

湖南科技学院二〇一九年下学期期末考试

计科、软件专业 2018 年级 数据结构 试题

考试类型：闭卷

试卷类型：A 卷

考试时量：120 分钟

题号	一	二	三	四	五				总分	统分人
得分										
阅卷人										
复查人										

一、选择题（每小题 2 分，20 小题共 40 分）

1、与数据元素本身的形式、内容、相对位置、个数无关的是数据的（ ）。

- A. 存储结构 B. 存储实现
C. 逻辑结构 D. 运算实现

2、试分析下面程序段的时间复杂度（ ）。

```
x=90; y=100;
while(y>0)
    if(x>100)
        {x=x-10;y--;}
    else x++;
```

- A. $O(1)$ B. $O(n^2)$ C. $O(\log_3 n)$ D. $O(\sqrt{n})$

3、顺序表中第一个元素的存储地址是 100，每个元素的长度为 2，则第 5 个元素的地址是（ ）。

- A. 110 B. 108 C. 100 D. 120

4、链接存储的存储结构所占存储空间（ ）。

- A. 分两部分，一部分存放结点值，另一部分存放表示结点间关系的指针
B. 只有一部分，存放结点值
C. 只有一部分，存储表示结点间关系的指针
D. 分两部分，一部分存放结点值，另一部分存放结点所占单元数

5、创建一个包括 n 个结点的有序单链表的时间复杂度是（ ）。

- A. $O(1)$ B. $O(n)$ C. $O(n^2)$ D. $O(n\log_2 n)$

6、在双向链表存储结构中，删除 p 所指的结点时须修改指针（ ）。

- A. $p \rightarrow next \rightarrow prior = p \rightarrow prior$; $p \rightarrow prior \rightarrow next = p \rightarrow next$;
B. $p \rightarrow next = p \rightarrow next \rightarrow next$; $p \rightarrow next \rightarrow prior = p$;
C. $p \rightarrow prior \rightarrow next = p$; $p \rightarrow prior = p \rightarrow prior \rightarrow prior$;
D. $p \rightarrow prior = p \rightarrow next \rightarrow next$; $p \rightarrow next = p \rightarrow prior \rightarrow prior$;

7、若已知一个栈的入栈序列是 1, 2, 3, ..., n ，其输出序列为 $p_1, p_2, p_3, \dots, p_n$ ，若 $p_1 = n$ ，则 p_i 为（ ）。

- A. i B. $n-i$ C. $n-i+1$ D. 不确定

8、设栈 S 和队列 Q 的初始状态为空，元素 e_1, e_2, e_3, e_4, e_5 和 e_6 依次进入栈 S ，一个

元素出栈后即进入 Q, 若 6 个元素出队的序列是 e2、e4、e3、e6、e5 和 e1, 则栈 S 的容量至少应该是 ()。

- A. 2 B. 3 C. 4 D. 6

9、若一个栈以向量 $V[1..n]$ 存储, 初始栈顶指针 top 设为 $n+1$, 则元素 x 进栈的正确操作是 ()。

- A. $top++$; $V[top]=x$; B. $V[top]=x$; $top++$;
C. $top--$; $V[top]=x$; D. $V[top]=x$; $top--$;

10、最大容量为 n 的循环队列, 队尾指针是 $rear$, 队头是 $front$, 则队空的条件是 ()。

- A. $(rear+1)\%n==front$ B. $rear==front$
C. $rear+1==front$ D. $(rear-1)\%n==front$

11、假设以行序为主序存储二维数组 $A=array[1..100,1..100]$, 设每个数据元素占 2 个存储单元, 基址为 10, 则 $LOC[5,5]=$ ()。

- A. 808 B. 818 C. 1010 D. 1020

12、若对 n 阶对称矩阵 A 以行序为主序方式将其下三角形的元素(包括主对角线上所有元素)依次存放于一维数组 $B[1..(n(n+1))/2]$ 中, 则在 B 中确定 a_{ij} ($i < j$) 的位置 k 的关系为 ()。

- A. $i*(i-1)/2+j$ B. $j*(j-1)/2+i$ C. $i*(i+1)/2+j$ D. $j*(j+1)/2+i$

13、一个具有 1025 个结点的二叉树的高 h 为 ()。

- A. 11 B. 10 C. 11 至 1025 之间 D. 10 至 1024 之间

14、深度为 h 的满 m 叉树的第 k 层有 () 个结点。 ($1 \leq k \leq h$)

- A. m^{k-1} B. m^k-1 C. m^{h-1} D. m^h-1

15、设哈夫曼树中有 50 个叶结点, 则该哈夫曼树中有 () 个结点。

- A. 99 B. 100
C. 101 D. 102

16、下面 () 算法适合构造一个稀疏图 G 的最小生成树。

- A. Prim 算法 B. Kruskal 算法 C. Floyd 算法 D. Dijkstra 算法

17、用邻接表表示图进行广度优先遍历时, 通常借助 () 来实现算法。

- A. 栈 B. 队列 C. 树 D. 图

18、折半查找有序表 (4, 6, 10, 12, 20, 30, 50, 70, 88, 100)。若查找表中元素 58, 则它将依次与表中 () 比较大小, 查找结果是失败。

- A. 20, 70, 30, 50 B. 30, 88, 70, 50
C. 20, 50 D. 30, 88, 50

19、下列关键字序列中, () 是堆。

- A. 16, 72, 31, 23, 94, 53 B. 94, 23, 31, 72, 16, 53
C. 16, 53, 23, 94, 31, 72 D. 16, 23, 53, 31, 94, 72

20、下述几种排序方法中, () 是稳定的排序方法。

- A. 希尔排序 B. 快速排序 C. 归并排序 D. 堆排序

二、程序阅读题 (每题 4 分, 共 8 分)

1、int fun(int n)

```
{  
    if (n==1 || n==2)  
        return n;  
    else  
        return fun(n-1)+fun(n-2);  
}
```

```
}
```

执行 fun(6)的结果是_____

2、

```
int fun(int a[],int n,int k)
{
    int i;
    for (i=0;i<n;i+=2)
        if (a[i]==k)
            return i;
    for (i=1;i<n;i+=2)
        if (a[i]==k)
            return i;
    return -1;
}
```

当 a[]={2, 6, 3, 8, 1, 7, 4, 9}时, 执行 fun(a, 8, 5)的结果是_____

三、程序填空题（每空 2 分，共 8 分）

已知单链表的存储结构为

```
typedef struct LNode          //定义单链表结点类型
{
    ElemType data;            //数据域
    struct LNode *next;       //指向后继结点
} LinkNode
```

将以下程序补充完整，实现用尾插法建立单链表。

```
void CreateListR(LinkNode *&L, ElemType a[], int n)
{
    LinkNode *s, *r;    //r 为尾指针
    int i;
    L=(LinkNode *)malloc(sizeof(LinkNode));
    _____ (1) _____
    for (i=0;i<n;i++)
    {
        s=(LinkNode *)malloc(sizeof(LinkNode));
        s->data=a[i];
        _____ (2) _____
        _____ (3) _____
    }
    _____ (4) _____
}
```

四、综合分析题（共 32 分）

1、已知一棵二叉树的中序序列为 *cbedahgijf*，后序序列为 *cedbhjigfa*，给出该二叉树树形表示。（7分）

2、设一组关键字为（7，15，20，31，48，53，64，76，82，99），Hash 函数 $H(\text{key}) = \text{key} \% 11$ ，Hash 表表长 $m=11$ ，用线性探测法解决冲突，试构造 Hash 表，并计算查找成功和不成功情况下的平均查找长度。（9分）

3、已知一个图的顶点集 V 和边集 E 分别为：

$V=\{1,2,3,4,5,6,7\}$;

$E=\{(1,2)3,(1,3)5,(1,4)8,(2,5)10,(2,3)6,(3,4)15,(3,5)12,(3,6)9,(4,6)4,(4,7)20,(5,6)18,(6,7)25\}$;

说明：边集合中（1,2）3 中 1,2 代表顶点，3 为边上的权值

画出该图并用克鲁斯卡尔算法得到最小生成树，试写出最小生成树的边的生成顺序过程。（9分）

4、将整数序列（4，5，7，2，1，3，6）中的元素依次插入到一棵空的二叉排序树中，试构造相应的二叉排序树，要求用图形给出构造过程。（7分）

五、算法设计题（12分）

已知图的邻接表存储如下所示，

```
typedef struct ANode
```

```
{    int adjvex;           //该边的终点编号

    struct ANode *nextarc; //指向下一条边的指针

    InfoType info;        //该边的权值等信息
} ArcNode;
```

```
typedef struct Vnode
```

```
{    Vertex data;          //顶点信息

    ArcNode *firstarc;     //指向第一条边
} VNode;
```

```
typedef struct
```

```
{    VNode adjlist[MAXV]; //邻接表

    int n, e;            //图中顶点数 n 和边数 e
} AdjGraph;
```

假设图 G 采用邻接表存储，设计一个算法，输出图 G 中从顶点 u 到顶点 v 的所有简单路径，写出算法代码。

湖南科技学院二〇一九年下学期期末考试

计科、软件专业 2018 年级 数据结构 参考答案

一、选择题（每小题 2 分，20 小题共 40 分）

1	2	3	4	5	6	7	8	9	10
C	A	B	A	C	A	C	B	C	B
11	12	13	14	15	16	17	18	19	20
B	B	C	A	C	B	B	A	D	C

二、程序阅读题（每题 4 分，共 8 分）

1、 13

2、 -1

三、程序填空题（每空 2 分，共 8 分）

1、 `r=L;`

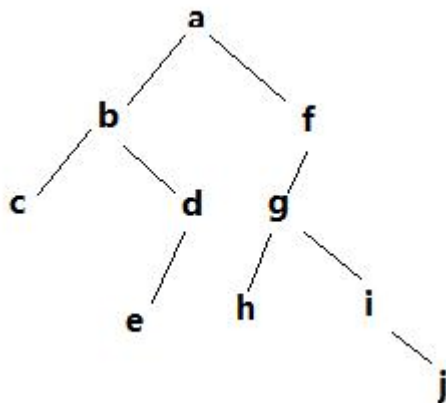
2、 `r->next=s;`

3、 `r=s;`

4、 `r->next=NULL;`

四、综合分析题（共 32 分）

1、（7 分）（错一个点扣一分，扣至 0 分为止）



2、（9 分）

答：(1)用线性探测法解决冲突的 Hash 用表格表示如下：

（3 分）

H(key)	0	1	2	3	4	5	6	7	8	9	10
key	53	64	76	99	15	48	82	7		20	31

(2) 关键字比较次数：

（3 分）

key	53	64	76	99	15	48	82	7	20	31
比较次数	3	4	4	4	1	2	2	1	1	2

(3) 查找成功的平均查找长度为： $(3+4*3+1*3+2*3)/10=2.4$

（1.5 分）

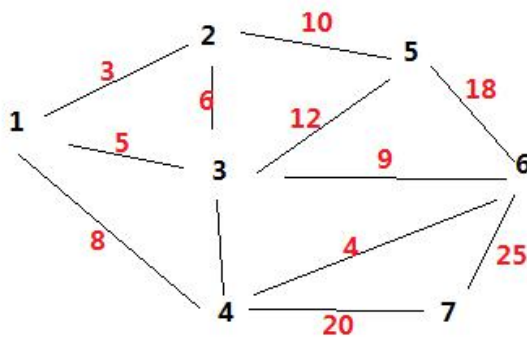
不成功情况下的平均查找长度为： $(9+8+7+6+5+4+3+2+1+11+10)/11=6$

（1.5 分）

3、（9 分）

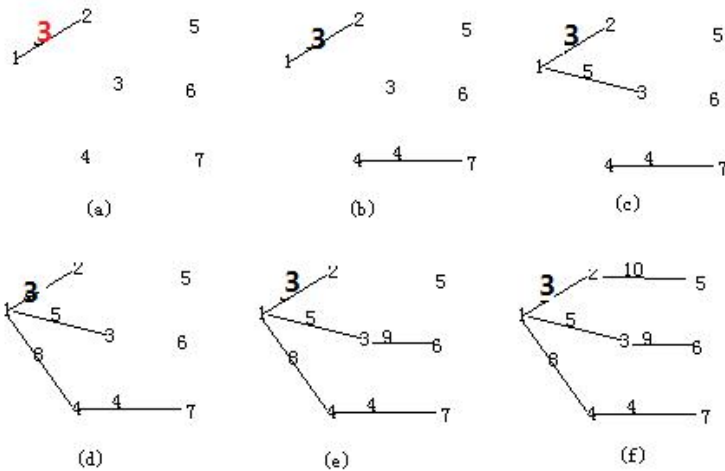
答：（1）该图为：

（3 分）



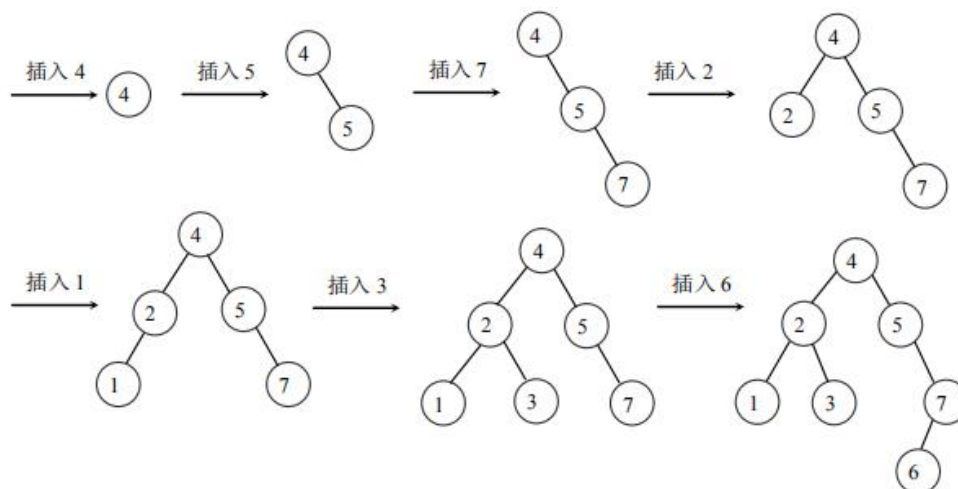
(2) 克鲁斯卡尔算法过程：（画图或写边集均可，错一条边扣 1 分）

（6 分）



或 (1, 2), (4, 7), (1, 3), (1, 4), (3, 6), (2, 5)

4、（7分）（一步一分）



五、算法设计题（12分）

void FindAllPath(AGraph *G, int u, int v, int path[], int d)

{ //d 表示 path 中的路径长度，初始为-1

int w,i; ArcNode *p;

d++; path[d]=u; //路径长度 d 增 1, 顶点 u 加入到路径中 (1分)

visited[u]=1; //置已访问标记 (1分)

if (u==v && d>=1) //找到一条路径则输出 (4分)

```

{
    for (i=0;i<=d;i++)
        printf("%2d",path[i]);
    printf("\n");
}

```

p=G->adjlist[u].firstarc; //p 指向顶点 u 的第一个相邻点 (1分)

while (p!=NULL) (4分)

```

{
    w=p->adjvex; //w 为顶点 u 的相邻顶点
    if (visited[w]==0) //若 w 顶点未访问,递归访问它
        FindAllPath(G, w, v, path, d);
    p=p->nextarc; //p 指向顶点 u 的下一个相邻点
}

```

visited[u]=0; (1分)

}

注：如有不同的方法，可根据实际情况酌情给分。