装 订 线

任课教师：

装 订 线

学号：

姓名：

装 订 线

班级：

西 安 电 子 科 技 大 学

考试时间 **120** 分钟

试　　题

题号	一	二	三	四	总分
分数					

1.考试形式：闭卷；　2.本试卷共四大题，满分 100 分；

3.考试日期：　　年　　月　　日；(答题内容请写在装订线外)

一、 单项选择题(本大题共 15 小题，每小题 2 分，共 30 分)

1. 下面关于数据结构的说法，正确的是（ **C** ）。

(A) 逻辑上相邻的元素在存储位置上一定相邻

(B) 算法的实现依赖于采用的逻辑结构

(C) 数据的逻辑结构分为线性结构和非线性结构

(D) 以上说法都正确

2. 算法分析的两个主要方面是(**A**)：

(A) 空间复杂度和时间复杂度　　**(B)** 正确性和简明性

(C) 可读性和文档性　　**(D)** 数据复杂性和程序复杂性

3. 具有线性结构的数据结构是(**C**)。

(A) 广义表　**(B)** 数组　**(C)** 双端队列　**(D)** 二叉树

4. 将两个各有 **n** 个元素的有序表归并成一个有序表，其最少的比较次数是（ **A** ）。

(A) n　**(B) 2n-1**　**(C) 2n**　**(D) n-1**

5. 在具有 **n** 个结点的单链表中，实现（ **A** ）的操作，其算法的时间复杂度是 **O(n)**。

(A) 遍历链表或求链表的第 **i** 个结点　**(B)**删除地址为 **P** 的结点的后继结点

(C) 在地址为 **P** 的结点之后插入一个结点　**(D)** 删除首元结点

6. 若一个栈的输入序列是 **1,2,3, …, n**，输出序列的第一个元素是 **n**，则第 **i** 个输出元素是（ **B** ）。

(A) n-i　**(B) n-i+1**　**(C) i**　**(D) n-i-1**

7. 约定满队时队首指针与队尾指针相差 **1**，判定一个循环队列 **Q**（队列空间大小为 **n**）为满队列的条件是(**A**) 。

(A) Q.front == (Q.rear+1)% n　　(B) Q.rear - Q.front – 1 == n

(C) Q.front == Q.rear　　(D) Q.front == Q.rear+1

8. 设 A 为一个 10 阶的对称矩阵，采用压缩存储方式，以行序为主序，a_{11} 为第一元素，其存储地址为 1，每个元素占一个地址空间，则 a_{85} 的地址为（ B ）。

(A) 13　　(B) 33　　(C) 18　　(D) 40

9. 若树中用一个分支把两个结点连接起来，则 (B)。

(A) 不一定出现环　　(B) 一定出现环

(C) 使树的度数增一　　(D) 前面说法都不正确

10. 下面几个符号串编码集合中，不是前缀编码的是（ B ）。

(A) { 0, 10, 110, 1111 }　　(B) { 11, 10, 001, 101, 0001 }

(C) { 00, 010, 0110, 1000}　　(D) { b, c, aa, ac, aba, abb, abc }

11. 把一棵树转换为二叉树后，这棵二叉树的形态是（ A ）。

(A) 唯一的　　(B) 有多种，但根结点都没有左孩子

(C) 有多种　　(D) 有多种，但根结点都没有右孩子

12. 深度优先遍历类似于二叉树的（ A ）。

(A) 先序遍历　　(B) 中序遍历　(C) 后序遍历　　(D) 层次遍历

13. 如果要求一个线性表既能较快地查找，又能适应动态变化的要求，则可采用（ A ）查找方法。

(A) 分块查找　　(B) 顺序查找

(C) 折半查找　　(D) 基于属性的查找

14. 哈希函数有一个共同的性质，即函数值应当以(D)取其值域的每个值。

(A) 最大概率　(B) 最小概率　(C) 平均概率　(D) 同等概率

15. 下列 (D)排序算法在最坏情况下不会达到 $O(n^2)$。

(A) 插入排序　　(B) 冒泡排序

(C) 快速排序　　(D) 2-路归并排序

二、 填空题(本大题共 13 小题，每空 1 分，共 20 分)

1. 数据结构一般包括三个方面的内容，也就是它的三要素，它们分别是：

数据的逻辑结构，数据的存储结构，数据的运算。

2. 在一个具有 n 个结点的有序单链表中插入一个新结点并仍然有序的时间复杂度为 O(n)。

3. 在非空双向循环链表中（结点有 prior 和 next 指针），在指针 q 所指的结点前插入一个由指针 p 所指结点的过程依次为：p->next=q; p->prior=q->prior;

q->prior=p; p->prior->next=p 。

4. 循环队列定义为：rear 指示队尾元素的位置，length 指示元素的个数，队列已分配空间的大小为 Maxlen。循环队列的队满条件为 length==Maxlen ，队头元素的位置为 front = (rear-length+1+Maxlen)%Maxlen 。

5. 模式串 p=“abaabcac”的 next 函数值序列为 01122312 。

6. 广义表 A=(a, (a,b), d, e, ((i,j), k))，其长度为 5 ，深度为 3 ，Head(A)的结果为 a ，Head(Tail(A)) 的结果为 (a, b) 。

7. 一棵有 n 个结点的树，所有结点的度之和为 n-1 。

8. 用一维数组存放的一棵完全二叉树：ABCDEFGHIJKL。请写出该二叉树的后序遍历序列 HIDJKEBLFGCA 。

9. 设 F 是由 T_1, T_2, T_3 三棵树组成的森林，与 F 对应的二叉树为 B（按 T_1, T_2, T_3 依次合并而成）。若 T_1, T_2, T_3 的结点数分别为 N_1, N_2, N_3，则二叉树 B 的左子树中有 N_1-1 个结点，二叉树 B 的右子树中有 N_2+N_3 个结点。

10. 设 n_0 为赫夫曼树的叶子结点数，则该赫夫曼树共有 $2n_0-1$ 个结点。

11. 设有向图有 n 个顶点和 e 条边，采用邻接表进行存储，对其进行拓扑排序，其时间复杂度为 O(n+e) 。

12. 一个无序序列可以通过构造一棵 二叉排序 树而变成一个有序序列，构造树的过程即为对无序序列进行排序的过程。

13. 基数 排序方法不需要进行记录关键字间的比较。

三 应用题(本大题共 5 小题，共 32 分)

1.（6 分） 分析下面三小段程序的时间复杂度。

a.
```
for (i=0;  i<n; i++)
    for (j=0; j<m; j++)
        A[i][j]=0;
```
答：O（m*n）--2 分

b.
```
s=0;
for i=0; i<n; i++)
    for(j=0; j<n; j++)
        s+=B[i][j];
    sum=s;
```
答：O（n^2）--2 分

c.
```
x=0;
for(i=1; i<n; i++)
  for (j=1; j<=n-i; j++)
    x++;
```
解：因为 x++共执行了 n-1+n-2+……＋1= n(n-1)/2，所以执行时间为 **O（n^2）** **--2 分**

2.（6 分）设有编号为 1，2，3，4 的四辆列车，顺序进入一个栈式结构的车站，具体写出这四辆列车开出车站的所有可能的顺序。

答：至少有 14 种。 （酌情给分）

① 全进之后再出情况，只有 1 种：4，3，2，1

② 进 3 个之后再出的情况，有 3 种，3,4,2,1　3,2,4,1　3,2,1,4

③ 进 2 个之后再出的情况，有 5 种，2,4,3,1　2,3,4,1　2,1, 3,4　2,1,4,3　2,1,3,4

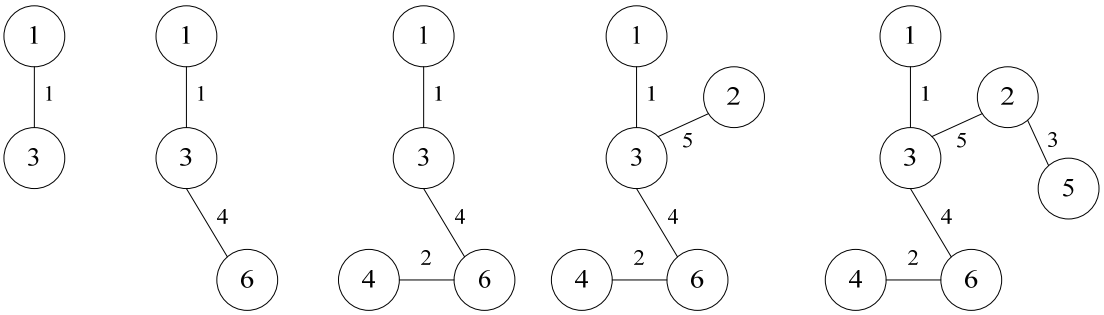
④ 进 1 个之后再出的情况，有 5 种，1,4,3,2　1,3,2,4　1,3,4,2　1, 2,3,4　1,2,4,3

3、（6 分）已知图 G 的邻接矩阵如下所示：（1）求从顶点 1 出发的广度优先搜索序列；（2）根据 prim 算法，求图 G 从顶点 1 出发的最小生成树，要求表示出其每一步生成过程。（用图或者表的方式均可）。

$$\begin{Bmatrix} \infty & 6 & 1 & 5 & \infty & \infty \\ 6 & \infty & 5 & \infty & 3 & \infty \\ 1 & 5 & \infty & 5 & 6 & 4 \\ 5 & \infty & 5 & \infty & \infty & 2 \\ \infty & 3 & 6 & \infty & \infty & 6 \\ \infty & \infty & 4 & 2 & 6 & \infty \end{Bmatrix}$$

答案：(1)广度优先遍历序列：1; 2, 3, 4; 5; 6 --3 分

(2)最小生成树（prim 算法） --3 分



4、（6 分）设哈希表 HT 表长 m 为 13，哈希函数为 H(k)=k MOD m，给定的关键值序列为{19,14,23,10,68,20,84,27,55,11}。试求出用线性探测法解决冲突时所构造的哈希表，并求出在等概率的情况下查找成功的平均查找长度 ASL。

答案：(1)表形态： --3 分

0	1	2	3	4	5	6	7	8	9	10	11	12
	14	27	68	55		19	20	84		23	10	11
	1	2	1	2		1	1	3		1	2	2

(2)平均查找长度：ASL(10)=(1*5+2*4+3*1)/10=1.6 --3 分

5．(8 分)某一序列为{49，38，20，90，51，82，75，10，66}。请回答以下四小问题：（1）以第一元素为枢轴，写出快速排序第一趟结果；（2）写出进行 2-路归并排序的过程；（3）判断该序列是否为堆。若不是，将该序列调整成小顶堆；（4）对 n 个元素进行排序，写出上述三种排序方法所需辅助空间和在最坏情况下的时间复杂度。

答：（1）{10， 38， 20， 49， 51，82，75，90，66} (---2 分)

（2） (---2 分)

49	38	20	90	51	82	75	10	66
38	49	20	90	51	82	10	75	66
20	38	49	90	10	51	75	82	66
10	20	38	49	51	75	82	90	66
10	20	38	49	51	66	75	82	90

（3）不是堆。小顶堆：{10，38，20，49，51，82，75，90，66} (---2 分)

（4）辅助空间：堆 O(1)，快速 O(logn)， 归并 O(n).

时间复杂度：堆 O(nlogn)，快速 $O(n^2)$，归并 O(nlogn). (---2 分)

三 算法题(本大题共 2 小题，共 18 分)

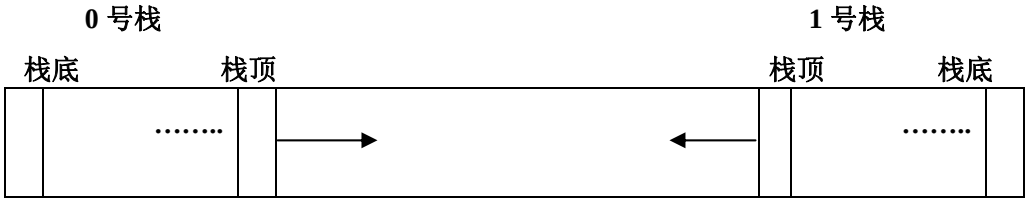
1.（8 分）设计算法将一个带头结点的单链表 A 分解为两个具有相同结构的链表 B、C，其中 B 表的结点为 A 表中值小于零的结点，而 C 表的结点为 A 表中值大于零的结点（链表 A 的元素类型为整型，要求 B、C 表利用 A 表的结点）。

```
void decompose(Linklist La,Linklist &Lb,Linklist &Lc)
{
Linknode *p;
Lc=(Linknode *)malloc(sizeof(Linknode));
Lc->next=NULL;
p=La->next;
Lb=La;
Lb->next=NULL;
while(p)
{
    La=p->next;
```

```
        if(p->data>0)
        {
            p->next=Lc->next;
            Lc->next=p;
        }
        else
        {
             p->next=Lb->next;
            Lb->next=p;
        }
        p=La;
    }
}
```

2．(10 分)一个双向栈 S 是在同一向量空间内实现的两个栈(顺序栈)，它们的栈底分别设在向量空间的两端。如下图所示。 试为此双向栈设计初始化 InitStack (S) 、入栈 Push(S , i , x) 和出栈 Pop(S , i)等三个子操作算法， 其中 i 为 0 或 1， 用以表示栈号。

双向栈 S




```
//双向栈数据类型定义
#define STACK_SIZE 100; Typedef struct
{
    SElemType  * base_one, * base_two;//栈底指针
    ElemType  * top_one, * top_two;//栈顶指针          int stacksize;
} SqStack;


Status InitStack ( SqStack &S)
{ //初始化双向栈
    //第一个栈底和栈顶指向数组起点
```

```
    S.base_one=S.top_one=( SElemType *)malloc(STACK_SIZE * sizeof(SElemType));
    // 第二个栈底和栈顶指向数组终点
    S.base_two=S.top_two=S.base_one +STACK_SIZE-1;
    S.stacksize= STACK_SIZE;
     return OK;
}//InitStack
Status Push ( SqStack &S, int i, SElemType e)
{   //入栈操作，i=0 时将 e 存入前栈，i=1 时将 e 存入后栈
    if( S.top_two < S.top_one)   return OVERFLOW;//栈满，不考虑追加空间
     if( i = = 0 )
        *S.top_one++ = e;
      else
        *S.top_two-- = e;
     return OK;
}//Push

SElemType  Pop ( SqStack &S, int i)
{   //出栈操作，i=0 出前栈，i=1 出后栈
    if( i==0 )
    {
       if ( S.top_one==S.base_one) return ERROR;
       S.top_one--; e=*S.top_one;
     }
     else
     {
          if ( S.top_two==S.base_two) return ERROR;
           S.top_two++; e=*S.top_two;
    }
    return e;
  }//Pop
```