数据结构算法汇总

[第一部分·实验教程 4](#_Toc85276741)

[第五章·链表结构算法设计技术 4](#_Toc85276742)

[5-1单链表的遍历。要求设计算法依次输出链表中所有结点的值。 4](#_Toc85276743)

[5-2设计算法求单链表的长度。 4](#_Toc85276744)

[5-3设计算法以判断链表L是否是递增的。若递增，返回TRUE，否则返回FALSE。 5](#_Toc85276745)

[5-4设计算法实现单链表的逆置，即将无头结点的单链表L就地逆置，也就是要求利用原结点空间将其逆置。 5](#_Toc85276746)

[5-5设两个链表L1和L2分别表示两个集合，设计算法以判断集合L1是否是集合L2的子集，即判断L1中的元素是否是集合L2的子集，即判断L1中的元素是否都是L2中的元素。若成立，则返回TRUE；否则返回FALSE。 6](#_Toc85276747)

[5-6判断带头结点的双循环链表L是否为对称的。 6](#_Toc85276748)

[5-7设计算法将带头结点的双循环链表逆置。 7](#_Toc85276749)

[5-8设计算法将一个带头结点的单链表A分解成两个链表A和B，使A表中含有原表中奇数项的结点，B表中含有原表中偶数项的结点，且保持结点间原有的相对顺序。 7](#_Toc85276750)

[5-9设链表中结点的值都是整型数，设计算法将一个带头结点的单链表A分解为两链表A、B，使A表中结点元素值均为奇数、B表中结点元素值均为偶数，且保持结点间原有的相对次序。 8](#_Toc85276751)

[5-10已知A、B为两个递增有序的单链表，试设计算法利用原结点空间将其合并为一个递增有序链表C。 8](#_Toc85276752)

[5-11设计一个基于表头插入法建表方法实现单链表逆置的算法。 9](#_Toc85276753)

[5-12已知A、B为两个递增有序的单链表，试设计算法利用原结点空间将其合并为一个递减有序链表C，并要求算法的时间复杂度为O(|A|+|B|)。 9](#_Toc85276754)

[第六章·二叉树和树结构算法设计技术 10](#_Toc85276755)

[6-1设计算法按先序方式输出二叉树T中叶子结点的值。 10](#_Toc85276756)

[6-2设计算法求二叉树T的结点数。 10](#_Toc85276757)

[6-3设计算法求出二叉树中度为2的结点。 11](#_Toc85276758)

[6-4设计算法按后序方式输出二叉树T的前K个结点的值（K>=0）. 11](#_Toc85276759)

[6-5求两个整数M和N的最大公因子(M,N)的递归函数int hcf(int M,int N)。 12](#_Toc85276760)

[6-6设要求以T为根指针的二叉树的结点数，函数原型为int nodes(Bnode \*T)。 12](#_Toc85276761)

[6-7设函数原型为void nodes(Bnode \*T,int &n))，其功能是“求以T为根的二叉树的结点数并赋予参数n”。 13](#_Toc85276762)

[6-8设函数原型为void nodes(Bnode \*T,int &n)，其功能描述是：“将以T为根的整个二叉树的结点数累加到参数n中”。因此，在主函数中调用算法前要将n设置为0。 13](#_Toc85276763)

[6-9设函数原型为void nodes(Bnode \*T)，其功能描述是：“将以T为根的整个二叉树的结点数累加到全程变量n中”。因此，在主函数中调用算法前要将n设置为0。 14](#_Toc85276764)

[6-10设函数原型为void nodes(Bnode \*T)，其功能描述是：“将以T为根的整个二叉树的结点数并赋给全程变量n”。由此可知，n不必事先置为0。 14](#_Toc85276765)

[6-11设计算法求二叉树的叶子结点数目 15](#_Toc85276766)

[6-12设计算法以求二叉树T的高度 15](#_Toc85276767)

[6-13设计算法以释放二叉树的所有结点的存储空间 16](#_Toc85276768)

[6-14设计算法以判断二叉树T是否是二叉排序树 16](#_Toc85276769)

[6-15设计算法求树（或森林）的叶子结点数 17](#_Toc85276770)

[6-16设计算法求树（或森林）的高度 17](#_Toc85276771)

[6-17设计算法，以扩展二叉树的先序序列作为输入数据来构建二叉树 18](#_Toc85276772)

[6-18建立二分查找的判定树，已知查找的元素下标范围为1~n 18](#_Toc85276773)

[6-19一棵完全二叉树以顺序方式存储在数组A[1+n]中，设计算法构造出该二叉树的二叉链表T 19](#_Toc85276774)

[6-20已知二叉树的先序序列和中序序列分别存储在数组pre[n]和pin[n]中，设计算法构造该二叉树的二叉链表T 19](#_Toc85276775)

[6-21设计算法将一棵以二叉链表形式存储的二叉树按顺序方式存储到数组A[]中 20](#_Toc85276776)

[6-22已知数组A的每个元素有两个分量data和brother，用于存储树或森林的相关信息，其存储方式如下： 20](#_Toc85276777)

[第七章·基于图的遍历的算法设计技术 21](#_Toc85276778)

[7-1设计算法以判断有向图G是否是一棵以v0为根的有向树。 21](#_Toc85276779)

[7-2设计算法在图中找出一条包含所有顶点的简单路径。 21](#_Toc85276780)

[7-3试设计算法将整数1,2,3，…，n（不妨设为20）放入一个环中，使任意两个相邻数之和为质数。 22](#_Toc85276781)

[7-5设计算法求出距离顶点v0的最短路径长度（以弧数为单位）为最长的所有顶点，要求尽可能节省时间。 22](#_Toc85276782)

[7-5设计算法求出距离顶点v0的最短路径长度（以弧数为单位）为k的所有顶点，要求尽可能节省时间。 23](#_Toc85276783)

[7-6设计算法求顶点v0到图中其余每个顶点的最短路径，要求尽可能节省时间。 23](#_Toc85276784)

[第八章·面向数组的典型算法设计技术 24](#_Toc85276785)

[8-1已知数组A[n]的元素类型为整型，试设计算法将数组划分为左右两部分，使得左边元素为奇数，右边元素为偶数，并要求算法的时间复杂度为o(n)。 24](#_Toc85276786)

[8-2设计算法将数组A[n]循环右移k位，并要求时间复杂度为o(n)。只能用一个元素的辅助空间。 24](#_Toc85276787)

[第二部分·真题篇 25](#_Toc85276788)

[真题2014 25](#_Toc85276789)

[1、设计算法题以实现对无向图G的深度遍历，要求：将每个连通分量中的顶点以一个表的形式输出。例如：图2的输出结果为(1,3)(2,6,7,4,5,8)(9,10)。 25](#_Toc85276790)

[2、设计一个非递归算法以输出二叉树t中先序序列中最后一个结点的值。 25](#_Toc85276791)

[3、给定顺序表A，给出快速排序的算法实现过程（递归或非递归都可），并给出最好时间性能分析。 26](#_Toc85276792)

[真题2015 26](#_Toc85276793)

[1、已知递增有序单链表A、B分别表示一个集合，设计算法求A、B的交集A=A∩B，要求所花费时间尽可能少。 26](#_Toc85276794)

[2、设计算法按先序次序输出二叉树中每个结点的值及其所对应的层次数。 27](#_Toc85276795)

[3、设计算法以判断无向图G顶点vi到vj是否存在路径，若存在，返回true，否则返回false。 27](#_Toc85276796)

[真题2016 28](#_Toc85276797)

[1、假设递增有序的链表L表示一个集合，试设计算法在表中插入一个值为x的元素结点，使其仍然保持递增有序。 28](#_Toc85276798)

[2、设计算法将以二叉链表T存储的二叉树转换为对应的顺序存储结构A[max]中。要求：空的元素用null表示，并返回所存储的最大的元素下标。 28](#_Toc85276799)

[3、设计算法判断无向图G是否是一棵树，若是则返回TRUE，否则返回FALSE。 29](#_Toc85276800)

[真题2017 29](#_Toc85276801)

[1、设计算法将单链表L逆置。 29](#_Toc85276802)

[2、设计算法以递增有序数组int A[n]中元素为输入数据，构造一棵平衡的二叉树。 30](#_Toc85276803)

[3、设计算法以判断有向图G中是哦福存在一条顶点v0到vi路径。若存在返回TRUE，否则返回FALSE。 30](#_Toc85276804)

[真题2018 31](#_Toc85276805)

[1、已知递增有序单链表A、B分别表示一个集合，设计算法求C=A∩B。 31](#_Toc85276806)

[2、设计算法将二叉树的二叉链表存储结构转换为对应的顺序表结构，并存储到数组A[n]中。要求： 31](#_Toc85276807)

[3、设计算法以判断无向图G是否连通，若连通则返回TRUE，否则返回FALSE。 32](#_Toc85276808)

[真题2019 32](#_Toc85276809)

[1、设计算法在递增有序带头结点的单链表中，插入一个值为x的结点，并保持递增有序的次序，如果表中存在值相同的元素，则不必执行插入操作。 32](#_Toc85276810)

[2、设计一个按先序次序遍历先序线索二叉树的非递归算法，并不许用栈。 33](#_Toc85276811)

[3、设计算法判断有向图G中顶点v0到每一个顶点是否都有路径。若是，返回TRUE;否则返回FALSE。 33](#_Toc85276812)

[真题2020 34](#_Toc85276813)

[1、已知顺序表A中元素按非降次序排列，请设计一个高效的算法以删除A中重复出现的元素，并分析算法的时间复杂度。例如，对顺序表(2,2,5,5,5,8,8,9)执行算法得到的结果为(2,5,8,9) 34](#_Toc85276814)

[2、已知给定的二叉树T中各结点的值两两不同，设计算法以判断T是否是二叉排序树。若是，返回TRUE；否则，返回FALSE。 34](#_Toc85276815)

[3、在连通无向图G中，求出距离顶点v最短路径长度为最远的一个顶点。 35](#_Toc85276816)

[真题2021 35](#_Toc85276817)

[1、求递增有序的带头结点的单循环链表A、B分别表示两个集合，设计算法以求解A=A∪B。 35](#_Toc85276818)

[2、设计算法输出二叉树T的每个结点的值及其对应的层次数。 36](#_Toc85276819)

[3、设计算法以判断有向图G是否是一棵以V0为根的有向树。若是有向树，则返回TRUE，否则返回FALSE。 36](#_Toc85276820)

[第三部分·考试重点及模拟十三套卷核心算法题 37](#_Toc85276821)

# 第一部分·实验教程

## 第五章·链表结构算法设计技术

### 5-1单链表的遍历。要求设计算法依次输出链表中所有结点的值。

|  |
| --- |
|  |

### 5-2设计算法求单链表的长度。

|  |
| --- |
|  |

### 5-3设计算法以判断链表L是否是递增的。若递增，返回TRUE，否则返回FALSE。

|  |
| --- |
|  |

### 5-4设计算法实现单链表的逆置，即将无头结点的单链表L就地逆置，也就是要求利用原结点空间将其逆置。

|  |
| --- |
|  |

### 5-5设两个链表L1和L2分别表示两个集合，设计算法以判断集合L1是否是集合L2的子集，即判断L1中的元素是否是集合L2的子集，即判断L1中的元素是否都是L2中的元素。若成立，则返回TRUE；否则返回FALSE。

|  |
| --- |
|  |

### 5-6判断带头结点的双循环链表L是否为对称的。

|  |
| --- |
|  |

### 5-7设计算法将带头结点的双循环链表逆置。

|  |
| --- |
|  |

### 5-8设计算法将一个带头结点的单链表A分解成两个链表A和B，使A表中含有原表中奇数项的结点，B表中含有原表中偶数项的结点，且保持结点间原有的相对顺序。

|  |
| --- |
|  |

### 5-9设链表中结点的值都是整型数，设计算法将一个带头结点的单链表A分解为两链表A、B，使A表中结点元素值均为奇数、B表中结点元素值均为偶数，且保持结点间原有的相对次序。

|  |
| --- |
|  |

### 5-10已知A、B为两个递增有序的单链表，试设计算法利用原结点空间将其合并为一个递增有序链表C。

|  |
| --- |
|  |

### 5-11设计一个基于表头插入法建表方法实现单链表逆置的算法。

|  |
| --- |
|  |

### 5-12已知A、B为两个递增有序的单链表，试设计算法利用原结点空间将其合并为一个递减有序链表C，并要求算法的时间复杂度为O(|A|+|B|)。

|  |
| --- |
|  |

## 第六章·二叉树和树结构算法设计技术

### 6-1设计算法按先序方式输出二叉树T中叶子结点的值。

|  |
| --- |
|  |

### 6-2设计算法求二叉树T的结点数。

|  |
| --- |
|  |

### 6-3设计算法求出二叉树中度为2的结点。

|  |
| --- |
|  |

### 6-4设计算法按后序方式输出二叉树T的前K个结点的值（K>=0）.

|  |
| --- |
|  |

### 6-5求两个整数M和N的最大公因子(M,N)的递归函数int hcf(int M,int N)。

|  |
| --- |
|  |

### 6-6设要求以T为根指针的二叉树的结点数，函数原型为int nodes(Bnode \*T)。

|  |
| --- |
|  |

### 6-7设函数原型为void nodes(Bnode \*T,int &n))，其功能是“求以T为根的二叉树的结点数并赋予参数n”。

|  |
| --- |
|  |

### 6-8设函数原型为void nodes(Bnode \*T,int &n)，其功能描述是：“将以T为根的整个二叉树的结点数累加到参数n中”。因此，在主函数中调用算法前要将n设置为0。

|  |
| --- |
|  |

### 6-9设函数原型为void nodes(Bnode \*T)，其功能描述是：“将以T为根的整个二叉树的结点数累加到全程变量n中”。因此，在主函数中调用算法前要将n设置为0。

|  |
| --- |
|  |

### 6-10设函数原型为void nodes(Bnode \*T)，其功能描述是：“将以T为根的整个二叉树的结点数并赋给全程变量n”。由此可知，n不必事先置为0。

|  |
| --- |
|  |

### 6-11设计算法求二叉树的叶子结点数目

|  |
| --- |
|  |

### 6-12设计算法以求二叉树T的高度

|  |
| --- |
|  |

### 6-13设计算法以释放二叉树的所有结点的存储空间

|  |
| --- |
|  |

### 6-14设计算法以判断二叉树T是否是二叉排序树

|  |
| --- |
|  |

### 6-15设计算法求树（或森林）的叶子结点数

|  |
| --- |
|  |

### 6-16设计算法求树（或森林）的高度

|  |
| --- |
|  |

### 6-17设计算法，以扩展二叉树的先序序列作为输入数据来构建二叉树

|  |
| --- |
|  |

### 6-18建立二分查找的判定树，已知查找的元素下标范围为1~n

|  |
| --- |
|  |

### 6-19一棵完全二叉树以顺序方式存储在数组A[1+n]中，设计算法构造出该二叉树的二叉链表T

|  |
| --- |
|  |

### 6-20已知二叉树的先序序列和中序序列分别存储在数组pre[n]和pin[n]中，设计算法构造该二叉树的二叉链表T

|  |
| --- |
|  |

### 6-21设计算法将一棵以二叉链表形式存储的二叉树按顺序方式存储到数组A[]中

|  |
| --- |
|  |

### 6-22已知数组A的每个元素有两个分量data和brother，用于存储树或森林的相关信息，其存储方式如下：

①整个树或森林的各结点信息按先序遍历顺序存储在数组的各元素中。

②数组中每个元素的data分量存放结点的值，brother分量存放该结点的右兄弟结点在数组中的下标，如果不存在右兄弟，则其内容为-1.

|  |
| --- |
|  |

## 第七章·基于图的遍历的算法设计技术

### 7-1设计算法以判断有向图G是否是一棵以v0为根的有向树。

|  |
| --- |
|  |

### 7-2设计算法在图中找出一条包含所有顶点的简单路径。

|  |
| --- |
|  |

### 7-3试设计算法将整数1,2,3，…，n（不妨设为20）放入一个环中，使任意两个相邻数之和为质数。

|  |
| --- |
|  |

### 7-5设计算法求出距离顶点v0的最短路径长度（以弧数为单位）为最长的所有顶点，要求尽可能节省时间。

|  |
| --- |
|  |

### 7-5设计算法求出距离顶点v0的最短路径长度（以弧数为单位）为k的所有顶点，要求尽可能节省时间。

|  |
| --- |
|  |

### 7-6设计算法求顶点v0到图中其余每个顶点的最短路径，要求尽可能节省时间。

|  |
| --- |
|  |

## 第八章·面向数组的典型算法设计技术

### 8-1已知数组A[n]的元素类型为整型，试设计算法将数组划分为左右两部分，使得左边元素为奇数，右边元素为偶数，并要求算法的时间复杂度为o(n)。

|  |
| --- |
|  |

### 8-2设计算法将数组A[n]循环右移k位，并要求时间复杂度为o(n)。只能用一个元素的辅助空间。

|  |
| --- |
|  |

# 第二部分·真题篇

## 真题2014

### 1、设计算法题以实现对无向图G的深度遍历，要求：将每个连通分量中的顶点以一个表的形式输出。例如：图2的输出结果为(1,3)(2,6,7,4,5,8)(9,10)。

|  |
| --- |
|  |

### 2、设计一个非递归算法以输出二叉树t中先序序列中最后一个结点的值。

|  |
| --- |
|  |

### 3、给定顺序表A，给出快速排序的算法实现过程（递归或非递归都可），并给出最好时间性能分析。

|  |
| --- |
|  |

## 真题2015

### 1、已知递增有序单链表A、B分别表示一个集合，设计算法求A、B的交集A=A∩B，要求所花费时间尽可能少。

|  |
| --- |
|  |

### 2、设计算法按先序次序输出二叉树中每个结点的值及其所对应的层次数。

|  |
| --- |
|  |

### 3、设计算法以判断无向图G顶点vi到vj是否存在路径，若存在，返回true，否则返回false。

|  |
| --- |
|  |

## 真题2016

### 1、假设递增有序的链表L表示一个集合，试设计算法在表中插入一个值为x的元素结点，使其仍然保持递增有序。

|  |
| --- |
|  |

### 2、设计算法将以二叉链表T存储的二叉树转换为对应的顺序存储结构A[max]中。要求：空的元素用null表示，并返回所存储的最大的元素下标。

|  |
| --- |
|  |

### 3、设计算法判断无向图G是否是一棵树，若是则返回TRUE，否则返回FALSE。

|  |
| --- |
|  |

## 真题2017

### 1、设计算法将单链表L逆置。

|  |
| --- |
|  |

### 2、设计算法以递增有序数组int A[n]中元素为输入数据，构造一棵平衡的二叉树。

|  |
| --- |
|  |

### 3、设计算法以判断有向图G中是哦福存在一条顶点v0到vi路径。若存在返回TRUE，否则返回FALSE。

|  |
| --- |
|  |

## 真题2018

### 1、已知递增有序单链表A、B分别表示一个集合，设计算法求C=A∩B。

|  |
| --- |
|  |

### 2、设计算法将二叉树的二叉链表存储结构转换为对应的顺序表结构，并存储到数组A[n]中。要求：

①数组中没有对应结点的元素值置为null。

②给出相应的初始化及调用操作。

|  |
| --- |
|  |

### 3、设计算法以判断无向图G是否连通，若连通则返回TRUE，否则返回FALSE。

|  |
| --- |
|  |

## 真题2019

### 1、设计算法在递增有序带头结点的单链表中，插入一个值为x的结点，并保持递增有序的次序，如果表中存在值相同的元素，则不必执行插入操作。

|  |
| --- |
|  |

### 2、设计一个按先序次序遍历先序线索二叉树的非递归算法，并不许用栈。

|  |
| --- |
|  |

### 3、设计算法判断有向图G中顶点v0到每一个顶点是否都有路径。若是，返回TRUE;否则返回FALSE。

|  |
| --- |
|  |

## 真题2020

### 1、已知顺序表A中元素按非降次序排列，请设计一个高效的算法以删除A中重复出现的元素，并分析算法的时间复杂度。例如，对顺序表(2,2,5,5,5,8,8,9)执行算法得到的结果为(2,5,8,9)

|  |
| --- |
|  |

### 2、已知给定的二叉树T中各结点的值两两不同，设计算法以判断T是否是二叉排序树。若是，返回TRUE；否则，返回FALSE。

|  |
| --- |
|  |

### 3、在连通无向图G中，求出距离顶点v最短路径长度为最远的一个顶点。

|  |
| --- |
|  |

## 真题2021

### 1、求递增有序的带头结点的单循环链表A、B分别表示两个集合，设计算法以求解A=A∪B。

|  |
| --- |
|  |

### 2、设计算法输出二叉树T的每个结点的值及其对应的层次数。

|  |
| --- |
|  |

### 3、设计算法以判断有向图G是否是一棵以V0为根的有向树。若是有向树，则返回TRUE，否则返回FALSE。

|  |
| --- |
|  |

# 第三部分·考试重点及模拟十三套卷核心算法题

1、栈、队列、线性表

①符号匹配问题与计算表达式问题

②带头结点、不带头结点的单链表就地逆置

③求两有序递增单链表的交集（可要求时间尽可能短或空间尽可能小）：A=A∩B C=A∩B

④求两有序递增单链表的并集（可要求时间尽可能短或空间尽可能小）：A=A∪B C=A∪B

⑤删除顺序表A中的重复元素

⑥在递增有序带头结点/不带头结点的单链表中插入一个值为x的结点。

⑦判断某单链表是否是单调递增的/是否当前所指元素小于后两元素之和（结点数>=2）/元素的值是否是序号的2倍。

2、数组和广义表

按一定的格式要求输出广义表中的元素

3、树

①以递归或非递归的形式先序、中序、后序遍历树/二叉树

②输出所有叶子结点的值、输出所有度为1/0/2的结点的值、输出遍历叶子结点的所在层次、求解树的高度

③先序、中序、后序线索二叉树的构建、前驱后继的查找以及遍历

④判断是否是二叉排序树

⑤将二叉链表转化为顺序表结构，并存储入A[n]中

⑥设计算法构造平衡二叉树

4、图

①判断两个顶点间是否存在路径（有向图，无向图），并打印路径

②判断图是否是无向图

③判断图是否是连通图/强连通图/完全连通图

④判断图是否是最小生成树

⑤广度优先遍历判断一个顶点到另一个顶点的最短路径（无权值）

⑥深度优先遍历的应用

5、排序算法

直接插入排序、希尔排序、冒泡排序、堆排序、直接选择排序、堆排序、归并排序的代码考察