （初始化）

输入：2 2 （待查找的时间和地点）

输入：2 4 4 4 4 4 （要插入的位置和信息）

输入：2（要删除的位置）



实验二 栈和队列的应用

1. 实验目的：
2. 掌握栈和队列的逻辑结构及存储结构；
3. 运用栈和队列原理完成设计的内容。
4. 问题描述

1、完成数字十进制到八进制的转换。

输入示例：

请输入需转换的数的个数：

3

请输入需转换的数：

28，58，190

输出示例：

转换结果为：

1、 34

2、 72

3、 276

2、银行排队系统实现

功能要求：

1. 客户进入排队系统；
2. 客户离开；
3. 查询当前客户前面还有几人；
4. 查询截至目前总共办理多少客户。

输出要求：每进行一次操作后，输出当前排队成员情况。

三、数据结构设计（选用的数据逻辑结构和存储结构实现形式说明）

（1）逻辑结构设计

数制转换：使用除八取余法，将待转化数据每次除以8，令其余数依次入栈，等待其完成入栈后依次出栈。

银行排队：设计一个队列，先来的客人先进入队列，后来的客人后进入队列。最后先来的客人先走出队列，后来的客人后走出队列。

（2）存储结构设计

二者都分配了100个存储空间。

（3）存储结构形式说明

数制转换：给栈分配100个存储空间，将每次转换之后得到的数放到栈中，再制作一个数组，将每次出栈后的元素依次放到数组中。

银行排队：给队列分配100个存储空间，将每次用户入队放到队列中。再依次出队

四、算法设计

（1）算法列表（说明各个函数的名称，作用，完成什么操作）

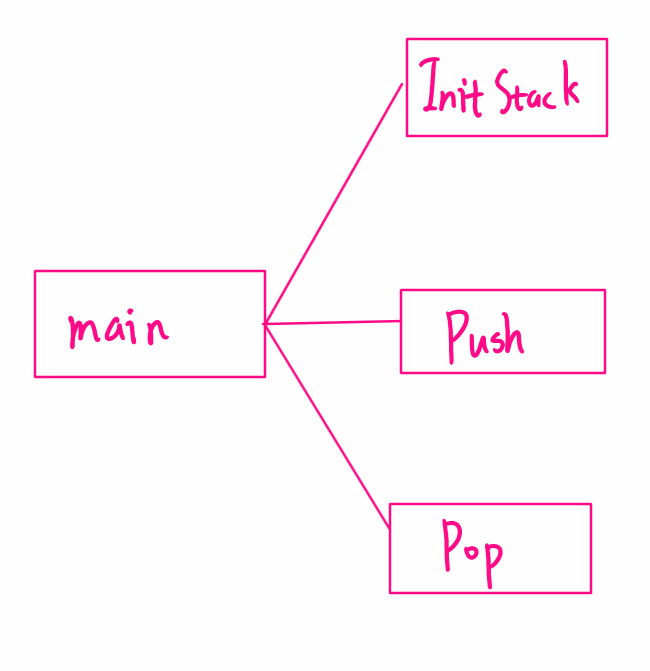
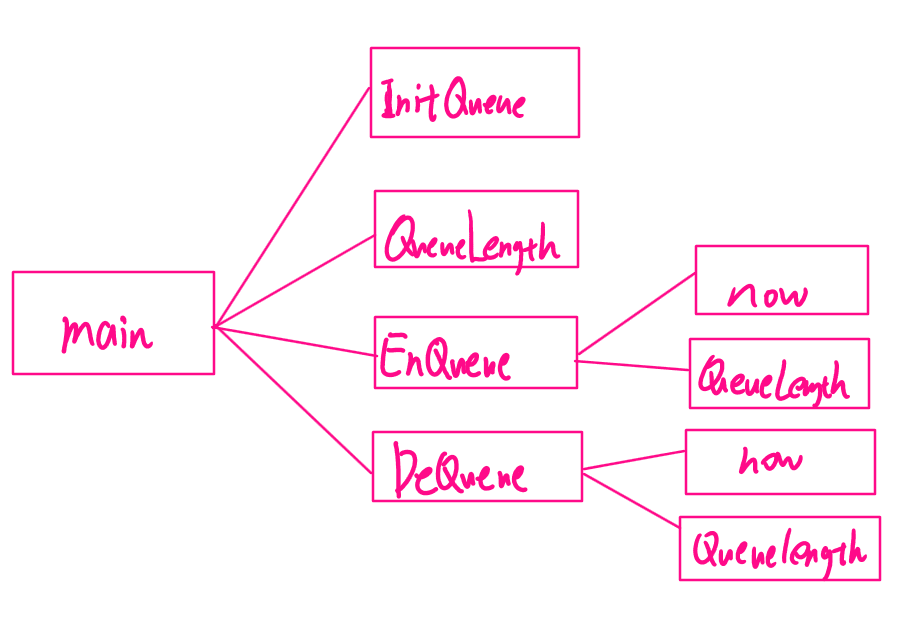
进制转换：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 栈的初始化 | InitStack | 给栈分配存储空间和基地址 |
| 2 | 元素入栈 | Push | 将元素e压入栈 |
| 3 | 元素出栈 | Pop | 将元素e拿出栈 |
| 4 | 主函数 | main | 主函数 |

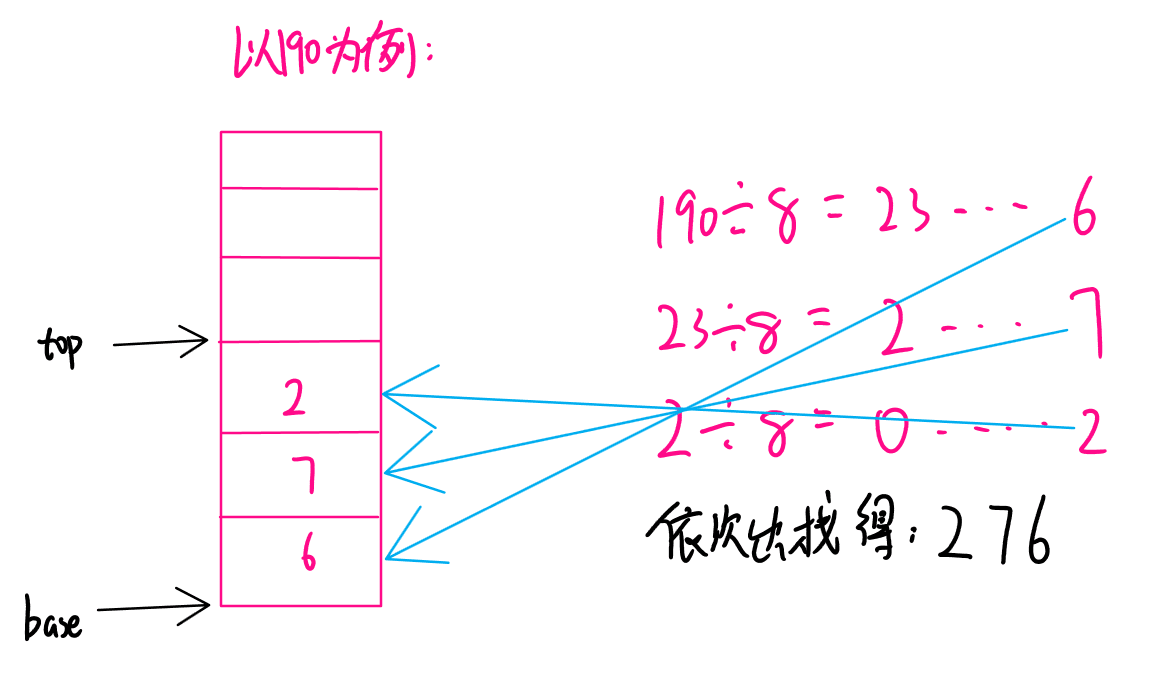
银行排队：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 队列的初始化 | InitQueue | 给队列分配存储空间和基地址 |
| 2 | 队列长度 | QueueLength | 返回队列的长度 |
| 3 | 入队 | EnQueue | 在队尾插入一个元素 |
| 4 | 出队 | DeQueue | 将队头元素删除 |
| 5 | 遍历 | now | 输出当前遍历 |
| 6 | 主函数 | Main | 主函数 |

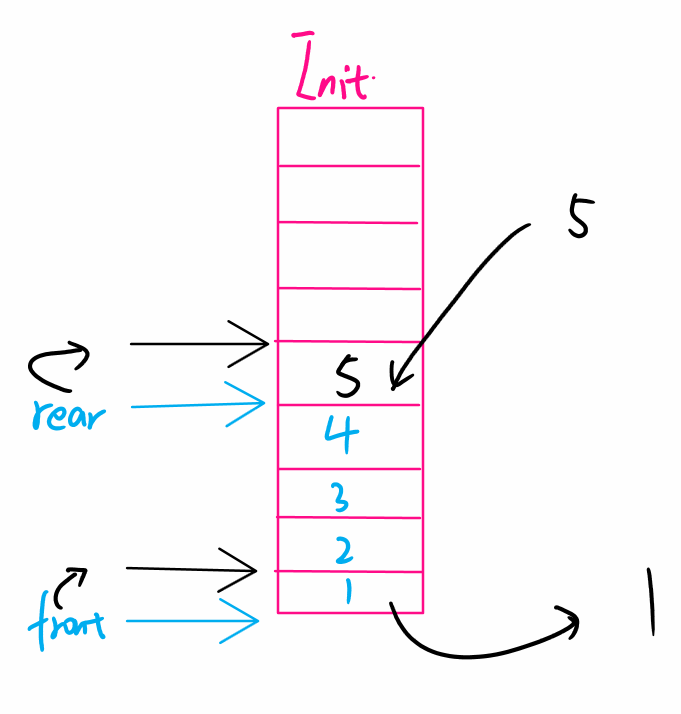
（2）各函数间调用关系（画出函数之间调用关系）

（3）关键算法描述







五、调试记录（调试过程中遇到的主要问题，是如何解决的，对设计和编码的回顾讨论和分析；改进设想等）

进制转换：

第一次调试成功：成功找到了入栈出栈的实现过程；

第二次调试成功：将一位十进制数成功转换为8进制数；

第三次调试失败：循环出了问题；

第四次调试成功：完成了基本功能。

银行排队：

第一至四次调试失败：没有完全掌握队列的使用方法；

第五次调试成功：成功实现了队列的入队和出队；

第六次调试成功：加入了求队列长度的功能；

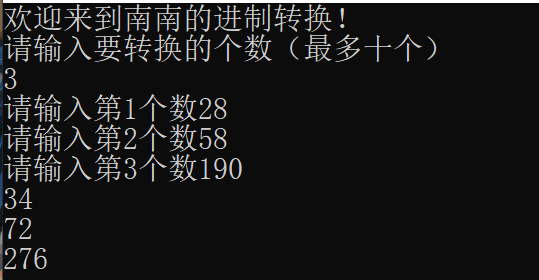
第七次调试成功：加入了求当前位置的功能；

第八次调试成功：主函数加入了循环；

六、运行说明（列出测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于示例中所列数据）

欢迎来到南南的进制转换！

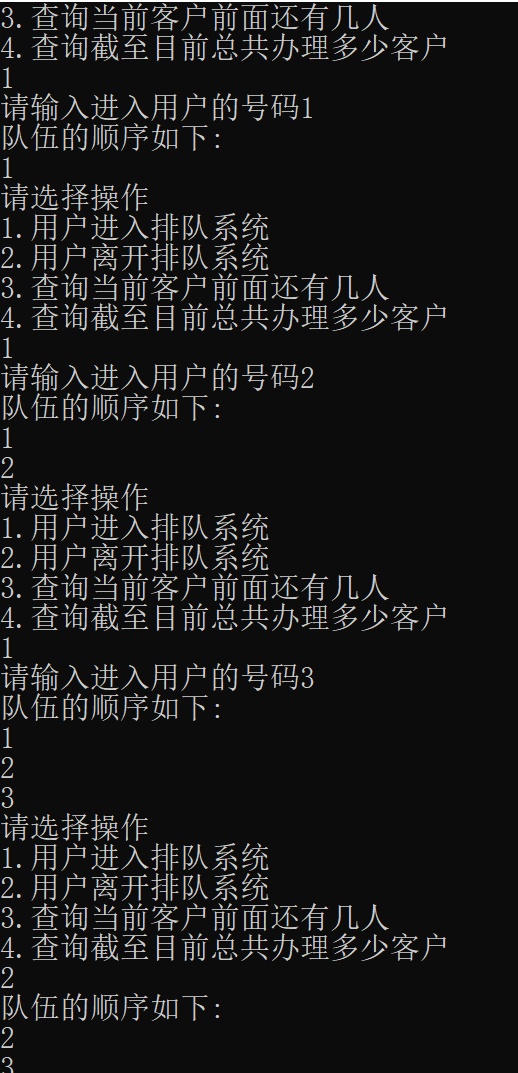
输入你要转换数的个数以及你要转换的数即可。

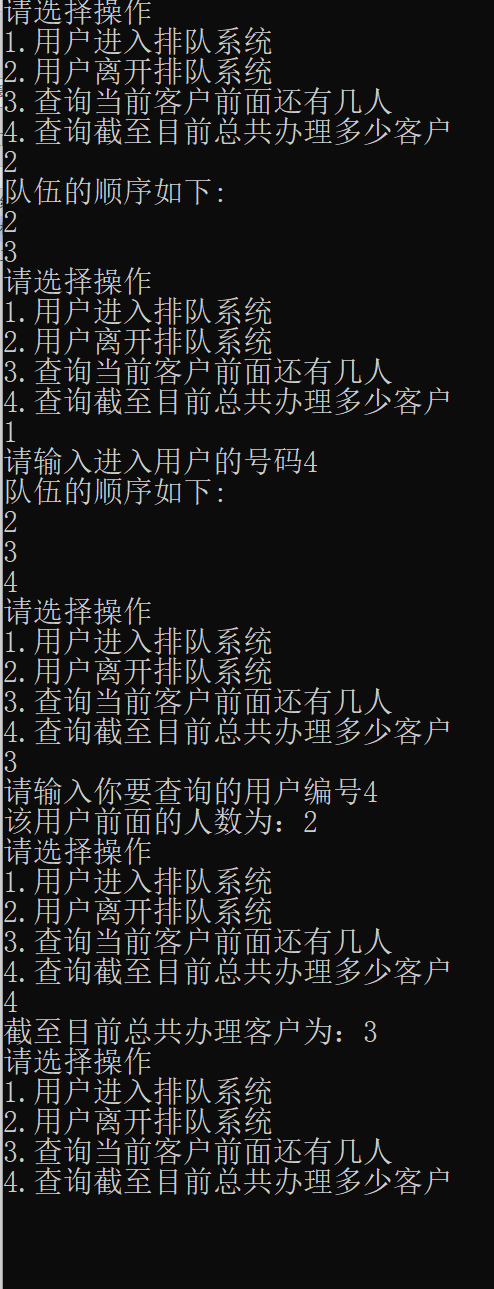


欢迎来到南南的银行排队系统！

初始会进入一个循环，你可以输入四个编号来选择你要执行的操作。

当银行排队系统排到四个人之后开始进行操作：入队、出队、统计人数、统计当前。





实验三 二叉树的应用

1. 实验目的：
2. 掌握二叉树的定义和存储表示，掌握二叉树建立的算法；
3. 掌握二叉树的遍历（先序、中序、后序）算法
4. 问题描述
5. 查找并绘制自己家族的族谱二叉树；
6. 族谱二叉树的建立（树的深度要>=4）；
7. 三种不同遍历算法遍历此二叉树；
8. 统计二叉树的深度，输出叶子结点的信息。

三、数据结构设计（选用的数据逻辑结构和存储结构实现形式说明）

（1）逻辑结构设计

我设计了一个树的结构，通过遍历输出结果。

（2）存储结构设计

给树的每个节点依次分配存储空间，每当需要使用时递归调用生成节点。

（3）存储结构形式说明

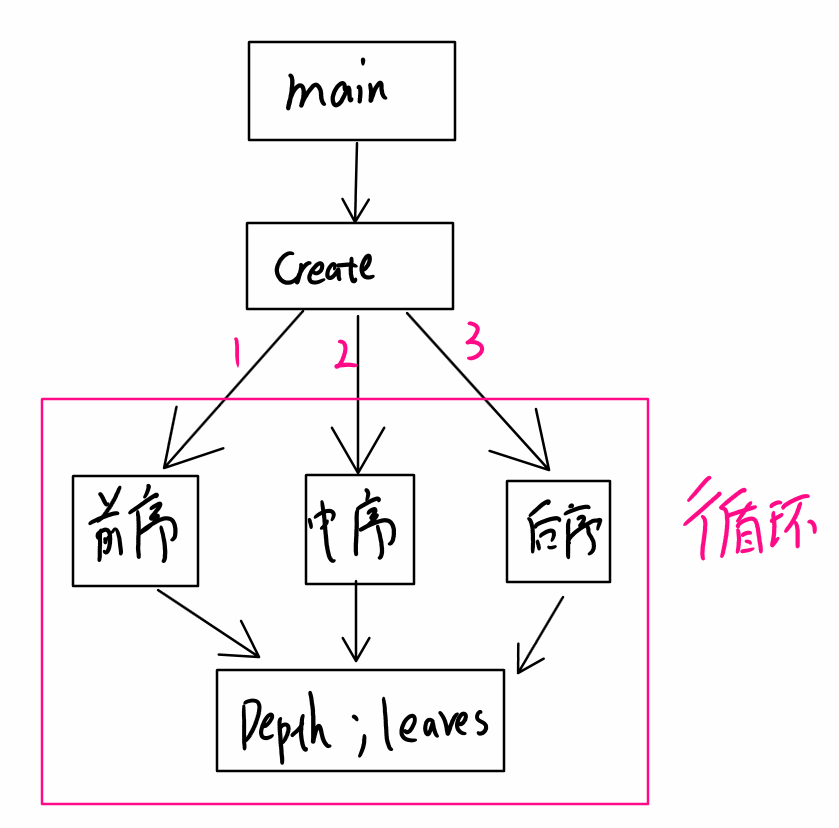
将每一个节点分成三个部分：左孩子指针、数据域、右孩子指针。需要使用的时候进行调用即可。

四、算法设计

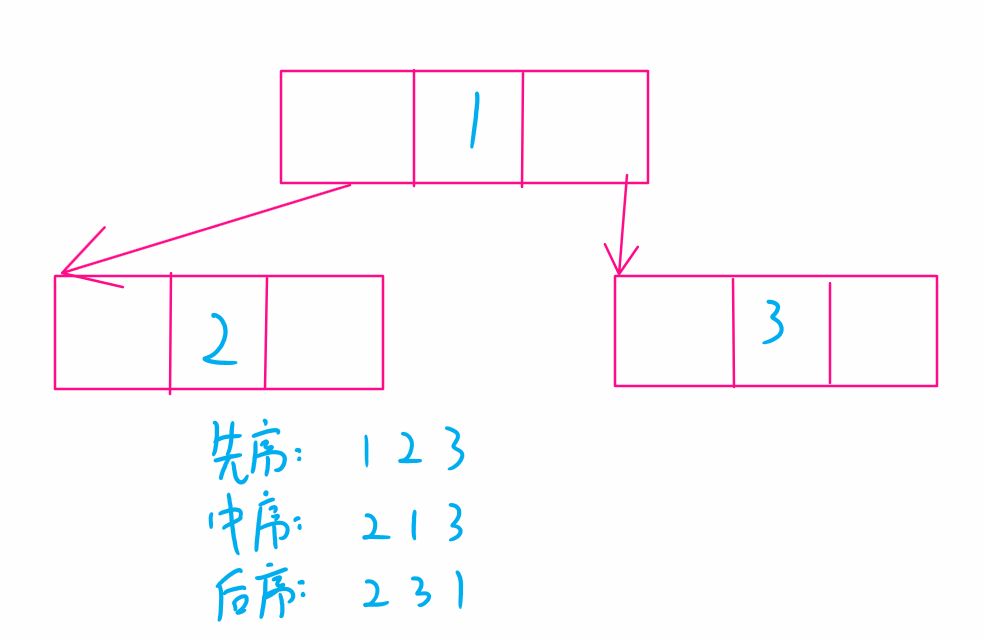
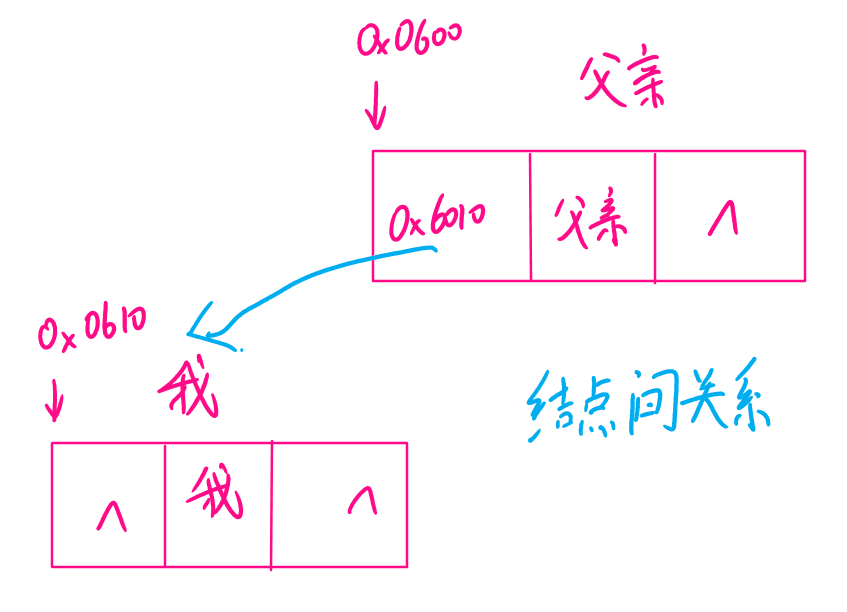
（1）算法列表（说明各个函数的名称，作用，完成什么操作）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 创建二叉树 | CreateBiTree | 递归先序创建二叉树 |
| 2 | 先序遍历 | BeforeTheOrderTraverse | 先序遍历整个二叉树 |
| 3 | 中序遍历 | InOrderTraverse | 中序遍历整个二叉树 |
| 4 | 后序遍历 | BehindTheOrderTraverse | 后序遍历整个二叉树 |
| 5 | 树深 | Depth | 计算树的深度 |
| 6 | 叶子统计 | LeafCount | 统计目前叶子的数目 |

（2）各函数间调用关系（画出函数之间调用关系）



（3）算法描述



五、调试记录（调试过程中遇到的主要问题，是如何解决的，对设计和编码的回顾讨论和分析；改进设想等）

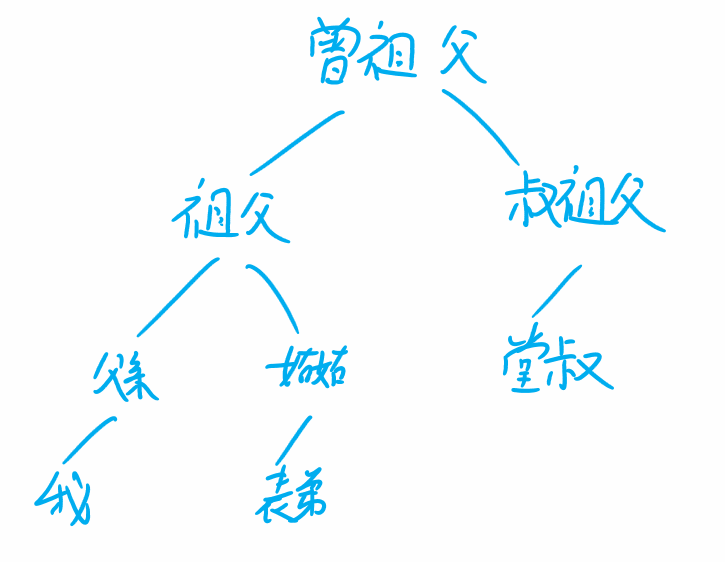
第一至六次调试失败：没有理解函数的递归调用；

第七次调试成功：可以先序遍历创建二叉树，可以先序遍历输出各个节点元素；

第八次调试成功：可以计算树的深度以及叶子数目。

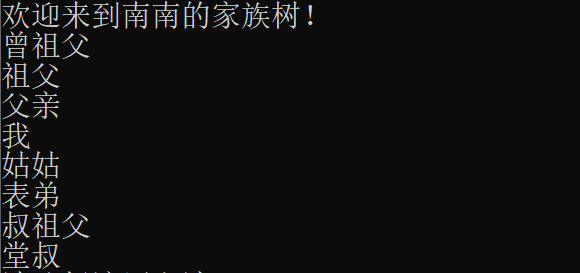
六、运行说明（列出测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于示例中所列数据）

族谱如下图所示：

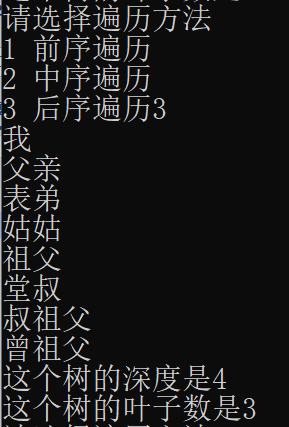
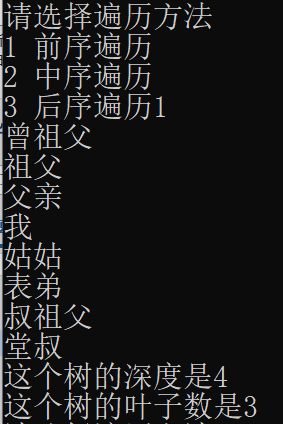


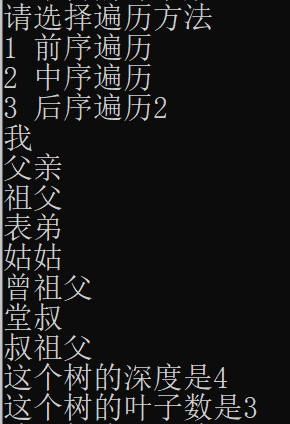
欢迎来到南南的家族树！

先序遍历创建：



先序遍历 中序遍历 后序遍历





实验四 图的应用

1. 实验目的：
2. 掌握图的基本概念和存储方法；
3. 掌握图的遍历算法，最短路径算法。
4. 问题描述
5. 绘制基于理工的地图网（结点不少于6），注：边的权值代表距离；实现网的创建；
6. 按照深度遍历和广度遍历算法输出结点信息；
7. 实现从西门到香樟的最短路径算法。

三、数据结构设计（选用的数据逻辑结构和存储结构实现形式说明）

（1）逻辑结构设计

我设计了一个图的结构，将理工大学的地图进行存储。

（2）存储结构设计

两个二维数组、一个队列。

（3）存储结构形式说明

先采用一个二维数组存储所有节点信息，然后再采用邻接矩阵的方法创建一张无向网。

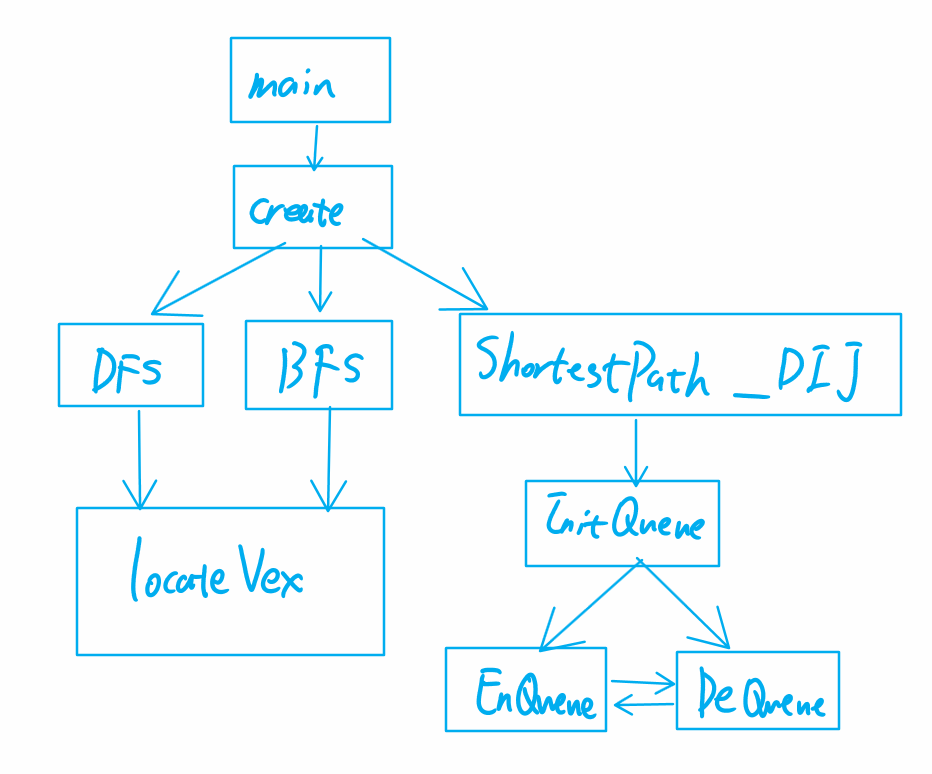
在进行最短路径的算法时，要构造一个队列进行计算。

四、算法设计

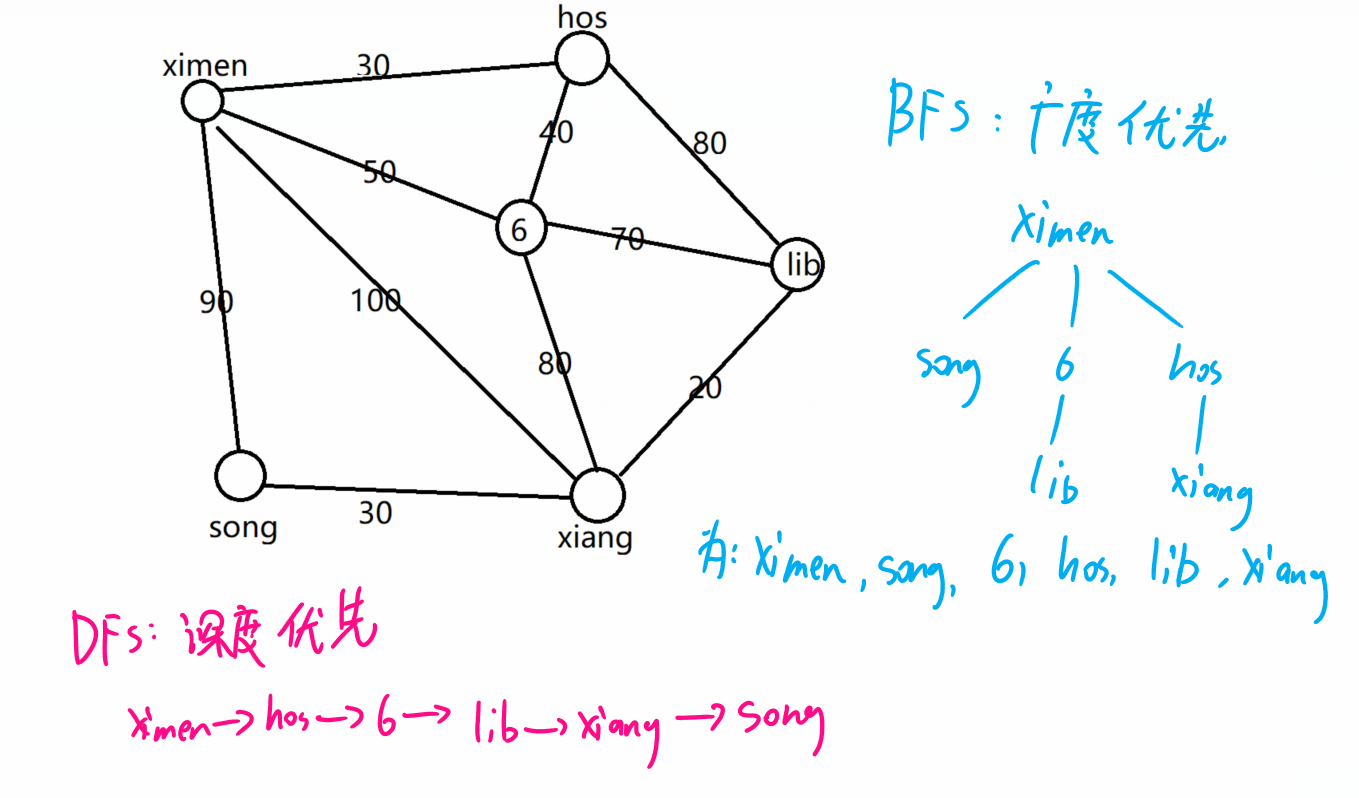
（1）算法列表（说明各个函数的名称，作用，完成什么操作）

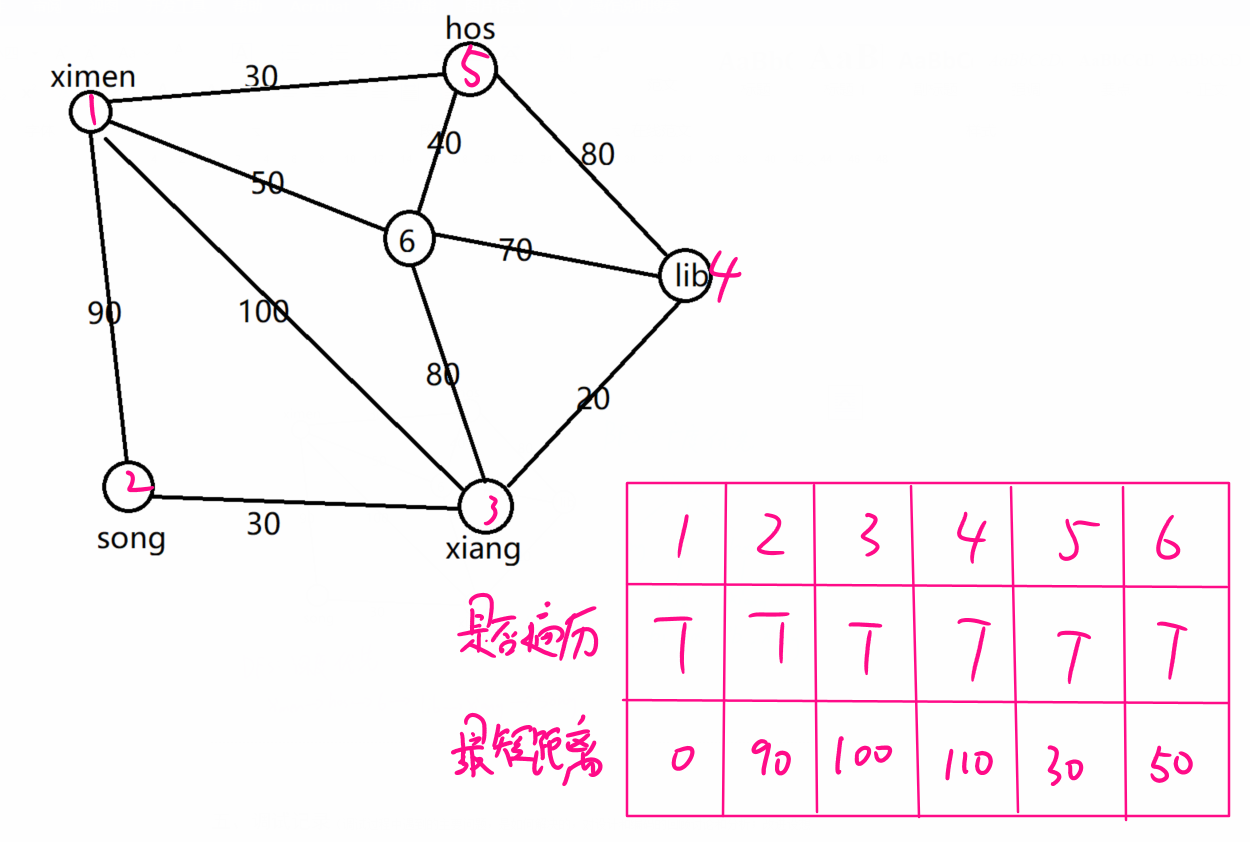
|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 定位 | LocateVex | 找到相应的邻接矩阵的位置 |
| 2 | 创建 | CreateUDN | 邻接矩阵法创建无向网 |
| 3 | 深度优先遍历 | DFS\_AM | 对整个图进行深度遍历操作 |
| 4 | 广度优先遍历 | BFS\_AM | 对整个图进行广度遍历操作 |
| 5 | 迪杰斯特拉 | ShortestPath\_DIJ | 用迪杰斯特拉算法计算最短路径 |
| 6 | 初始化 | InitQueue | 初始化辅助队列 |
| 7 | 入队 | EnQueue | 在队尾插入元素 |
|  | 出队 | DeQueue | 在队头删除元素 |

（2）各函数间调用关系（画出函数之间调用关系）



（3）算法描述





五、调试记录（调试过程中遇到的主要问题，是如何解决的，对设计和编码的回顾讨论和分析；改进设想等）

第一次至第七次调试失败：图的逻辑结构和存储结构过于复杂，难以捉摸；

第八次调试成功：成功创建出无向图，并输出第一个节点；

第九次至第十二次调试失败：无法完成深度优先遍历的过程；

第十三次调试成功：成功完成深度优先遍历以及广度优先遍历；

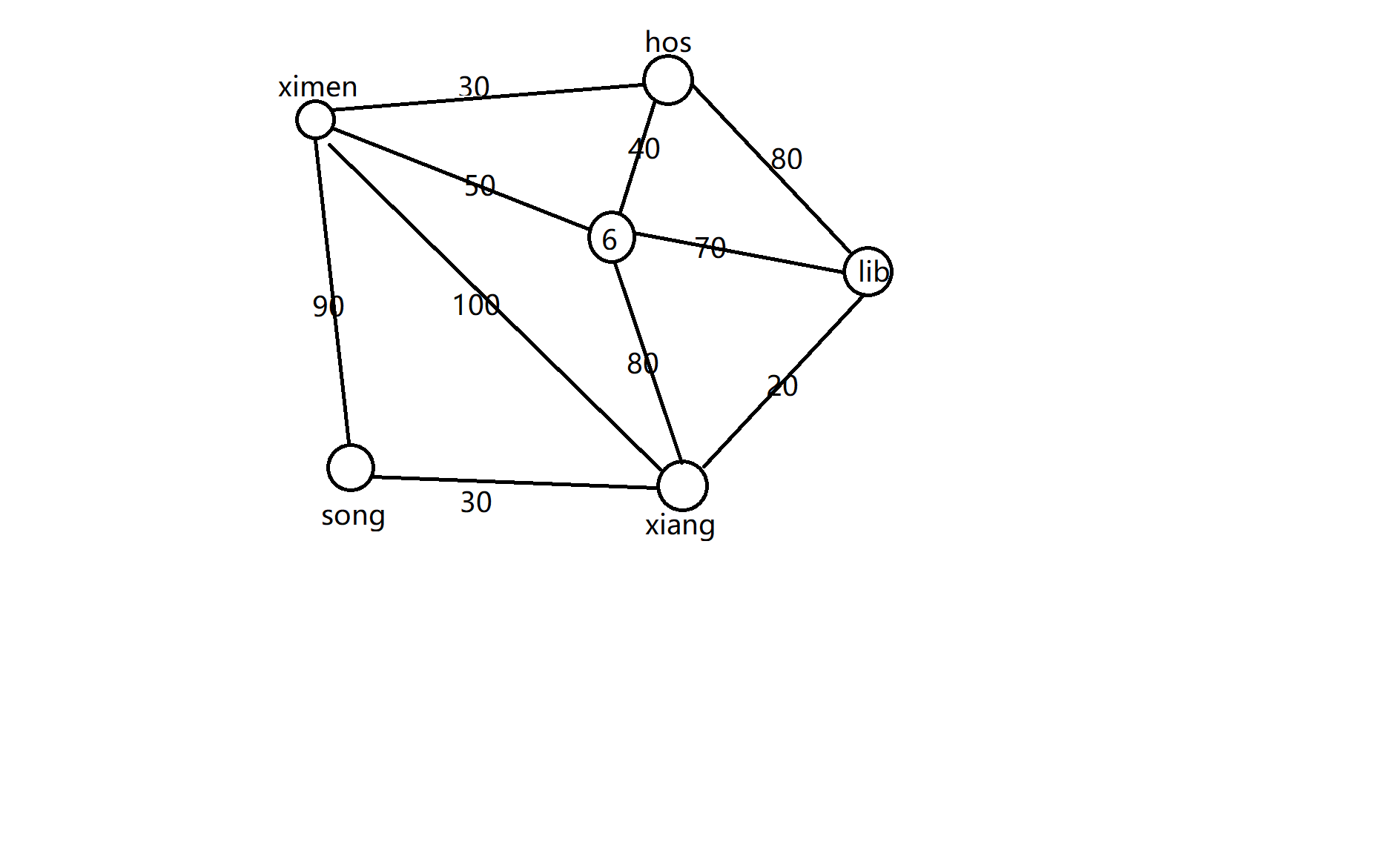
第十四至第二十五次调试失败：地界特拉斯算法需要一些自己写的函数，我在自己写的时候遇到了些麻烦。

第二十六次调试成功：麻烦得到了解决，我成功完成了各项功能。

六、运行说明（列出测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于示例中所列数据）

欢迎来到追梦少年南南的理工地图网！

我绘制的理工大学地图网如下图所示：



首先输入我们的顶点数：6个顶点。

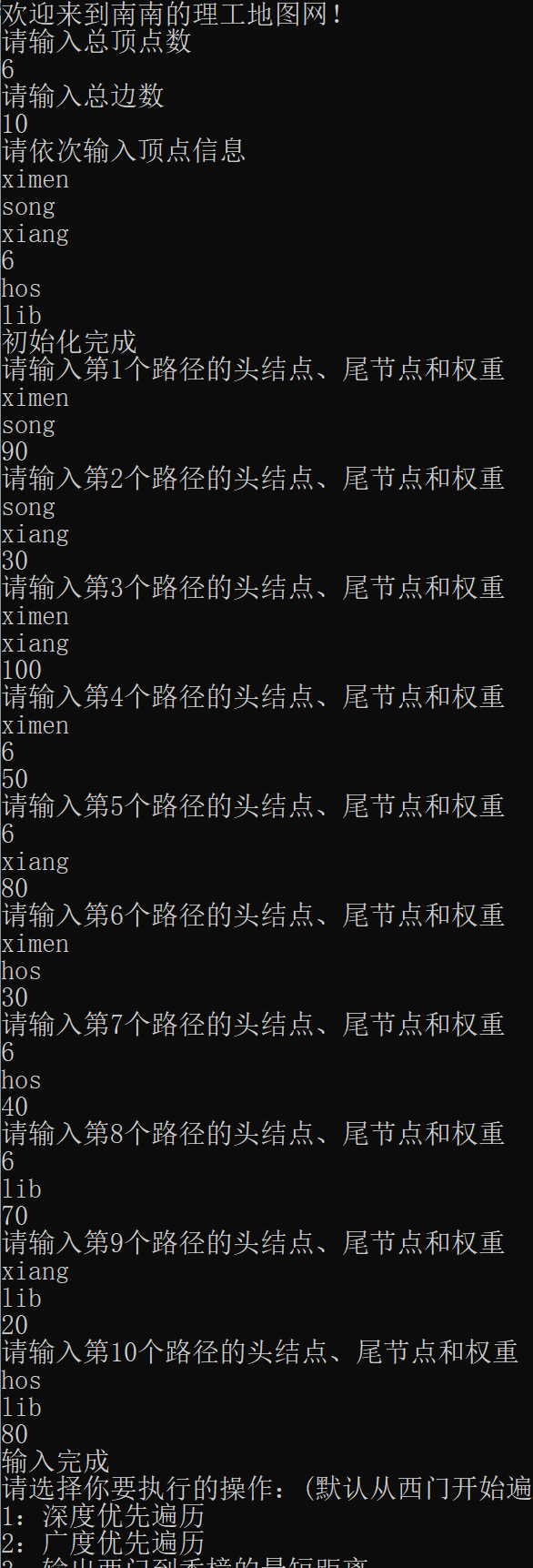
然后输入他们分别是什么：ximen song xiang 6 lib hos；

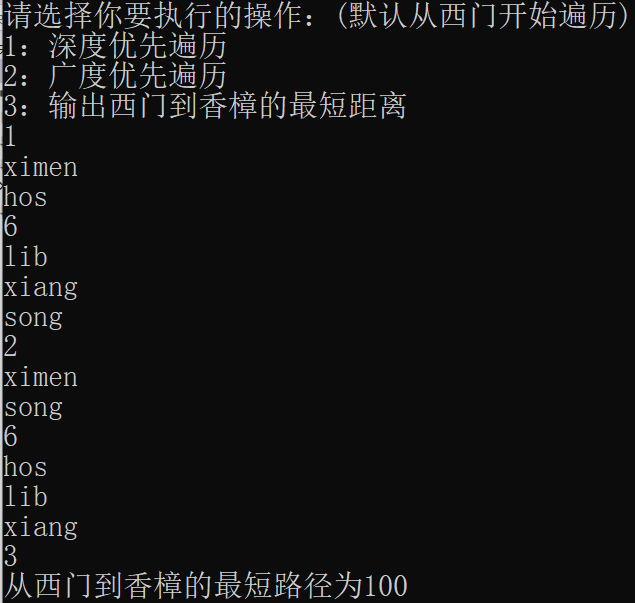
分别代表：西门，松林，香樟，六教，图书馆，校医院。

然后依次输入这几个点的信息，以便构成邻接矩阵。

然后对邻接矩阵进行深度优先遍历以及广度优先遍历。

最后用迪杰斯特拉算法求出最短路径长度即可。





实验五 查找的应用

1. 实验目的：
2. 掌握各种查找方法及适用场合，并能在解决实际问题时灵活应用。
3. 增强上机编程调试能力。
4. 问题描述
5. 分别利用顺序查找和折半查找方法完成查找。

有序表（3,4,5,7,24,30,42,54,63,72,87,95）

输入示例：

请输入查找元素：52

输出示例：

顺序查找：

第一次比较元素95

第二次比较元素87 ……..

查找成功，i=\*\*/查找失败

折半查找：

第一次比较元素30

第二次比较元素63 …..

1. 利用序列（12,7,17,11,16,2,13,9,21,4）建立二叉排序树，并完成指定元素的查询。

输入输出示例同题1的要求。

三、数据结构设计（选用的数据逻辑结构和存储结构实现形式说明）

（1）逻辑结构设计

设计了一个顺序表的结构来进行查找操作；以及一个树的结构来进行查找操作

（2）存储结构设计

分配了一系列存储空间。

（3）存储结构形式说明

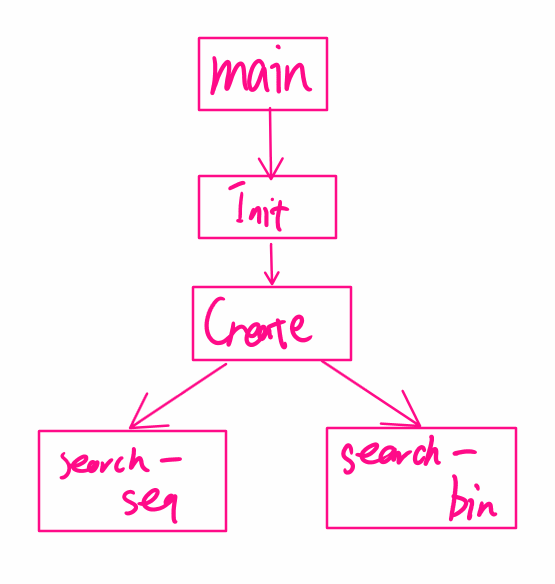
将想要查找的一系列数存到我构建的队列中，再存储一个待查找的元素。或者是将待查找的元素排成二叉树的形式来进行查找。

四、算法设计

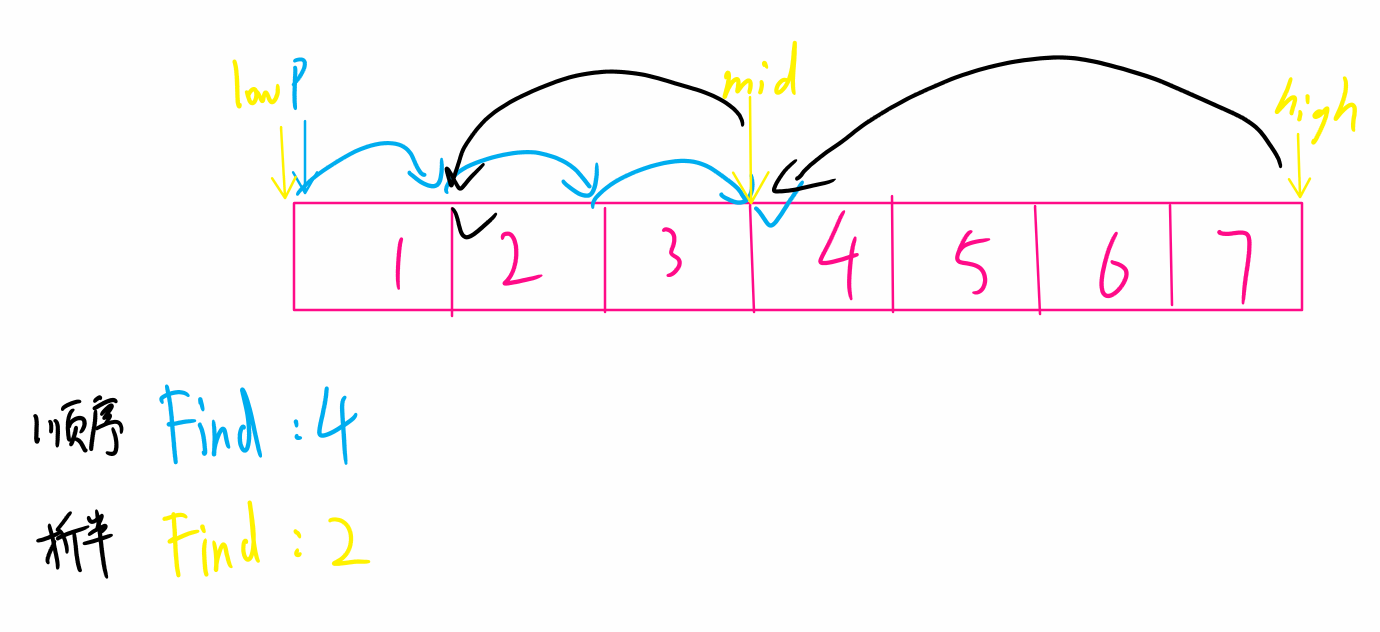
（1）算法列表（说明各个函数的名称，作用，完成什么操作）

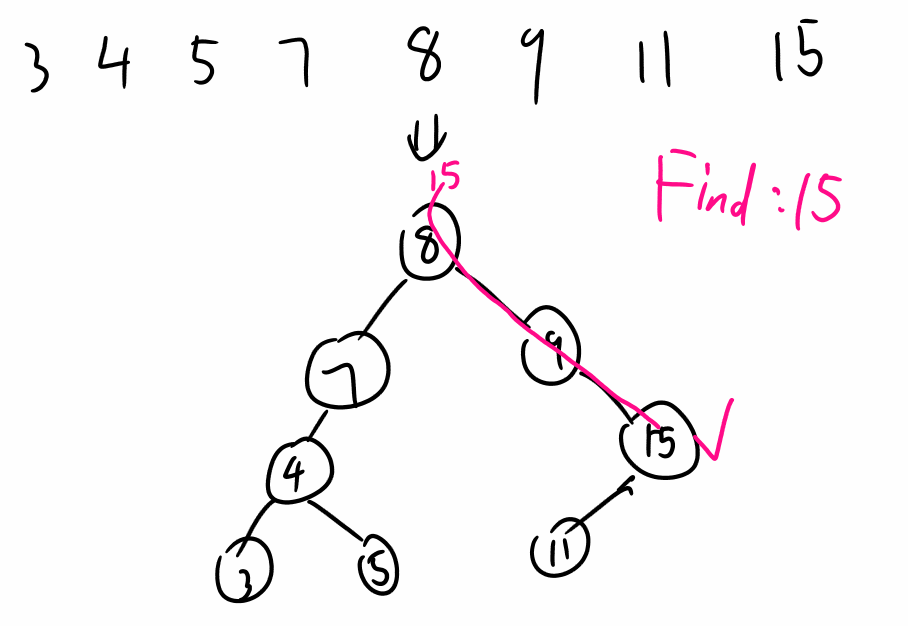
|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 初始化 | InitList | 初始化线性表 |
| 2 | 输入 | ListCreate | 按顺序输入顺序表 |
| 3 | 顺序查找 | Search\_Seq | 对顺序表进行顺序查找 |
| 4 | 折半查找 | Search\_Bin | 对顺序表进行折半查找 |
| 5 | 主函数 | Main | 主函数 |
| 6 | 创建 | InsertBST | 中序遍历创建而二叉树 |
| 7 | 查找 | SearchBST | 在二叉树中进行查找 |

（2）各函数间调用关系（画出函数之间调用关系）



（3）算法描述





五、调试记录（调试过程中遇到的主要问题，是如何解决的，对设计和编码的回顾讨论和分析；改进设想等）

第一次调试成功：完成表格的创建；

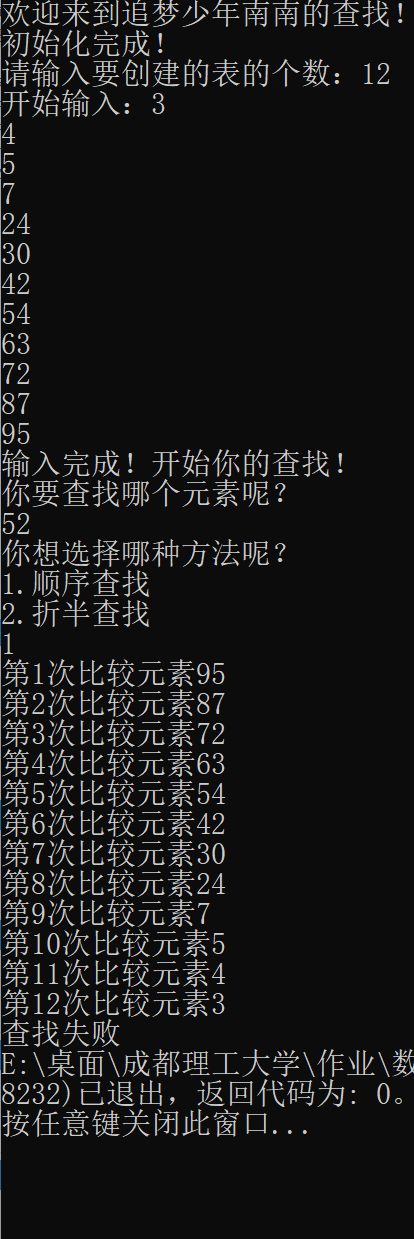
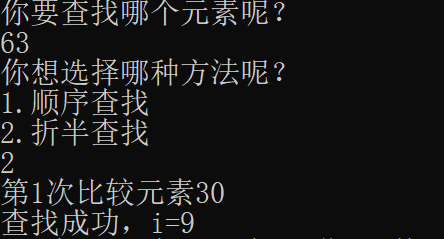
第二次调试成功：完成顺序查找以及折半查找；

第三次调试失败：创建二叉树失败； d

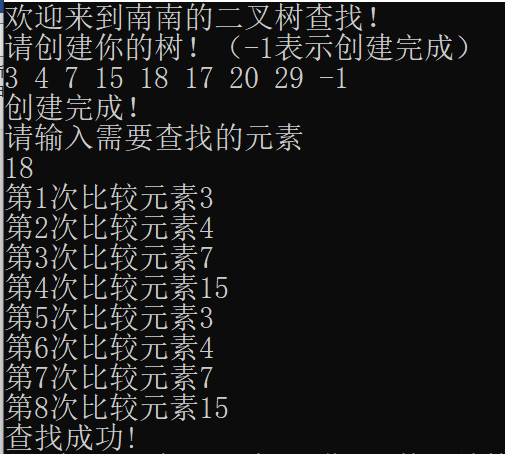
六、运行说明（列出测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于示例中所列数据）

欢迎来到追梦少年南南的查找！

首先输入要查找的数列，然后输入数据进行查找即可。

欢迎来到南南的二叉树查找！



实验六 排序的应用

1. 实验目的：
2. 掌握直接插入排序、折半插入排序、冒泡排序、快速排序和归并排序等排序算法的思想。
3. 实现直接插入排序、折半插入排序、冒泡排序、快速排序和归并排序等排序算法的编程应用。
4. 问题描述

实现数据的折半插入排序、冒泡排序、快速排序和二路归并排序。

输入实例：

请输入待排序数据数目：

3

请输入待排序数据：23,6,45

输出示例：

折半插入排序：

比较次数

移动元素次数

排序结果6,23,45。

三、数据结构设计（选用的数据逻辑结构和存储结构实现形式说明）

（1）逻辑结构设计

我用一个线性表存储数据进行排序。

（2）存储结构设计

给需要存储的每一个数据分配一个存储空间。

（3）存储结构形式说明

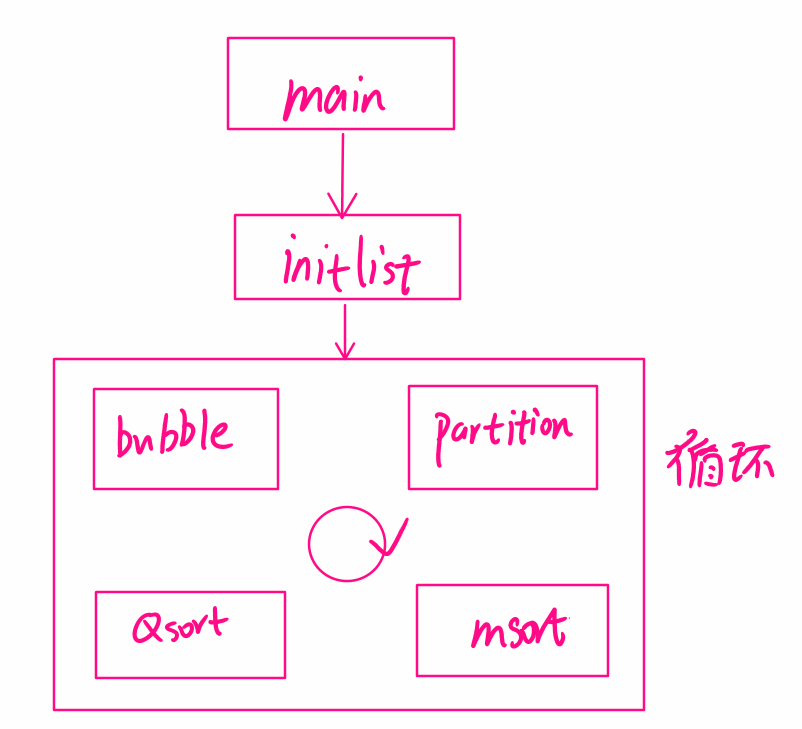
需要排序的数据给予其存储空间，然后对其进行排序即可。

四、算法设计

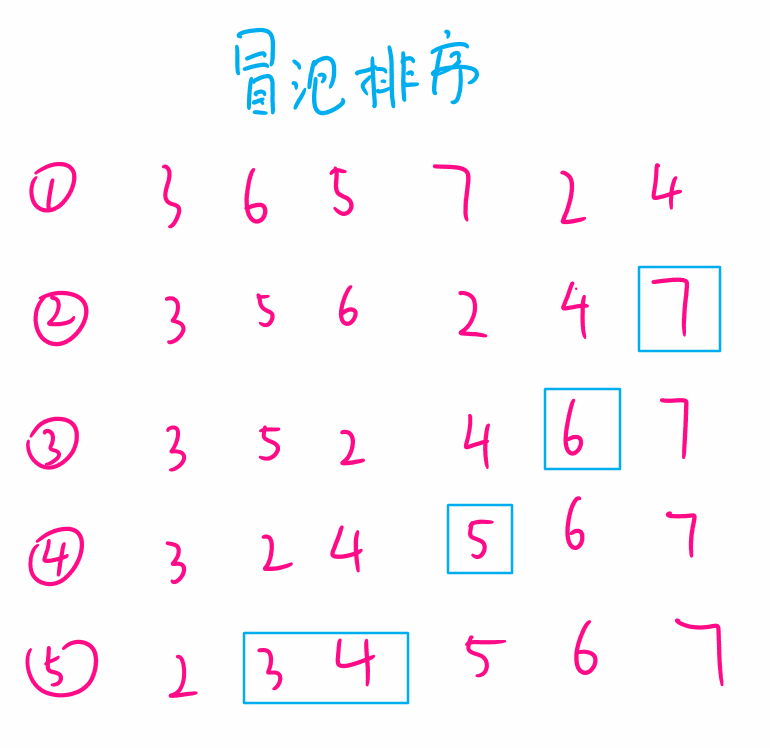
（1）算法列表（说明各个函数的名称，作用，完成什么操作）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 函数表示符 | 操作说明 |
| 1 | 初始化 | InitList | 初始化线性表 |
| 2 | 折半插入排序 | BInsertSort | 折半插入排序 |
| 3 | 冒泡排序 | BubbleSort | 冒泡排序 |
| 4 | 快速排序 | Partition | 快速排序 |
| 5 | 快速排序递归 | QSort | 递归快速排序 |
| 6 | 二路归并排序 | MSort | 二路归并排序 |

（2）各函数间调用关系（画出函数之间调用关系）



（3）算法描述（各算法简要描述）



五、调试记录（调试过程中遇到的主要问题，是如何解决的，对设计和编码的回顾讨论和分析；改进设想等）

第一次调试成功：成功构造出顺序表；

第二次调试失败：语法出现错误；

第三次调试成功：实现冒泡排序；

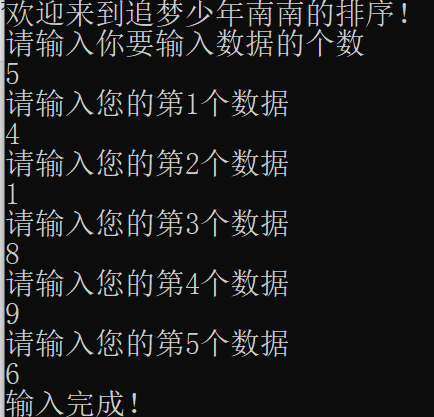
第四次调试失败：没能实现别的排序方法；

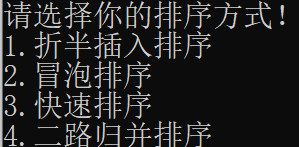
第五次调试成功：实现了各种排序功能；

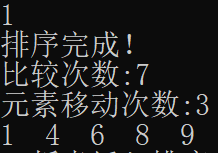
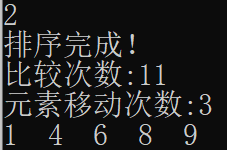
第六次排序成功：实现了排序中添加过程。

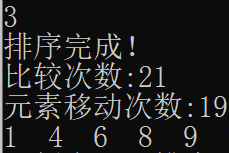
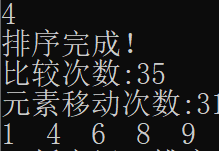
六、运行说明（列出测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于示例中所列数据）

欢迎来到追梦少年南南的排序！





|  |  |
| --- | --- |
| **学生实验 心得** | 在数据结构的学习过程中，我掌握了一些基本的数据结构的构成方法，一些常用的基本的算法，以及一些强化过后的算法等等。在此基础上，我也理解了一些关于计算机底层的结构，可谓是受益匪浅。  在完成实验时，我秉持着求真务实的态度，不懂就问，我也通过自己的学习写出了属于自己的程序。当程序成功运行的那一刻，我认为我的学习取得了成效。  最后，我要感谢朱杰老师以及和我一起讨论交流的小伙伴们。在我彷徨无助的时候给予我帮助，在我困惑不解时帮助我理解。正因为有你们，才有了我的完整的实验过程。  学生（签名）：  2022年7月7日 |
| **指导**  **教师**  **评语** | 成绩评定：  指导教师（签名）：  年 月 日 |