**2013年全国硕士研究生统一入学考试自命题试题（副卷）**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

学科与专业名称：计算机技术，软件工程

考试科目代码与名称：830 数据结构

**一. 选择题(每题2分，共30分)**

1．在数据结构中，从逻辑上可以把数据分为（ **C** ）。

A. 动态结构和静态结构 B. 紧凑结构和非紧凑结构

C. 线性结构和非线性结构 D. 内部结构和外部结构

2. 设某无向图中有n个顶点e条边，则该无向图中所有顶点的度之和为（**D** ）。

A．n B. e C. 2n D. 2e

3. 在内部排序中,排序时不稳定的有( **C** )。

A. 插入排序 B. 冒泡排序 C. 快速排序 D. 归并排序

4. 在循环队列中，若front与rear 分别表示队头元素和队尾元素的位置，则判断循环队列空的条件是( **C** ) 。

A．front==rear+1 B．rear==front+1 C．front==rear D．front==0

5. 设单链表中指针p指着结点A,若要删除A之后的结点(若存在),则需要修改指针的操作为( **A** )。

A. p->next=p->next->next B. p=p->next

C. p=p->next->next D. p->next=p

6. 最坏情况下堆排序的时间复杂度是( **C** )。

A. O(log2n) B. O(log2n2) C. O(nlog2n) D. O(n2)

7. 设使用的邻接表表示某有向图，则顶点vj在表结点中出现的次数等于（ **C**  ）。

A. 顶点vj的度 B. 顶点vj的出度 C. 顶点vj的入度 D. 无法确定

8．树最适合用来表示（ **C** ）。

A．有序数据元素 B． 无序数据元素

C．元素之间具有分支层次关系的数据 D．元素之间无联系的数据

9．具有n个顶点的连通图至少应有（ **A**  ）条边。

A．n-1 B．n C．n(n-1)/2 D．2n

10. 时间复杂度不受数据初始状态影响而恒定为O(nlog2n）的是（  **A**  ）。

A．堆排序 B．冒泡排序 C．希尔排序 D．快速排序

11．任何一颗二叉树的叶子结点在前序、中序、后序遍历序列中的相对次序（ A ）。

A．不变 B．发生改变 C．不能确定 D．以上全不对

12. 一组记录（50，40，95，20，15，70，60，45，80）进行冒泡排序时，第一趟需进行相邻记录的交换的次数为（  **C**  ）。

A.5 B. 6 C. 7 D.8

13. 循环队列中是否可以插入下一个元素 （ **D**  ）。

A. 与曾经进行过多少次插入操作有关.

B. 只与队尾指针的值有关,与队头指针的值无关.

C. 只与数组大小有关,与队首指针和队尾指针的值无关

D. 与队头指针和队尾指针的值有关.

14. 某二叉树的先序遍历序列为 abdgcefh, 中序遍历序列为 dgbaechf， 则它的左子树的结点数目为（ **A** ）。

A．3 B．4 C．5 D．6

15. 对于元素是整数( 占2个字节)的对称矩阵A,采用以行序为主的压缩存储方式( 下三角),若A[0][0]的地址是400, 则元素A[8][5]的存储地址是（  **C** ）。

A.440 B. 480

C.482 D. 582

**二．填空题(每题2分，共20分)**

1．稀疏矩阵一般的压缩存储方法主要有两种，即 **三元组** 和 **十字链表**  。

2．线性结构中元素之间存在 **一对一** 的关系，树形结构中元素之间存在  **一对多** 的关系。

3．由n个权值构成的哈夫曼树共有  **2\*n-1**  个结点。

4．在散列表(hash)查找中，评判一个散列函数优劣的两个主要条件是： **出现冲突的概率**

和  **遇到冲突后的处理方法** 。

5．线索二叉树的左线索指向 **前驱结点** ，右线索指向 **后继结点**  。

6．在一棵二叉树中，度为零的结点的个数为n0，度为2 的结点的个数为n2，则该二叉树有

个叶子结点。

**7．**有一个100×90的稀疏矩阵，非0元素有10，设每个整型数占2个字节，则用三元组表示该矩阵时，所需的字节数是  **66**  。

8．带头结点的循环单链表L为空的条件是  **L->next==NULL**  。

9．设给定权值集合w={9,2,5,7} ，对应huffman树的加权路径长度WPL为  **44** 。

10．若某记录序列的关键字序列是（50，40，95，20，15，70），用简单选择法进行排序，第一次收集的结果是 (50,40,70,20,15,95) 。

**三．判断题（每题1分，共10分，正确的选t，错误的选f）**

1．采用邻接表存储的图的深度优先遍历相当于树的中序遍历。（ **×** ）

2．无向图的邻接矩阵一定是对称的。（ **√** ）

3．线性表中的每一个元素都有一个前驱和后继元素。（ **×** ）

4．B和B+树都能有效地支持随机查找。（ **√** ）

5．拓扑排序是按AOE网中每个结点事件的最早发生事件对结点进行排序。 （ **×** ）

6．一颗满二叉树同时又是一颗平衡树。（ **√** ）

7．对初始堆进行层次遍历可以得到一个有序序列。（ **×** ）

8. 冒泡排序是稳定的。( **√** )

9. 哈夫曼树中权值最小的结点离跟最近。（ **×** ）

10．带权无向图的最小生成树是唯一的。( **×** )

**四. 简答题（50分）**

1. 对图1.所示的有向带权图，使用Dijkstra（迪杰斯特拉）算法求出从顶点0到其余各顶点的最短路径， 要求写出过程。（10 分）

**4**

**10**

**100**

**30**

**50**

**20**

**60**

**10**

**3**

**2**

**0**

**1**

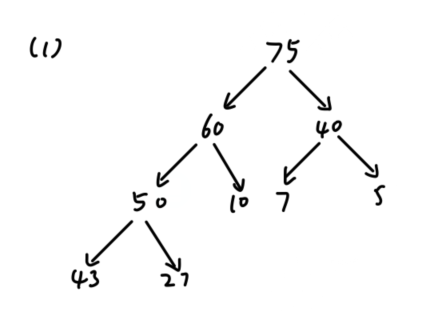
图 1.

**答：**

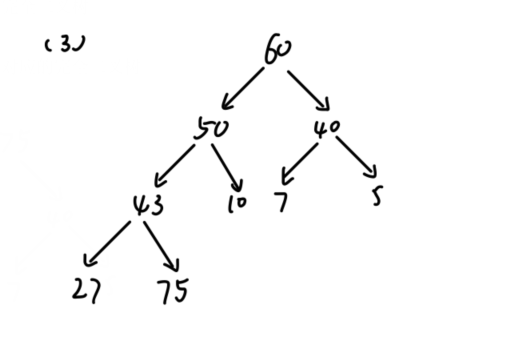
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** |
| **0** | **∞** | **∞** | **∞** | **∞** |
| **0** | **1** | **∞** | **3** | **10** |
| **0** | **1** | **6** | **3** | **10** |
| **0** | **1** | **5** | **3** | **9** |
| **0** | **1** | **5** | **3** | **6** |

1. 设使用堆排序法对关键字序列 T=(10, 27, 5, 50, 60, 7, 40, 43, 75）进行排序：（10 分）
   1. 画出初始大根堆对应的完全二叉树
   2. 写出大根堆序列
   3. 画出第一趟排序后新堆对应的完全二叉树

**答：**



**（2）（75，60，40，50，10，7，5，43，27）**



1. 简述下列算法的功能。（6分）

typedef struct BiTNode{

int data;

Struct BiTNode \*lchild;

Struct BiTNode \*rchild;

}BiTNode,\*BiTree;

int func(BiTree T)

{

if (T＝＝NULL) return(0);

else

if (T->data == 0)

return(1＋func(T->lchild)+ func(T->rchild));

else

return(func(T->lchild)+ func(T->rchild));

}

**答：该算法的功能是计算二叉树中结点的data=0的结点的个数。**

1. 使用Prime算法构造出图1所示的图G的一棵最小生成树（要求写出构造过程）。（10分）

16

v1

v2

6

11

21

19

6

33

14

v6

v3

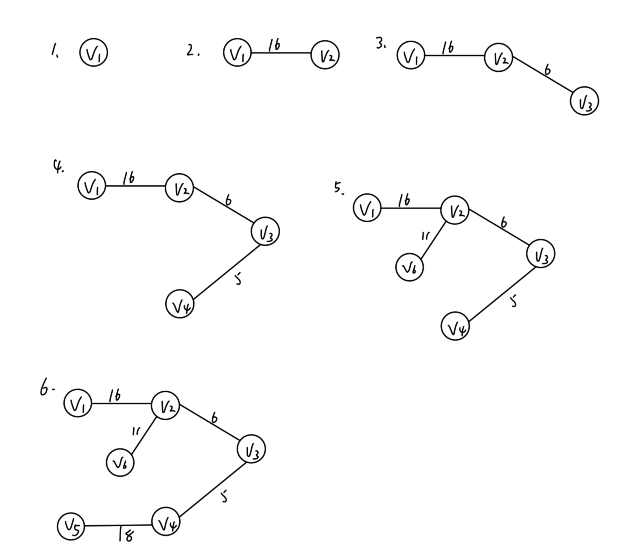
5

v5

v4

18

图1

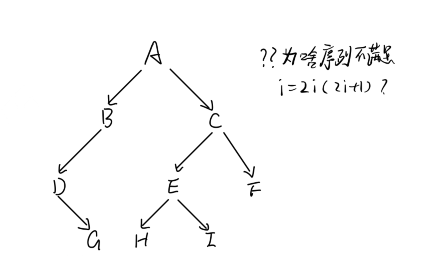


5. 假设二叉树采用顺序存储结构，如图2所示。 （6分）

* 1. 画出二叉树表示
  2. 写出先序遍历，中序遍历，后序遍历的结果

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D |  | E | F |  | G | H | I |

图 2

**答：（1）**

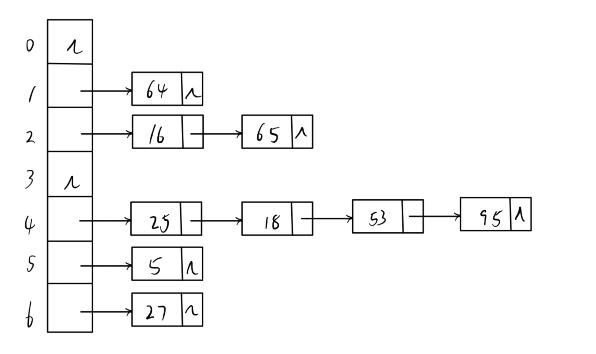
1. **先序：A,B,D,G,C,E,H,I,F**

**中序：D,G,B,A,H,E,I,C,F**

**后序：G,D,B,H,I,E,F,C,A**

6．设关键字序列为（64, 5,95, 53, 18, 25, 65, 27, 16），散列函数为H(key)=key%7，采用链地址法解决冲突，请回答：（8分）

1. 画出散列表示意图（用头插法向单链表中插入结点）
2. 查找关键字95时，需要依次与哪些关键字比较
3. 求等概率下查找成功的平均查找长度

**答：（1）**

**（2）需要依次和25，18，53比较**

**（3）（1+1+2+1+2+3+4+1+1）/9=16/9**

**五．算法填空，（每空2分,共18分）**

1. 设计一个函数功能为：在带头结点的单链表中删除值最小的元素。请将代码补充完整。

typedef int DataType;

typedef struct Node

{ DataType data;

struct Node \* next;

}LinkList;

void deleteMin(LinkList \*L)

{ LinkList \*p=L->next,\*q;

q=p;

while( **p!=NULL**  )

{ if( p->data < q->data)

q=p;

**p=p->next** ;

}

if(!q) return;

p=L;

while(p->next!=q)

p=p->next;

**p->next = p->next->next**  ;

**free(q)**  ;

}

1. 以下程序使用冒泡排序法对存放在a[1]，a[2]，……，a[n]中的序列进行排序，完成程序中的空格部分，其中n是元素个数，要求按升序排列。

typedef struct{

int key;

infotype otherinfo;

}Node;

void bsort (Node a[ ], int n)

{ NODE temp;

int i,j,flag;

for(j=1; **j<n** ;j++);

{flag=0;

for(i=1; **i<=n-j** ;i++)

if(a[i].key>a[i+1].key)

{flag=1;

temp=a[i];

**a[i] = a[i+1]** ;

**a[i+1]=a[i]** ;

}

if( **!flag** )break;

}

}

**六．编写算法（22分）**

1．设计在顺序有序表中实现折半查找的算法。（10分）

**int sort(int list[], int n, int key){**

**int l = 0, r = n-1;**

**while(l < r){**

**int mid = (l+r)>>1;**

**if(list[mid]==key) return mid;**

**else if(list[mid] > key) r = mid-1;**

**else l = mid+1;**

**}**

**return -1; //返回-1表示没有找到**

**}**

2．设计AOV-网拓扑排序的算法（12分）

**#define MAXN 500**

**void solve(int G[][MAXN], int n){**

**int \*sort = (int \*)malloc(n\*sizeof(int));**

**int \*degree = (int \*)malloc(n\*sizeof(int));**

**for(int i = 0; i < n; i++) degree[i] = 0;**

**for(int i = 0; i < n; i++){**

**for(int j = 0; j < n; j++){**

**if(G[i][j]) dree[j]++;**

**}**

**}**

**int \*queue = (int \*)malloc(n\*sizeof(int));**

**int head = 0, tail = 0;**

**int \*vis = (int \*)malloc(n\*sizeof(int));**

**for(int i = 0; i < n; i++) vis[i] = 0;**

**for(int i = 0; i < n; i++){**

**if(degree[i]==0){**

**sort[cnt++] = i;**

**queue[tail++] = i;**

**vis[i] = 1;**

**}**

**}**

**while(head!=tail){**

**int u = queue[head++];**

**for(int i = 0; i < n; i++){**

**if(!vis[i] && G[u][i]==1){**

**degree[i]-=1;**

**if(degree[i]==0) queue[tail++] = i;**

**}**

**}**

**}**

**if(cnt!=n) printf("该图不存在拓扑序列\n");**

**}**