

# 2000 年上海海运学院攻读硕士学位研究生入学考试试题

(本试题的答案必须做在答题纸上, 写在试题上不给分)

## 数 据 结 构

一、(本题15分, 每小题1.5分) 判断下列叙述的正确性, 将判断的结果写在答题纸上。

- 1 数据的逻辑结构说明数据元素之间的顺序关系, 它依赖于计算机的存储结构。
- 2 顺序存储方式只能用于存储线性结构。
- 3 为了很方便地插入和删除数据, 可以使用双向链表存放数据。
- 4 若输入序列为1, 2, 3, 4, 5, 6, 则通过一个栈可以输出序列3, 2, 5, 6, 4, 1。
- 5 散列表的节点只包含数据元素自身的数据, 不包含任何指针。
- 6 用链表 (llink--rlink) 存储包含 $n$ 个节点的二叉树时, 节点的 $2n$ 个指针区域中有 $n+1$ 个空指针。
- 7 每个节点的度数小于等于2的树是二叉树。
- 8 在查找树 (二叉排序树) 中插入一个新节点, 总是插入到叶节点下面。
- 9 邻接矩阵适用于有向图和无向图的存储, 但不能存储网络图 (带权的有向图和无向图), 而只能使用邻接表存储形式来存储它。
- 10 快速排序和归并排序在最坏情况下的比较次数都是  $O(n \log_2 n)$ 。

二、(本题15分, 每小题1.5分) 从供选择的答案中选出正确答案写在答题纸上

- 1 若某线性表中最常用的操作是在最后一个元素之后插入一个元素和删除第一个元素, 则采用\_\_\_\_\_存储方式最节省运算时间。

供选择的答案:

- A 单链表      B 仅有头指针的单循环链表  
C 双向链表    D 仅有尾指针的单循环链表

- 2 对于有 $n$ 个节点的二叉树, 其高度为\_\_\_\_\_。

供选择的答案:

- A  $n \log_2 n$     B  $\log_2 n$     C  $\lfloor \log_2 n \rfloor + 1$     D 不确定

- 3 \_\_\_\_\_, 它的先序遍历和中序遍历具有同样的顺序。

供选择的答案:

- A 二叉树    B 完全二叉树    C 无左子树的二叉树    D 无右子树的二叉树

- 4 将数据以\_\_\_\_\_结构存放, 查找一个数据的时间不依赖数据个数。

供选择的答案:

- A 顺序    B 链表    C 散列表    D 二叉排序树

- 5 可以用\_\_\_\_\_这种数据结构辅助实现图的广度优先遍历。

供选择的答案:

- A 栈    B 队列    C 广义表    D 树

- 6 有一散列表, 表的长度 $m$ 为100, 采用除余法构造散列函数, 即 $H(k) = k \text{ MOD } p$  ( $p \leq m$ ), 为使散列函数具有较好的性能,  $p$ 的选择应是\_\_\_\_\_。

供选择的答案:

- A 99    B 97    C 93    D 91

7 用某种排序方法对线性表( 96, 37, 45, 86, 99, 76, 43, 19, 67, 26, 65, 72, 85, 14 )进行排序时, 表内数据变化的情况依次如下:

- (1) 96 37 45 86 99 76 43 19 67 26 65 72 85 14
- (2) 65 37 19 14 26 76 43 45 67 99 96 72 85 86
- (3) 14 26 19 43 37 67 65 45 72 85 86 76 99 96
- (4) 14 19 26 37 43 45 65 67 72 76 85 86 96 99

则采用的是\_\_\_\_\_排序方法。

供选择的答案:

- A 选择 B 希尔 C 归并 D 快速

8 各种排序算法都需要一些辅助空间暂存数据, 插入排序、冒泡排序、选择排序和快速排序四种算法中, \_\_\_\_\_所需的辅助空间最多。

供选择的答案:

- A 插入排序 B 冒泡排序 C 选择排序 D 快速排序

9 对有14个元素的有序表A[1..14]作二分法查找, 查找元素A[4]时的被比较元素依次为\_\_\_\_\_。

供选择的答案:

- A A[1], A[2], A[3], A[4] B A[1], A[14], A[7], A[4]  
C A[7], A[3], A[5], A[4] D A[7], A[5], A[3], A[4]

10 顺序文件采用顺序结构实现文件的存储, 对大型顺序文件的少量修改要求重新复制整个文件, 代价很高。采用\_\_\_\_\_的方法则可降低所需的代价。

供选择的答案:

- A 附加文件 B 按关键字大小排序 C 按记录输入先后排序 D 连续存取

三、(本题10分) 有一个二维数组A[1:6, 0:7], 每个数组元素用相邻的6个字节存储, 存储器按字节编址, 假设存储数组元素A[1, 0]的第一个字节的地址是0, 那么存储数组的最后一个元素的第一个字节的地址是多少? 若按行存储, 则A[2, 4]和A[5, 3]的第一个字节的地址是多少? 若按列存储, 则A[3, 5]和A[5, 7]的第一个字节的地址是多少?

四、(本题16分, 第1小题6分, 第2小题10分)

1 已知稀疏矩阵

$$A = \begin{bmatrix} 0 & 5 & 0 & 6 \\ -7 & 0 & 0 & 10 \\ 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 0 \end{bmatrix}$$

请用二种存储形式描述它的压缩存储结构。

2 假设一个二叉树的两种遍历如下:

先序遍历 ABECDFGH

中序遍历 EBCDAFHG

请画出这棵二叉树以及它的二种存储方式。

```

while ( p->next !=NULL )
{ q= p->next;
  r=q;
  while ( q->next !=NULL )
  { if ( q->next->value<__(4)__ )
    r=q;
    q=q->next;
  }
  if ( r != p )
  { s=__(5)__;
    __(6)___=s->next;
    s->next=__(7)__;
    __(8)___=s;
  }
  p=p->next;
}
* h=h1->next;
free(h1);
}
int test-data[ ]={ 5,9,3,4,5,7,8 };
main( )
{ struct node * h, * p;
  h=create(test-data, sizeof test data / sizeof test-data[0]);
  for( p=h; p; p=p->next ) printf ("%5d", p->value);
  printf("\n");
  sort(&h);
  for( p=h; p; p=p->next ) printf ("%5d", p->value);
  printf("\n");
}

```

五、(本题10分) 已知图的邻接矩阵为

	V1	V2	V3	V4	V5	V6	V7	V8
V1	0	1	1	1	0	0	0	0
V2	0	0	1	0	0	1	0	0
V3	0	0	0	0	1	0	0	0
V4	0	0	0	0	0	0	1	0
V5	0	0	0	0	0	0	1	1
V6	0	0	0	0	1	0	0	1
V7	0	0	0	0	0	0	0	0
V8	0	0	0	0	0	0	0	0

试画出该有向图以及它的邻接表存储结构, 并写出其拓扑有序序列。

六、(本题18分, 第1小题6分, 第2小题12分) 已知长度为9的线性表, 其关键字输入序列为 (58, 34, 18, 26, 75, 67, 48, 93, 81)

- 1 试按关键字序列依次插入到一棵初始为空的二叉排序树, 画出插入完成后的二叉排序树, 并计算其在等概率情况下, 查找成功的平均查找长度。
- 2 设有哈希函数为  $H(\text{key}) = \text{key} \text{ MOD } 11$ , 哈希表HT的长度为11, 解决冲突的方法为线性探测开放定址法和链地址法, 试分别构造哈希表, 并计算其在等概率情况下, 查找成功的平均查找长度。

七、(本题16分) 阅读下列程序说明和C程序, 将应填入程序中\_\_(n)\_\_\_处的字句, 写在答案纸上。

[程序说明]

本程序给出两个函数。函数 create() 根据已知整数数组构造一个线性链表。函数 sort() 采用选择排序方法对已知链表进行排序。为排序方便, 函数 sort() 于排序前在链表首表元之前生成一个辅助表元。排序完成后, 将该辅助表元删去。

[程序]

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int value;
    struct node * next;
};
struct node * create ( int a[ ], int n )
{ struct node * h, * q;
  for ( h=NULL; n; n-- )
  { q=( struct node * )malloc(sizeof(struct node));
    q->value=__(1)__;
    __(2)__;
    __(3)__;
  }
  return h;
}
void sort (struct node ** h)
{ struct node * p, * q, * r, * s, * h1;
  h1=p=( struct node * ) malloc(sizeof(struct node));
  p->next= * h;
```