

## 数据结构得基本概念

### 选择题

- (1) 顺序存储结构中数据元素之间得逻辑关系就是由（\_\_\_\_）表示得，链接存储结  
构中得数据元素之间得逻辑关系就是由（\_\_\_\_）表示得。  
A. 线性结构    B. 非线性结构    C. 存储位置    D. 指针
- (2) 假设有如下遗产继承规则：丈夫与妻子可以相互继承遗产，子女可以继承父亲  
或母亲得遗产；子女间不能相互继承，则表示该遗产继承关系得最合适得数据结构  
应该就是（\_\_\_\_）。  
A. 树    B. 图    C. 线性表    D. 集合
- (3) 计算机所处理得数据一般具有某种内在联系，这就是指（\_\_\_\_）。  
A. 数据与数据之间存在某种关系    B. 元素与元素之间存在某种关系  
C. 元素内部具有某种结构    D. 数据项与数据项之间存在某种关系
- (4) 在数据结构中，与所使用得计算机无关得就是数据得（\_\_\_\_）。  
A. 树    B. 图    C. 线性表    D. 集合
- (5) 在存储数据时，通常不仅要存储各数据元素得值，还要存储（\_\_\_\_）。  
A. 数据得处理方法    B. 数据元素得类型  
C. 数据元素之间得关系    D. 数据得存储方法
- (6) 在链接存储结构中，要求（\_\_\_\_）。  
A. 每个结点占用一片连续得存储区域    B. 所有结点占用一片连续得存储区域  
C. 结点得最后一个域就是指针类型    D. 每个结点有多少个后继就设多少个指针
- (7) 下列说法不正确得就是（\_\_\_\_）。  
A. 数据元素就是数据得基本单位    B. 数据项就是数据中不可分割得最小单  
位  
C. 数据可由若干个数据项构成    D. 数据元素可由若干个数据项构成
- (8) 以下与数据得存储结构无关得术语就是（\_\_\_\_）。  
A. 循环队列    B. 链表    C. 散列表    D. 栈
- (9) 以下术语属于逻辑结构得就是（\_\_\_\_）。  
A. 顺序表    B. 哈希表    C. 有序表    D. 单链表
- (10) 可以用（\_\_\_\_）定义一个完整得数据结构。  
A. 数据元素    B. 数据对象    C. 数据关系    D. 抽象数据类型
- (11) 对于数据结构得描述，下列说法中不正确得就是（\_\_\_\_）。  
A. 相同得逻辑结构对应得存储结构也必相同  
B. 数据结构由逻辑结构、存储结构与基本操作三方面组成  
C. 数据结构基本操作得实现与存储结构有关  
D. 数据得存储结构就是数据得逻辑结构得机内实现
- (12) 以下关于链接存储结构得叙述中，（\_\_\_\_）就是不正确得。  
A. 结点除数据信息外还包括指针域，因此存储密度小于顺序存储结构  
B. 逻辑上相邻得结点物理上不一定相邻  
C. 可以通过计算得到第 i 个结点得存储地址  
D. 插入与删除操作方便，不必移动结点
- (13) 可以用（\_\_\_\_）、数据关系与基本操作定义一个完整得抽象数据类型。  
A. 数据元素    B. 数据对象    C. 原子类型    D. 存储结构

### 应用题

- (14) 设有数据结构  $(D, R)$ , 其中  $D=\{1, 2, 3, 4, 5, 6\}$ ,  $R=\{(1, 2), (2, 3), (2, 4), (3, 4), (3, 5), (3, 6), (4, 5), (4, 6)\}$ 。试画出其逻辑结构图并指出属于何种结构。
- (15) 试描述数据结构与抽象数据类型得概念与程序设计语言中数据类型概念得区别。
- (16) 说明数据得逻辑结构与存储结构之间得关系。
- (17) 抽象数据类型得主要特点就是什么? 数据类型与抽象数据类型得关系如何?  
使用抽象数据类型得主要好处就是什么?

# 1 算法与算法分析

## 选择题

- (1) 算法指得就是 (\_\_\_\_)。  
A. 对特定问题求解步骤得一种描述，就是指令得有限序列  
B. 计算机程序  
C. 解决问题得计算方法  
D. 数据处理
- (2) 下面 (\_\_\_\_) 不就是算法所必须具备得特性。  
A. 有穷性      B. 确切性      C. 高效性      D. 可行性
- (3) 算法必须具备输入、输出与 (\_\_\_\_) 等特性。  
A. 可行性、可移植性与可扩充性      B. 可行性、确定性与有穷性  
C. 确定性、稳定性与有穷性      D. 易读性、稳定性与健壮性
- (4) 算法应该具有确定性、可行性与有穷性，其中有穷性就是指 (\_\_\_\_)。  
A. 算法在有穷得时间内终止      B. 输入就是有穷得  
C. 输出就是有穷得      D. 描述步骤就是有穷得
- (5) 当输入非法错误时，一个“好”得算法会进行适当处理，而不会产生难以理解得输出结果，这称为算法得 (\_\_\_\_)。  
A. 可读性      B. 健壮性      C. 正确性      D. 有穷性
- (6) 算法分析得目得就是 (\_\_\_\_)，算法分析得两个主要方面就是 (\_\_\_\_)。  
A. 找出数据结构得合理性      B. 研究算法中输入与输出得关系  
C. 分析算法得效率以求改进      D. 分析算法得易读性与文档性  
E. 空间性能与时间性能      F. 正确性与简明性  
G. 可读性与文档性      H. 数据复杂性与程序复杂性
- (7) 算法得时间复杂度与 (\_\_\_\_) 有关。  
A. 问题规模      B. 计算机硬件性能  
C. 编译程序得质量      D. 程序设计语言
- (8) 算法得时间复杂度与 (\_\_\_\_) 有关。  
A. 问题规模      B. 待处理数据得初态  
C. 算法得易读性      D. A 与 B
- (9) 某算法得时间复杂度就是  $\mathcal{O}(n^2)$ ，表明该算法 (\_\_\_\_)。  
A. 问题规模就是  $n^2$       B. 执行时间等于  $n^2$   
C. 执行时间与  $n^2$  成正比      D. 问题规模与  $n^2$  成正比
- (10) 下面说法错误得就是 (\_\_\_\_)。  
①算法原地工作得含义就是指示不需要如何额外得辅助空间  
②在相同得规模  $n$  下，复杂度  $\mathcal{O}(n)$  得算法在时间上总就是优于复杂度  $\mathcal{O}(2^n)$  得算法  
③所谓时间复杂度就是指最坏情况下，估算算法执行时间得一个上界  
④同一个算法，实现语言得级别越高，执行效率就越低
- (11) 算法
- ```
for (i=n-1; i>=1; i--)  
    for (j=1; j<=i; j++)  
        if (a[j]>a[j+1]) a[j] 与 a[j+1] 交换;
```
- 其中  $n$  为正整数，则最后一行语句得频度（执行次数）在最坏情况下就是 (\_\_\_\_)。  
A.  $\mathcal{O}(n)$       B.  $\mathcal{O}(n \log_2 n)$       C.  $\mathcal{O}(n^3)$       D.  $\mathcal{O}(n^2)$



- ① 在递归计算  $F_n$  的时候，需要对较小得  $F_{n-1}$ ,  $F_{n-2}$ , ...,  $F_1$ ,  $F_0$  精确计算多少次？
  - ② 用大O表示法给出递归计算时递归函数得时间复杂度就是多少？
- (21) 运算就是数据结构得一个重要方面。举例说明两个数据结构得逻辑结构与存储方式完全相同，只就是对于运算得定义不同，因而具有不同得特性，则这两个数据结构就是不同得。
- (22) 针对给定得实际问题建立数据结构时，应从哪些方面考虑。

## 2 线性表得逻辑结构

### 选择题

- (1) 线性表就是具有 n 个（\_\_\_\_）得有限序列。  
A. 数据      B. 字符      C. 数据元素      D. 数据项
- (2) 线性表就是（\_\_\_\_）。  
A. 一个有限序列，可以为空      B. 一个有限序列，不能为空  
C. 一个无限序列，可以为空      D. 一个无限序列，不能为空
- (3) 关于线性表，下列说法中正确得就是（\_\_\_\_）。  
A. 线性表中每个元素都有一个直接前驱与一个直接后继  
B. 线性表中得数据元素可以具有不同得数据类型  
C. 线性表中数据元素得类型就是确定得  
D. 线性表中任意一对相邻得数据元素之间存在序偶关系
- (4) （\_\_\_\_）就是一个线性表。  
A. 由 n 个实数组成得集合      B. 由所有实数组成得集合  
C. 由所有整数组成得序列      D. 由 n 个字符组成得序列

### 3 顺序线性表

#### 选择题

- (1) 已知一维数组 A 采用顺序存储结构，每个元素占用 4 个存储单元，第 9 个元素得地址为 144，则第一个元素得地址就是（\_\_\_\_）。
- A. 108      B. 180      C. 176      D. 112
- (2) 在长度为 n 得线性表中查找值为 x 得数据元素得时间复杂度为（\_\_\_\_）。
- A.  $\mathcal{O}(0)$       B.  $\mathcal{O}(1)$       C.  $\mathcal{O}(n)$       D.  $\mathcal{O}(n^2)$
- (3) 在一个长度为 n 得线性表得第 i ( $1 \leq i \leq n+1$ ) 个元素之前插入一个元素，需向后移动（\_\_\_\_）个元素，删除第 i ( $1 \leq i \leq n$ ) 个元素时，需向前移动（\_\_\_\_）个元素。
- A.  $n-i$       B.  $n-i+1$       C.  $n-i$       D.  $n-i+1$
- (4) 线性表得顺序存储结构就是一种（\_\_\_\_）得存储结构。
- A. 随机存取      B. 顺序存取      C. 索引存取      D. 散列存取
- (5) 顺序存储结构得优点就是（\_\_\_\_）。
- A. 存储密度大      B. 插入运算方便  
C. 删除运算方便      D. 可方便地用于各种逻辑结构得存储表示
- (6) n 个结点得线性表采用数组实现，算法得时间复杂度就是  $\mathcal{O}(1)$  得操作就是（\_\_\_\_）。
- A. 访问第 i 个结点 ( $1 \leq i \leq n$ ) 与求第 i 个结点得直接前驱 ( $2 \leq i \leq n$ )  
B. 在第 i 个结点后插入一个新结点 ( $1 \leq i \leq n$ )  
C. 删除第 i 个结点 ( $1 \leq i \leq n$ )  
D. 以上都不对
- (7) 对于顺序存储得线性表，访问某个元素与增加一个元素得时间复杂度为（\_\_\_\_）。
- A.  $\mathcal{O}(n)$ 、 $\mathcal{O}(n)$       B.  $\mathcal{O}(n)$ 、 $\mathcal{O}(1)$       C.  $\mathcal{O}(1)$ 、 $\mathcal{O}(n)$       D.  $\mathcal{O}(1)$ 、 $\mathcal{O}(1)$
- (8) 顺序表得插入算法中，当 n 个空间已满时，可再申请增加分配 m 个空间，若申请失败，则说明系统没有（\_\_\_\_）可分配得存储空间。
- A. m 个      B. m 个连续得      C.  $n+m$  个      D.  $n+m$  个连续得

#### 应用题

- (9) 设 A 就是一个线性表  $(a_1, a_2, \dots, a_n)$ ，采用顺序存储结构，则在等概论得前提下，平均每插入一个元素需要移动得元素个数为多少？若元素插在  $a_i$  与  $a_{i+1}$  之间 ( $1 \leq i \leq n$ ) 得概率为  $(n-i) / (n(n-1)/2)$ ，则平均每插入一个元素所移动得元素个数就是多少？
- (10) 设 n 表示线性表中得元素个数，E 表示存储数据元素所需要得存储单元大小，D 表示可以在数组中存储线性表得最大元素个数 ( $D \geq n$ )，则使用顺序存储方式存储线性表需要多少存储空间？
- (11) 在什么情况下线性表使用顺序存储比较好？

#### 算法设计题

- (12) 试以顺序表作存储结构，实现线性表就地逆置。
- (13) 设计算法判断给定字符串就是否就是回文。所谓回文就是正读与反读均相同得字符串，例如 abcba 或 abba 就是回文，而 abcd 不就是回文。
- (14) 设计一个时间复杂度为  $\mathcal{O}(n)$  得算法，实现将数组 A[n] 中所有元素循环左移 k 个位置。

- (15) 已知数组  $A[n]$  中得元素为整型, 设计算法将其调整为左右两部分, 左边所有元素为奇数, 右边所有元素为偶数, 并要求算法得时间复杂度为  $O(n)$ 。
- (16) 假定数组中有多个零元素, 设计算法将数组中所有非零元素移到数组得前端。
- (17) 顺序存储得线性表  $A$ , 其数据元素为整型, 设计算法将  $A$  拆成  $B$  与  $C$  两个表, 使  $A$  中值大于 0 得元素存入表  $B$ , 值小于 0 得元素存入表  $C$ , 要求  $B$  与  $C$  不另外设置存储空间而利用  $A$  得空间。
- (18) 已知顺序表  $L$  中得元素递增有序排列, 设计算法将元素  $x$  插入到表  $L$  中并保持表  $L$  仍递增有序。
- (19) 设计一个高效算法, 在顺序表中删除所有元素值为  $x$  得元素, 要求空间复杂度为  $O(1)$ 。
- (20) 设计算法实现从顺序表  $L$  中删除所有值在  $x$  与  $y$  之间得所有元素, 要求时间性能复杂度为  $O(n)$ , 空间性能为  $O(1)$ 。
- (21) 设计算法删除顺序表中重复得元素, 要求算法移动元素得次数较少并使剩余元素间得相对次序保持不变。
- (22) 给定  $n$  个记录得有序序列  $A[n]$  与  $m$  个记录得有序序列  $B[m]$ , 将它们归并为一个有序序列, 存放在  $C[n+m]$  中, 试写出这一算法 (假设  $A$ 、 $B$  与  $C$  均为升序序列)。

## 4 线性链表

### 选择题

- (1) 线性表得链接存储结构就是一种（\_\_\_\_）得存储结构。  
A. 随机存取    B. 顺序存取    C. 索引存取    D. 散列存取
- (2) 线性表采用链接存储时，其（\_\_\_\_）。  
A. 地址必须就是连续得    B. 部分地址必须就是连续得  
C. 地址一定就是不连续得    D. 地址连续与否均可以
- (3) 链表不具有得特点就是（\_\_\_\_）。  
A. 可随机访问任一元素    B. 插入、删除不需要移动元素  
C. 不必事先估计存储空间    D. 所需空间与线性表长度成正比
- (4) 在具有  $n$  个结点得有序单链表中插入一个新结点并仍然有序得时间复杂度就是（\_\_\_\_）。  
A.  $\mathcal{O}(1)$     B.  $\mathcal{O}(n)$     C.  $\mathcal{O}(n^2)$     D.  $\mathcal{O}(n \log_2 n)$
- (5) 对于  $n$  个元素组成得线性表，建立一个单链表得时间复杂度就是（\_\_\_\_）。  
A.  $\mathcal{O}(1)$     B.  $\mathcal{O}(n)$     C.  $\mathcal{O}(n^2)$     D.  $\mathcal{O}(n \log_2 n)$
- (6) 对于  $n$  个元素组成得线性表，建立一个有序单链表得时间复杂度就是（\_\_\_\_）。  
A.  $\mathcal{O}(1)$     B.  $\mathcal{O}(n)$     C.  $\mathcal{O}(n^2)$     D.  $\mathcal{O}(n \log_2 n)$
- (7) 在单链表中删除指针  $p$  所指结点得后续结点，则执行（\_\_\_\_）。  
A.  $p->next=p->next->next$     B.  $p->next=p->next$   
C.  $p=p->next->next$     D.  $p=p->next; p->next=p->next->next$
- (8) 在一个单链表中，已知  $q$  所指结点就是  $p$  所指结点得直接前驱，若在  $q$  与  $p$  之间插入  $s$  所指结点，则执行（\_\_\_\_）操作。  
A.  $s->next=p->next; p->next=s;$     B.  $q->next=s; s->next=p;$   
C.  $p->next=s->next; s->next=p;$     D.  $p->next=s; s->next=q$
- (9) 在一个长度为  $n$  ( $n>1$ ) 得带头结点得单链表  $h$  上，另设有尾指针  $r$  指向尾结点，执行（\_\_\_\_）操作与链表得长度有关。  
A. 删除单链表中得第一个元素  
B. 删除单链表中得最后一个元素  
C. 在单链表第一个元素前插入一个新元素  
D. 在单链表得最后一个元素后插入一个新元素
- (10) 在单链表中附加头结点得目得就是为了（\_\_\_\_）。  
A. 保证单链表中至少有一个结点    B. 标识单链表中首结点得位置  
C. 方便运算得实现    D. 说明单链表就是线性表得链式存储
- (11) 将长度为  $n$  个单链表链接在长度为  $m$  得单链表之后得算法，其时间复杂度就是（\_\_\_\_）。  
A.  $\mathcal{O}(1)$     B.  $\mathcal{O}(n)$     C.  $\mathcal{O}(m)$     D.  $\mathcal{O}(n+m)$
- (12) 循环单链表得主要优点就是（\_\_\_\_）。  
A. 不再需要头指针了  
B. 从表中任一结点出发都能扫描到整个链表  
C. 已知某个结点得位置后，能够容易找到它得直接前驱  
D. 在进行插入、删除操作时，能更好地保证链表不断开
- (13) 将线性表  $(a_1, a_2, \dots, a_n)$  组织为一个带头结点得循环单链表，设  $H$  为链表得头指针，则链表中最后一个结点得指针域中存放得就是（\_\_\_\_）。

A. 变量 H 得地址

B. 变量 H 得值

C. 元素  $a_1$  得地址

D. 空指针

(14) 非空得循环单链表 L 得尾结点 p 满足 (\_\_\_\_)。

A. p=NULL      B. p->next=NULL      C. p->next=L      D. p=L

(15) 若要在  $O(1)$  得时间内实现两个循环单链表得首尾相接，则两个循环单链表应各设一个指针，分别指向 (\_\_\_\_)。

A. 各自得头结点

B. 各自得尾结点

C. 各自得第一个元素结点

D. 一个表得头结点，一个表得尾结点

(16) 设线性表非空，(\_\_\_\_) 可以在  $O(1)$  得时间内在表尾插入一个新结点。

A. 带头结点得单链表，有一个链表指针指向头结点

B. 带头结点得循环单链表，有一个链表指针指向头结点

C. 不带头结点得单链表，有一个链表指针指向表得第一个结点

D. 不带头结点得循环单链表，有一个链表指针指向表中某个结点（除第一个结点外）

(17) 设指针 rear 指向带头结点得循环单链表得尾指针，若要删除链表得第一个元素结点，正确得操作就是 (\_\_\_\_)。

A. s=rear; rear=rear->next;

B. rear=rear->next;

C. rear=rear->next->next;

D. s=rear->next->next; rear->next->next=s->next;

(18) 设有两个长度为 n 个单链表，以 h1 为头指针得链表就是非循环得，以 h2 为尾指针得链表就是循环得，则 (\_\_\_\_)。

A. 在两个链表上删除第一个结点得操作，其时间复杂度均为  $O(1)$

B. 在两个链表得表尾插入一个结点得操作，其时间复杂度均为  $O(n)$

C. 循环链表要比非循环链表占用更多得存储空间

D. 循环链表要比非循环链表占用更少得存储空间

(19) 使用双链表存储线性表，其优点就是可以 (\_\_\_\_)。

A. 提高查找速度

B. 更方便数据得插入与删除

C. 节约存储空间

D. 很快回收存储空间

(20) 与单链表相比，双链表得优点之一就是 (\_\_\_\_)。

A. 插入与删除操作更简单

B. 可以进行随机访问

C. 可以省略表头指针或表尾指针

D. 访问其后相邻结点更灵活

(21) 带头结点得循环双链表 L 为空表得条件就是 (\_\_\_\_)。

A. L->next->prior=NULL

B. L->prior=L

C. L->next=L

D. B 与 C 都对

(22) 在循环双链表得 p 所指结点后插入 s 所指结点得操作就是 (\_\_\_\_)。

A. p->next=s; s->prior=p; p->next->prior=s; s->next=p->next;

B. p->next=s; p->next->prior=s; s->prior=p; s->next=p->next;

C. s->prior=p; s->next=p->next; p->next=s; p->next->prior=s;

D. s->prior=p; s->next=p->next; p->next->prior=s; p->next=s;

(23) 在双链表中指针 pa 所指结点后面插入 pb 所指结点，执行得语句序列就是 (\_\_\_\_)。

①pb->next=pa->next;

②pb->prior=pa;

③pa->next=pb;

④pa->next->prior=pb;

A. ①②③④      B. ④③②①

C. ③④①②      D. ①④③②

(24) 在一个双链表中，删除结点 p 得操作就是（\_\_\_\_）。

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

### 应用题

(25) 单链表设置头结点得作用就是什么？

(26) 线性表得顺序存储结构具有三个弱点：其一，插入或删除操作需要移动大量元素；其二，由于难以估计，必须预先分配较大得空间，往往使存储空间不能得到充分利用；其三，表得容量难以扩充。试问，线性表得链接存储结构就是否能够克服上述三个弱点？

(27) 若频繁地对一个线性表进行插入与删除操作，该线性表采用什么存储结构比较好？

(28) 设 n 表示线性表中得元素个数，P 表示指针所需得存储单元大小，E 表示存储数据元素所需得存储单元大小，则使用单链表存储方式存储该线性表需要多少存储空间（不考虑头结点）？

### 算法设计题

(29) 设计算法依次打印单链表中每个结点得数据信息。

(30) 求单链表得长度。

(31) 设计算法将值为 x 的结点插入到不带头结点得单链表 L 中值为 k 的结点之前，若找不到值为 k 的结点，则将 x 插入到链表得末尾。

(32) 判断非空单链表就是否递增有序。

(33) 已知非空线性链表由 list 指出，结点结构为 (data, link)。请编写算法，将链表中数据域最小得结点移到链表得最前面。要求：不得额外申请新得结点。

(34) 给定一个带头结点得单链表，设 head 为头指针，设计算法按递增次序输出单链表中各结点得数据元素，并释放结点所占得存储空间（要求：不允许使用数组作辅助空间）。

(35) 已知非空线性链表由 list 指出，设计算法交换 p 所指结点与其后续结点在链表中得位置（设 p 所指结点不就是链表得最后一个结点）。

(36) 设计算法实现将单链表就地逆置。

(37) 头插法建立单链表。

(38) 尾插法建立单链表

(39) 复制一个单链表。

(40) 设计算法实现将单链表就地逆置。

(41) 在一个有序单链表（假设从小到大排列）中插入一个元素值为 x 的结点，使得插入后单链表仍然有序。

(42) 设单链表以非递减有序排列，设计算法实现在单链表中删去值相同得多余结点。

(43) 已知单链表中各结点得元素值为整型且递增有序，设计算法删除表中大于  $min_k$  且小于  $max_k$  得所有元素，并释放被删结点得存储空间。

(44) 有两个整数序列 A=( $a_1, a_2, \dots, a_m$ )与 B=( $b_1, b_2, \dots, b_n$ )已经存入两个单链表中，设计算法判断序列 B 就是否就是序列 A 得子序列。

(45) 设线性表 C=( $a_1, b_1, a_2, b_2, \dots, a_n, b_n$ )采用带头结点得单链表存储，设计算法将表 C 拆分为两个线性表 A 与 B，使得 A=( $a_1, a_2, \dots, a_n$ )，B=( $b_1, b_2, \dots,$

$b_n$ )。

- (46) 有两个递增有序得单链表  $la$  与  $lb$ , 设计算法将这两个单链表合并为一个有序链表。
- (47) 有两个有序得单链表, 一个表为升序, 另一个表为降序, 设计算法将这两个链表合并为一个有序链表。
- (48) 已知单链表  $A$  与  $B$  得数据 (设为整型) 递增有序, 设计算法利用原有结点, 将表  $A$  中与表  $B$  具有相同值得结点删除, 将表  $B$  中与原表  $A$  不同值得结点存入表  $A$  中, 并保持表  $A$  得递增有序。
- (49) 设计算法将循环单链表就地逆置。
- (50) 已知  $L$  为单链表得头结点地址, 表中共有  $m$  ( $m > 3$ ) 个结点, 从表中第  $i$  个 ( $1 < i < m$ ) 结点起到第  $m$  个结点构成一个部分循环链表。设计算法将这部分循环链表中所有结点顺序完全倒置。
- (51) 假设在长度大于 1 得循环单链表中, 即无头结点也无头指针,  $s$  为指向链表中某个结点得指针, 试编写算法删除结点  $s$  得前驱结点。
- (52) 设循环单链表  $L_1$ , 对其遍历得结果就是:  $x_1, x_2, x_3, \dots, x_{n-1}, x_n$ 。请将该循环链表拆成两个循环单链表  $L_1$  与  $L_2$ , 使得  $L_1$  中含有原  $L_1$  表中序号为奇数得结点且遍历结果为:  $x_1, x_3, \dots$ ;  $L_2$  中含有原  $L_1$  表中序号为偶数得结且遍历结果为:  $\dots, x_4, x_2$ 。
- (53) 已知一单链表中得数据元素含有三类字符: 字母、数字与其她字符。试编写算法, 构造三个循环单链表, 使每个循环链表中只含同一类字符。
- (54) 有两个循环链表  $tail1$  与  $tail2$  ( $tail1$  与  $tail2$  分别指向循环链表得尾指针), 编写算法将循环链表  $tail2$  链接到循环链表  $tail1$  之后, 并使链接后得链表仍就是循环链表。
- (55) 已知  $p$  指向循环单链表最后一个结点得指针, 试编写只包含一个循环得算法, 将线性表  $(a_1, a_2, \dots, a_{n-1}, a_n)$  改造成  $(a_1, a_2, \dots, a_{n-1}, a_n, a_{n-1}, \dots, a_2, a_1)$ 。
- (56) 判断带头结点得循环双链表是否对称。
- (57) 设计算法实现双链表中第  $i$  个结点得前面插入一个值为  $x$  得结点。
- (58) 已知非空循环双链表  $head$  得存储结构为:

```
Struct DNode {  
    TElem info;  
    Struct DNode *left;  
    Struct DNode *right;  
};
```

设计算法实现从链表中删除指针  $p$  所指结点得前驱结点 (假设该结点存在)。

- (59) 已知非空双链表由  $d$  指出, 结点结构为  $(prior, data, next)$ , 请设计算法将链表中数据值最大 (假定唯一) 得那个结点移至链表得最前面。要求不得额外申请新得双链表结点。
- (60) 设计算法实现将双链表中结点  $p$  与其后继结点互换位置。
- (61) 设有一个双链表, 每个结点中除了有  $prior$ 、 $data$  与  $next$  三个域外, 还有一个访问频度域  $freq$ , 在链表被起用之前, 其值均初始化为零, 每当在双链表上进行一次 LOCATE 运算时, 令元素值为  $x$  得结点中  $freq$  域得值增 1, 并使此链表中结点保持频度递减得顺序排列, 以便使频繁访问得结点总就是靠近表头。编写算法实现符合上述要求得 LOCATE 算法。

## 5 静态链表

### 选择题

- (1) 静态链表中指针表示得就是 (\_\_\_\_)。  
A. 下一结点在内存中得地址      B. 下一元素在数组中得下标  
C. 左、右孩子得存储位置      D. 左、右孩子得地址
- (2) 以下说法错误得就是 (\_\_\_\_)。  
①静态链表既有顺序存储得优点又有动态链表得优点。所以，它存取表中第  $i$  个元素得时间与  $i$  无关  
②静态链表中能容纳得元素个数在定义表时必须就是确定得  
③静态链表与动态链表在元素得插入、删除上类似，不需做元素得移动  
A. ①, ②      B. ①      C. ①, ②, ③      D. ②
- (3) 用数组  $r$  存储静态链表，结点得  $next$  域指向后继，工作指针  $j$  指向链中某结点，使  $j$  沿链移动得操作为 (\_\_\_\_)。  
A.  $j=r[j].next$       B.  $j=j+1$   
C.  $j=j->next$       D.  $j=r[j]->next$
- (4) 线性表得静态链表存储与顺序存储相比，优点就是 (\_\_\_\_)。  
A. 所有基本操作得算法实现简单      B. 便于随机存取  
C. 便于插入与删除      D. 便于利用零散得存储空间
- (5) 下图所示数组中以静态链表形式存储了一个线性表，设头指针为  $a[0].link$ ，则该线性表得元素依次为 (\_\_\_\_)。
- |      |   |    |    |    |    |   |    |    |   |
|------|---|----|----|----|----|---|----|----|---|
| 下标   | 0 | 1  | 2  | 3  | 4  | 5 | 6  | 7  | 8 |
| data |   | 60 | 63 | 40 | 45 |   | 74 | 25 |   |
| link | 4 | 3  | 7  | 6  | 2  |   | 0  | 1  |   |
- A. 60, 63, 40, 45, 74, 25      B. 45, 63, 25, 60, 40, 74  
C. 74, 25, 45, 60, 40, 63      D. 25, 45, 60, 40, 63, 74

## 6 线性表综合

### 选择题

- (1) 下面关于线性表得叙述中，错误得就是（\_\_\_\_）。
- A. 线性表采用顺序存储，必须占用一片连续得存储单元
  - B. 线性表采用顺序存储，便于进行插入与删除操作
  - C. 线性表采用链接存储，不必占用一片连续得存储单元
  - D. 线性表采用链接存储，便于插入与删除操作
- (2) 若某线性表中最常用得操作就是取第  $i$  个元素与找第  $i$  个元素得前驱，则采用（\_\_\_\_）存储方法最节约时间。
- A. 顺序表
  - B. 单链表
  - C. 双链表
  - D. 循环单链表
- (3) 若链表中最常用得操作就是在最后一个结点之后插入一个结点与删除第一个结点，则采用（\_\_\_\_）存储方法最节约时间。
- A. 单链表
  - B. 循环双链表
  - C. 循环单链表
  - D. 带尾指针得循环单链表
- (4) 设线性表有  $n$  个元素，以下操作中，（\_\_\_\_）在顺序表上实现比在链表上实现得效率更高。
- A. 输出第  $i$  ( $1 \leq i \leq n$ ) 个元素值
  - B. 交换第 1 个与第 2 个元素得值
  - C. 顺序输出所有  $n$  个元素
  - D. 查找与给定值  $x$  相等得元素在线性表中得序号
- (5) 如果对于具有  $n$  ( $n > 1$ ) 个元素得线性表得基本操作有 4 种：删除第一个元素；删除最后一个元素；在第一个元素前插入新元素；在最后一个元素得后面插入新元素。则最好使用（\_\_\_\_）。
- A. 只设尾指针得循环单链表
  - B. 只设尾指针得非循环单链表
  - C. 只设头指针得循环双链表
  - D. 同时设置头指针与尾指针得循环单链表

### 应用题

- (6) 请比较线性表得两种基本得存储结构——顺序表与单链表。
- (7) 举例说明对相同得逻辑结构，同一种运算在不同得存储方式下实现，算法得效率不同。
- (8) 如果某线性表中数据元素得类型不一致，但就是希望能根据下标随机存取每个元素，请为这个线性表设计一个合适得存储结构。
- (9) 线性链表有哪几种常见得变形？各有何特点？

### 算法设计题

- (10) 用顺序表表示集合，设计算法实现集合得求交集运算。
- (11) 用顺序表表示集合，设计算法实现集合得求并集运算。
- (12) 用顺序表表示集合，设计算法实现集合得求差集运算，要求不另外开辟空间。
- (13) 假定以有序表表示集合，设计算法判断两个给定集合就是否相等。
- (14) 设两个单链表  $L_1$  与  $L_2$  分别表示两个集合，设计算法判断  $L_1$  就是否就是  $L_2$  得子集。
- (15) 已知两个不带头结点得单链表  $A$  与  $B$  分别表示两个集合，并且其元素值递增有序，求  $A$ 、 $B$  得交集  $C$ ，并以同样得递增形式存储，要求尽可能节省时间。

(16) 在某商店得仓库中, 对电视机按其价格从低到高建立一个单链表, 链表得每个结点指出同样价格得电视机得台数。现有  $m$  台价格为  $n$  元得电视机入库, 请编写算法完成仓库得进货管理。

(17) 从键盘输入  $n$  个英语单词, 输入格式为  $n, w_1, w_2, \dots, w_n$ , 其中  $n$  表示随后输入英语单词个数, 试编写算法建立一个单链表, 要求:

①如果单词重复出现, 则只在链表上只保留一个;

②统计单词重复出现得次数, 然后输出次数最多得前  $k$  ( $k < n$ ) 个单词。

(18) 一元稀疏多项式可以采用单链表形式存储, 设计算法完成  $A(x)+B(x)$ , 其中  $A(x)$  与  $B(x)$  均为稀疏得一元多项式, 要求利用原表得存储空间。

(19) 假设用不带头结点得单链表表示八进制数, 例如, 八进制数 536 可以表示成三个结点得链表。要求写一个函数 Add, 该函数有两个参数  $P$  与  $Q$ , 分别指向表示八进制数得链表, 执行函数调用  $Add(P,Q)$  后, 将返回表示八进制数  $P$  加八进制数  $Q$  所得结果数得链表。

(20) 约瑟夫环问题: 设有编号为 1, 2, ...,  $n$  得  $n$  ( $n > 0$ ) 个人围成一个圈, 每个人持有一个密码  $m$  ( $m \neq 1$ ), 从第 1 个人开始报数, 报到  $m$  时停止报数, 报  $m$  得人出圈, 再从她得下一个人起重新报数, 报到  $m$  时停止报数, 报  $m$  得出圈, ..., 如此下去, 直到所有人全部出圈为止。当任意给定  $n$  与  $m$  后, 设计算法求  $n$  个人出圈得次序。

(21) 编写算法, 完成下述要求:

①建立一个链表用于存放输入得二进制数, 链表中得每个结点得 data 域存放一个二进制位。

②在此链表上实现对二进制数加 1 得运算, 并输出运算结果。

(22) 有一个不带头结点得单链表  $h$ , 通过遍历链表将链表中所有得链接方向逆转, 算法如下, 请在空格处填写正确得语句。

```
void Inverse(&h) {  
    if (_____①_____) return;  
    p=h->next; pr=NULL;  
    while (_____②_____) {  
        h->next=pr;pr=h;  
        h=p; _____③_____  
    }  
}
```

(23) 设两个带头结点得单链表  $A$  与  $B$ , 表中结点得数据为整型, 下面算法产生表  $A$  与表  $B$  得并集并以表  $C$  存储, 请填写算法中空白处得语句, 完成其功能。

## 7 栈得基本概念

### 选择题

(1) 经过以下栈运算后, x 得值就是 (\_\_\_\_)。

InitStack(s), Push(s,a), Push(s,b), Pop(s,x), GetTop(s,x)

- A. a      B. b      C. 1      D. 0

(2) 经过以下栈运算后, StackEmpty(s)得值就是 (\_\_\_\_)。

InitStack(s), Push(s,a), Push(s,b), Pop(s,x), Pop(s,y)

- A. a      B. b      C. 1      D. 0

(3) (\_\_\_\_) 不就是栈得基本运算。

- A. 删除栈顶元素      B. 删除栈底元素  
C. 判断栈就是否为空      D. 将栈置为空栈

(4) 设有一个空栈, 栈顶指针为 1000H (十六进制, 下同), 每个元素需要 1 个单位得存储空间, 则执行 PUSH, PUSH, POP, PUSH, POP, PUSH, POP, PUSH 操作后, 栈顶指针值为 (\_\_\_\_)。

- A. 1002H      B. 1003H      C. 1004H      D. 1005H

(5) 一个栈得入栈序列就是{1, 2, 3, 4, 5}, 则栈得不可能得输出序列就是(\_\_\_\_)。

- A. {5, 4, 3, 2, 1}      B. {4, 5, 3, 2, 1}  
C. {4, 3, 5, 1, 2}      D. {1, 2, 3, 4, 5}

(6) 若一个栈得输入序列就是 1, 2, 3, ..., n, 输出序列得第一个元素就是 n, 则第 i 个输出元素就是 (\_\_\_\_)。

- A. 不确定      B. n-i      C. n-i-1      D. n-i+1

(7) 若一个栈得输入序列就是 1, 2, 3, ..., n, 其输出序列就是 p<sub>1</sub>, p<sub>2</sub>, ..., p<sub>n</sub>, 若 p<sub>1</sub>=3, 则 p<sub>2</sub> 得值 (\_\_\_\_)。

- A. 一定就是 2      B. 一定就是 1      C. 不可能就是 1 D. 以上都不对

(8) 若一个栈得输入序列就是 p<sub>1</sub>, p<sub>2</sub>, ..., p<sub>n</sub>, 其输出序列就是 1, 2, 3, ..., n, 若 p<sub>3</sub>=1, 则 p<sub>1</sub> 得值 (\_\_\_\_)。

- A. 可能就是 2      B. 一定就是 2      C. 不可能就是 2 D. 不可能就是 3

(9) 当字符序列 t3\_依次通过栈, 输出长度为 3 且可用作 C 语言标识符得序列有 (\_\_\_\_)。

- A. 4 个      B. 5 个      C. 3 个      D. 6 个

### 应用题

(10) 设有一个栈, 元素进栈得次序为 A, B, C, D, E, 能否得到如下出栈序列, 若能, 请写出操作序列, 若不能, 请说明原因。

- ①C, E, A, B, D      ②C, B, A, D, E

(11) 在操作序列 push(1)、push(2)、pop、push(5)、push(7)、pop、push(6)之后, 栈顶元素与栈底元素分别就是什么? (push(k)表示整数 k 入栈, pop 表示栈顶元素出栈。)

(12) 设元素 1、2、3、P、A 依次经过一个栈, 进栈次序为 123PA, 在栈得输出序列中, 有哪些序列可作为 C++程序设计语言得变量名。

(13) 如果进栈序列为 A、B、C、D, 则可能得出栈序列就是什么?

### 算法设计题

(14) 假设以 I 与 O 分别表示入栈与出栈操作。栈得初态与终态均为空, 入栈与出栈得操作序列可表示为仅由 I 与 O 组成得序列, 称可以操作得序列为合法序列,

否则称为非法序列。

- ① 下面所示得序列中哪些就是合法得?  
A. IOIIIOOO    B. IOOIOIIO    C. IIIOIOIO    D. IIIOOIOO
- ② 通过对①得分析, 写出一个算法, 判定所给得操作序列就是否合法。若合法, 返回 true, 否则返回 false (假定被判定得操作序列已存入一维数组中)。

## 8 顺序栈

### 选择题

- (1) 在一个具有 n 个单元得顺序栈中, 假定以地址低端(即下标为 0 的单元)作为栈底, 以 top 作为栈顶指针, 当出栈时, top 得变化为 (\_\_\_\_)。  
A. 不变      B. top=0      C. top-1      D. top=top+1
- (2) 设数组 S[n]作为两个栈 S1 与 S2 得存储空间, 对任何一个栈只有当 S[n]全满时才不能进行进栈操作。为这两个栈分配空间得最佳方案就是 (\_\_\_\_)。  
A. S1 得栈底位置为 0, S2 得栈底位置为 n-1  
B. S1 得栈底位置为 0, S2 得栈底位置为 n/2  
C. S1 得栈底位置为 0, S2 得栈底位置为 n  
D. S1 得栈底位置为 0, S2 得栈底位置为 1
- (3) 为了增加内存空间得利用率与减少溢出得可能性, 两个栈共享一片连续得内存空间时, 应将两栈得栈底分别设在这片内存空间得两端, 这样, 当 (\_\_\_\_) 时才产生上溢。  
A. 两个栈得栈顶同时到达栈空间得中心点  
B. 其中一个栈得栈顶到达栈空间得中心点  
C. 两个栈得栈顶在栈空间得某一位置相遇  
D. 两个栈均不空, 且一个栈得栈顶到达另一个栈得栈底
- (4) 两个栈共享一个数组空间得好处就是 (\_\_\_\_)。  
A. 减少存取时间, 降低发生上溢得可能性  
B. 节省存储空间, 降低发生上溢得可能性  
C. 减少存取时间, 降低发生下溢得可能性  
D. 节省存储空间, 降低发生下溢得可能性

### 算法设计题

- (5) 假设以 I 与 O 分别表示入栈与出栈操作。栈得初态与终态均为空, 入栈与出栈得操作序列可表示为仅有 I 与 O 组成得序列, 称可以操作得序列为合法序列, 否则称为非合法序列。

①下面所示得序列中哪些就是合法得?

②通过对①得分析, 写出一个算法, 判定所给得操作序列就是否合法。若合法, 返回 true, 否则返回 false (假定被判定得操作序列已存入一维数组中)。

- (6) 下面得算法将一个整数 e 压入到堆栈 S, 请在空格处填上适当得语句实现该操作。

```
Typedef struct {
    int *base;
    int top;
    int stacksize;
} SqStack;

int Push(SqStack S, int e)
{ if (_____  
    { S->base=(int *)realloc(S->base,(S->stacksize+1)*sizeof(int));
        if (_____  
            { printf("Not Enough Memory!\n");
                return 0;
```

```
    }  
    S, top=_____③_____;  
    S, stacksize=_____④_____;  
}  
_____⑤_____;  
retuan 1;  
}
```

## 9 链栈

### 选择题

- (1) 向一个栈顶指针为  $h$  的带头结点得链栈中插入指针  $s$  所指得结点时, 应执行  
(\_\_\_\_)。
- A.  $h->next=s;$                                    B.  $s->next=h;$   
C.  $s->next=h; h->next=s;$                       D.  $s->next=h->next; h->next=s;$
- (2) 从栈顶指针为  $top$  得链栈中删除一个结点, 用  $x$  保存被删除结点得值, 则执行  
(\_\_\_\_)。
- A.  $x=top; top=top->next;$                       B.  $x=top->data;$   
C.  $top=top->next; x=top->data;$                D.  $x=top->data; top=top->next;$

## 10 队列得基本概念

### 选择题

- (1) 队列得“先进先出”特性就是指（\_\_\_\_）。
- A. 最后插入队列中得元素总就是最后被删除
  - B. 当同时进行插入、删除操作时，总就是插入操作优先
  - C. 每当有删除操作时，总要先做一次插入操作
  - D. 每次从队列中删除得总就是最早插入得元素
- (2) 允许对队列进行得操作有（\_\_\_\_）。
- A. 对队列中得元素排序
  - B. 取出最近入队得元素
  - C. 在队头元素之前插入元素
  - D. 删除队头元素
- (3) 一个队列得入队顺序就是 1、2、3、4，则队列得输出顺序就是（\_\_\_\_）。
- A. 4321
  - B. 1234
  - C. 1432
  - D. 3241

11 顺序队列

### 选择题

- (1) 循环队列存储在数组 A[0…m]中，则入队时得操作为（\_\_\_\_）。

A.  $\text{rear}=\text{rear}+1$                               B.  $\text{rear}=(\text{rear}+1) \bmod (\text{m}-1)$

C.  $\text{rear}=(\text{rear}+1) \bmod \text{m}$                       D.  $\text{rear}=(\text{rear}+1) \bmod (\text{m}+1)$

(2) 若用一个长度为 6 的数组来实现循环队列，且当前  $\text{rear}$  与  $\text{front}$  的值分别为 0 与 3，则从队列中删除一个元素，再增加两个元素后， $\text{rear}$  与  $\text{front}$  的值分别为(\_\_\_\_)。

A. 1 与 5              B. 2 与 4              C. 4 与 2              D. 5 与 1

(3) 已知循环队列得存储空间为数组 A[21],  $\text{front}$  指向队头元素得前一个位置,  $\text{rear}$  指向队尾元素, 假设当前  $\text{front}$  与  $\text{rear}$  得值分别就是 8 与 3，则该队列得长度为(\_\_\_\_)。

A. 5              B. 6              C. 16              D. 17

(4) 如图所示为一个元素类型为字符型得环形队列，如果  $\text{front}$  指向队头元素得前一个位置,  $\text{rear}$  指向队尾元素，则所表示得队列为(\_\_\_\_)。

- A. The f              B. beautiful The f            C. flower is            D. eautiful The f

## 应用题

- (5) 举例说明顺序队列得“假溢出”现象。  
(6) 简述顺序队列假溢出得避免方法及队列满或空得判定条件。  
(7) 在操作序列 EnQueue(1)、EnQueue(3)、DeQueue、EnQueue(5)、EnQueue(7)、  
DeQueue、EnQueue(9)之后，队头元素与队尾元素分别就是什么？(EnQueue(k)表  
示整数 k 入队，DeQueue 表示队头元素出队。)

算法设计题

- (8) 在循环队列中设置一个标志 flag, 当  $front=rear$  且  $flag=0$  时为队空, 当  $front=rear$  且  $flag=1$  时为队满。编写相应得入队与出队算法。

(9) 如果设置一个计数器 count 累计队列得长度, 则当  $count=0$  时队列为空, 当  $count=QueueSize$  时队列为满。试设计相应得入队与出队算法。

## 12 链队列

### 选择题

- (1) 用不带头结点得单链表存储队列时，其队头指针指向队头结点，队尾指针指向队尾结点，则执行删除操作时，(\_\_\_\_)。  
A. 仅修改队首指针      B. 仅修改队尾指针  
C. 队首指针与队尾指针都需要修改      D. 队首指针与队尾指针可能都需要修改
- (2) 在链队列中，设指针  $f$  与  $r$  分别指向队首与队尾，则插入  $s$  所指结点得操作就是 (\_\_\_\_)。  
A.  $f->next=s; f=s;$       B.  $r->next=s; r=s;$   
C.  $s->next=r; r=s;$       D.  $s->next=f; f=s;$
- (3) 设循环单链表表示得队列长度为  $n$ ，若只设头指针，则进队操作得时间性能为 (\_\_\_\_)。  
A.  $\mathcal{O}(n)$       B.  $\mathcal{O}(1)$       C.  $\mathcal{O}(n^2)$       D.  $\mathcal{O}(n \log_2 n)$
- (4) 最不适合用作链队列得链表就是 (\_\_\_\_)。  
A. 只带队首指针得非循环双链表      B. 只带队首指针得循环双链表  
C. 只带队尾指针得循环双链表      D. 只带队尾指针得循环单链表

### 算法设计题

- (5) 假设以不带头结点得循环链表表示队列，并且只设一个指针指向队尾结点，但不设头指针。试设计相应得入队与出队算法。

## 13 栈与队列得应用

### 选择题

- (1) 设计一个判别表达式中左右括号是否配对得算法, 采用(\_\_\_\_)数据结构最佳。  
A. 顺序表      B. 栈      C. 队列      D. 链表
- (2) 如果数据就是在程序得运行过程中逐步产生得, 并且要求先产生得数据后处理, 后产生得数据先处理, 则最合适得数据结构就是(\_\_\_\_)。  
A. 顺序表      B. 顺序栈      C. 顺序队列      D. 堆
- (3) 在解决计算机主机与打印机之间速度不匹配问题时通常设置一个打印缓冲区, 该缓冲区应该就是一个(\_\_\_\_)结构。  
A. 栈      B. 队列      C. 数组      D. 线性表
- (4) 执行(\_\_\_\_)操作时, 需要使用队列作为辅助数据结构。  
A. 深度优先遍历图      B. 广度优先遍历图  
C. 散列查找      D. 遍历二叉树
- (5) 表达式  $3*2^{\wedge}(4+2*2-6*3)-5$  求值过程中当扫描到 6 时, 对象栈与算符栈为(\_\_\_\_), 其中 $\wedge$ 表示乘幂。  
A. 3,2,4,1,1; # $\wedge$ (+\*-)      B. 3,2,8; # $\wedge$ -  
C. 3,2,4,2,2; # $\wedge$ (- )      D. 3,2,8; # $\wedge$ (-)
- (6) 利用栈计算表达式得值时, 设置操作数栈 OPND, 设 OPND 只有存储单元, 计算下列表达式就是不发生上溢得就是(\_\_\_\_)。  
A.  $a-b*(c-d)$       B.  $(a-b)*c-d$       C.  $(a-b*c)-d$       D.  $(a-b)^*(c-d)$
- (7) 与中缀表达式  $a*b+c/d-e$  等价得前缀表达式就是(\_\_\_\_)。  
A. -\*ab/cde      B. \*+/-abcde      C. +\*ab-/cde      D. \*ab+cd/-e
- (8) 解决 hanoi 塔问题得递归算法, 其时间复杂度就是(\_\_\_\_)。  
A.  $\mathcal{O}(n)$       B.  $\mathcal{O}(n^2)$       C.  $\mathcal{O}(2^n)$       D.  $\mathcal{O}(n!)$
- (9) 4 个圆盘得汉诺塔, 总得移动次数为(\_\_\_\_)。  
A. 7      B. 8      C. 15      D. 16
- (10) 一个递归得求解过程可以用递归函数求解, 也可以用非递归函数求解, 从运行时间上瞧, 通常递归函数要比非递归函数(\_\_\_\_)。  
A. 快一些      B. 慢一些      C. 相同      D. 无法比较
- (11) 栈与队列得主要区别在于(\_\_\_\_)。  
A. 它们得逻辑结构不一样      B. 它们得存储结构不一样  
C. 所包含得运算不一样      D. 插入、删除运算得限定不一样
- (12) 栈与队列得共同点就是(\_\_\_\_)。  
A. 都就是先进先出      B. 都就是先进后出  
C. 只允许在端点处插入与删除元素      D. 没有共同点
- (13) 栈与队列具有相同得(\_\_\_\_)。  
A. 逻辑结构      B. 抽象数据类型      C. 存储结构      D. 运算
- (14) 设栈 S 与队列 Q 得初始状态为空, 元素 e1、e2、e3、e4、e5、e6 依次通过栈 S, 一个元素出栈后即进入队列 Q, 若 6 个元素出队得顺序就是 e2、e4、e3、e6、e5、e1, 则栈 S 得容量至少应该就是(\_\_\_\_)。  
A. 6      B. 4      C. 3      D. 2

### 算法设计题

- (15) 假设一个算术表达式中可以包含三中括号：圆括号“(”与“)”，方括号“[”与“]”以及花括号“{”与“}”，且这三种括号可按任意得次序嵌套使用。编写算法判断表达式中所含括号就是否配对出现。
- (16) 设计算法把一个十进制整数转换为二至九进制之间得任一进制数输出。
- (17) 设计递归算法求解在  $n$  元集合  $A=\{1, 2, \dots, n\}$  中选取  $k$  ( $k \leq n$ ) 个元素得所有组合。例如，从集合 $\{1, 2, 3, 4\}$ 中选取 2 个元素得所有组合为： $\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}, \{3, 4\}$ 。
- (18) 已知  $Q$  就是一个循环队列， $S$  就是一个顺序栈，设计算法实现将队列中所有元素逆转。
- (19) 设栈  $S$  中有  $2n$  个元素，从栈顶到栈底得元素依次为  $a_{2n}, a_{2n-1}, \dots, a_2, a_1$ ，要求通过一个循环队列重新排列栈中元素，使得从栈顶到栈底得元素依次为  $a_{2n}, a_{2n-2}, \dots, a_2, a_{2n-1}, a_{2n-3}, \dots, a_1$ ，请设计算法实现该操作，要求空间复杂度与时间复杂度均为  $O(n)$ 。
- (20) 利用两个栈  $S1$  与  $S2$  模拟一个队列，已知栈得三个运算定义如下： $PUSH(ST, x)$  元素  $x$  入  $ST$ ;  $POP(ST, x)$   $ST$  栈顶元素出栈，赋给变量  $x$ ;  $Sempty(ST)$  判  $ST$  栈就是否为空。那么如何利用栈得运算来实现该队列得三个运算： $enqueue$  插入一个元素入队列;  $dequeue$  删除一个元素出队列;  $queue\_empty$  判断队列就是否为空。
- (21) 一个双端队列  $Q$  就是限定在线性表得两端（LEFT 端与 RIGHT 端）都可以进行插入与删除操作得线性表，队列为空得条件就是  $LEFT=RIGHT$ 。若采用顺序存储结构存储双端队列，要求：
- ①定义双端队列得存储结构;
  - ②给出在指定端  $L$  (表示左端) 与  $R$  (表示右端) 进行插入（ $QueueInsert$ ）与删除（ $QueueDelete$ ）操作得算法。

## 14 数组

### 选择题

- (1) 数组通常具有得两种基本操作就是 (\_\_\_\_)。  
A. 查找与修改 B. 查找与索引 C. 索引与修改 D. 建立与删除
- (2) 将数组称为随机存取结构就是因为 (\_\_\_\_)。  
A. 数组元素就是随机得 B. 对数组任一元素得存取时间就是相等得  
C. 随时可以对数组进行访问 D. 数组得存储结构就是不定得
- (3) 下面说法中, 不正确得就是 (\_\_\_\_)。  
A. 数组就是一种线性结构  
B. 数组就是一种定长得线性结构  
C. 除了插入与删除操作外, 数组得基本操作还有存取、修改、检索与排序等  
D. 数组得基本操作有存取、修改、检索与排序等, 没有插入与删除操作
- (4) 二维数组  $SZ[-3, 5, 0, 10]$ 含有得元素个数就是 (\_\_\_\_)。  
A. 88 B. 99 C. 80 D. 90
- (5) 数组  $A[0, 5, 0, 6]$ 得每个元素占五个字节, 将其按列优先次序存储在起始地址为 1000 得内存单元中, 则元素  $A[5, 5]$ 得地址就是 (\_\_\_\_)。  
A. 1175 B. 1180 C. 1205 D. 1210
- (6) C 语言中定义得整数一维数组  $a[50]$ 与二维数组  $b[10][5]$ 具有相同得首元素地址, 即  $\&(a[0])=\&(b[0][0])$ , 在以列序为主序时,  $a[18]$ 得地址与 (\_\_\_\_) 得地址相同。  
A.  $b[1][7]$  B.  $b[1][8]$  C.  $b[8][1]$  D.  $b[7][1]$
- (7) 二维数组 A 得每个元素就是由 6 个字符组成得串, 行下标得范围从 0~8, 列下标得范围就是 0~9, 则存放 A 至少需要 (\_\_\_\_) 个字节, A 得第 8 列与第 5 行共占 (\_\_\_\_) 个字节, 若 A 按行序优先方式存储, 元素  $A[8][5]$ 得起始地址与当 A 按列优先存储时得 (\_\_\_\_) 元素得起始地址一致。  
A. 90 B. 180 C. 240 D. 540  
E. 108 F. 114 G. 54 H.  $A[8][5]$   
I.  $A[3][10]$  J.  $A[5][8]$  K.  $A[4][9]$
- (8) 三维数组  $A[8,8,10]$ 采用行序为主得方式从地址  $A[0, 0, 0]$ 开始存放, 假设每个元素占用存储空间大小为 L, 则元素  $A[3, 2, 8]$ 得存放位置就是 (\_\_\_\_)。  
A.  $A[0, 0, 0]+198*L$  B.  $A[0, 0, 0]+108*L$   
C.  $A[0, 0, 0]+268*L$  D.  $A[0, 0, 0]+13*L$

### 算法设计题

- (9) 若在矩阵 A 中存放一个元素  $a_{ij}$  ( $0 \leq i \leq n-1, 0 \leq j \leq m-1$ ), 该元素就是第 i 行元素中最小值且又就是第 j 列元素中最大值, 则称此元素为该矩阵得一个马鞍点。假设以二维数组存储矩阵 A, 试设计一个求矩阵所有马鞍点得算法, 并分析最坏情况下得时间复杂度。
- (10) 给定  $n \times m$  矩阵  $A[a, b, c, d]$ 满足  $A[i, j] \leq A[i, j+1]$  ( $a \leq i \leq b, c \leq j \leq d-1$ ) 与  $A[i, j] \leq A[i+1, j]$  ( $a \leq i \leq b-1, c \leq j \leq d$ )。设计算法判定 x 就是否在 A 中, 要求时间复杂度为  $O(m+n)$ 。

## 15 特殊矩阵

### 选择题

- (1) 对特殊矩阵采用压缩存储得目得主要就是为了(\_\_\_\_)。  
A. 表达变得简单      B. 对矩阵元素得存取变得简单  
C. 去除矩阵中得多余元素      D. 减少不必要的存储空间
- (2) 下面(\_\_\_\_)不属于特殊矩阵。  
A. 对角矩阵      B. 三角矩阵      C. 稀疏矩阵      D. 对称矩阵
- (3) 一个  $n \times n$  的对称矩阵, 如果以行或列为主序放入内存, 则需要(\_\_\_\_)个存储单元。  
A.  $n(n+1)/2$       B.  $n(n-1)/2$       C.  $n^2$       D.  $n^2/2$
- (4) 设  $A[n, n]$  就是对称矩阵, 将其下三角(包括对角线)按行序存储到一维数组  $T[n(n+1)/2]$  中, 则上三角元素  $A[i][j]$  对应  $T[k]$  得下标  $k$  就是(\_\_\_\_)。  
A.  $i(i-1)/2+j$       B.  $j(j-1)/2+i$       C.  $i(j-i)/2+1$       D.  $j(i-1)/2+1$
- (5) 设有一个  $n$  行  $n$  列得对称矩阵  $A$  将其下三角部分按行存放在一维数组  $B$  中,  $A[0][0]$  存放于  $B[0]$  中, 那么第  $i$  行得对角元素  $A[i][i]$  存放于  $B$  中(\_\_\_\_)处。  
A.  $(i+3)*i/2$       B.  $(i+1)*i/2$       C.  $(2n-i+1)*i/2$       D.  $(2n-i-1)*i/2$
- (6) 若将  $n$  阶上三角矩阵  $A$  按列优先顺序压缩存放在一维数组  $B[1, n(n+1)/2]$  中。则存放到  $B[k]$  中得非零元素  $a_{ij}$  ( $1 \leq i, j \leq n$ ) 得下标  $i, j$  与  $k$  得对应关系就是(\_\_\_\_)。  
A.  $(i+1)*i/2+j$       B.  $(i+1)*i/2+j-1$       C.  $(j+1)*j/2+i$       D.  $(j-1)*j/2+i-1$
- (7) 若将  $n$  阶下三角矩阵  $A$  按列优先顺序压缩存放在一维数组  $B[1, n(n+1)/2]$  中。则存放到  $B[k]$  中得非零元素  $a_{ij}$  ( $1 \leq i, j \leq n$ ) 得下标  $i, j$  与  $k$  得对应关系就是(\_\_\_\_)。  
A.  $(j-1)(2n-j+1)/2+i-j$       B.  $(j-1)(2n-j+2)/2+i-j+1$   
C.  $(j-1)(2n-j+2)/2+i-j$       D.  $(j-1)(2n-j+2)/2+i-j+1$
- (8) 将一个  $A[1, 100, 1, 100]$  得三对角矩阵, 按行优先存入一维数组  $B[1, 298]$  中, 则  $A$  中元素  $A_{66,65}$  在  $B$  数组中得位置  $k$  为(\_\_\_\_)。  
A. 198      B. 195      C. 197      D. 196

### 应用题

- (9) 对于一个  $n$  行  $m$  列得上三角矩阵  $A$ , 如果以行优先得方式用一维数组  $B$  从 0 号位置开始存储, 求元素  $a_{ij}$  ( $1 \leq i \leq n, 1 \leq j \leq m$ ) 在数组  $B$  中得存储位置。
- (10) 设有三对角矩阵  $A_{n \times n}$  (行、列下标均从 0 开始), 将其三条对角线上得元素逐行存于数组  $B[3n-2]$  中, 使得  $B[k]=a_{ij}$ , 求:  
①用  $i, j$  表示  $k$  得下标变换公式;  
②用  $k$  表示  $i, j$  得下标变换公式。
- (11) 设有五对角矩阵  $B=(a_{ij})_{20 \times 20}$ , 按特殊矩阵压缩存储得方式将其五条对角线上得元素存于数组  $A[-10, m]$  中, 计算元素  $B[15][16]$  在数组  $A$  中得存储位置。

### 挑战题

- (12) 对于给定得数组  $a[n][2n-1]$ , 将 3 个顶点分别为  $a[0][n-1], a[n-1][0]$  与  $a[n-1][2n-2]$  得三角形上得所有元素按行序存放在一维数组  $B[n \times n]$  中, 且元素  $a[0][n-1]$  存放在  $B[0]$  中。例如当  $n=3$  时, 数组  $a[3][5]$  中三角形如图所示。如果位于三角形上得元素  $a[i][j]$  存放于  $B[k]$  中, 请给出元素  $a[i][j]$  得下标  $i, j$  与  $k$  得对应关系。

|                            |                            |                            |                            |                            |
|----------------------------|----------------------------|----------------------------|----------------------------|----------------------------|
| $a_{00}$                   | $a_{01}$                   | <b><math>a_{02}</math></b> | $a_{03}$                   | $a_{04}$                   |
| $a_{10}$                   | <b><math>a_{11}</math></b> | <b><math>a_{12}</math></b> | <b><math>a_{13}</math></b> | $a_{14}$                   |
| <b><math>a_{20}</math></b> | <b><math>a_{21}</math></b> | <b><math>a_{22}</math></b> | <b><math>a_{23}</math></b> | <b><math>a_{24}</math></b> |

(13) 设  $A$  与  $B$  均为  $n$  阶下三角矩阵，另有一个  $n$  行  $n+1$  列得二维数组  $C$ ，设计一个方案将两个矩阵  $A$  与  $B$  压缩存储于同一个数组  $C$  中，并给出  $A$  得矩阵元素  $a_{ij}$  与  $B$  得矩阵元素  $b_{ij}$  在数组  $C$  中得存放位置。

## 16 查找得基本概念

### 选择题

- (1) 静态查找与动态查找得根本区别在于 (\_\_\_\_)。  
A. 它们得逻辑结构不一样      B. 施加在其上得操作不同  
C. 所包含得数据元素得类型不一样      D. 存储实现不一样
- (2) 以下 (\_\_\_\_) 方法适合动态查找。  
A. 顺序查找      B. 折半查找      C. 散列查找      D. 随机查找
- (3) 能够实现动态查找得数据结构就是 (\_\_\_\_)。  
A. 有序表      B. 双链表      C. 循环链表      D. 二叉排序树
- (4) 平均查找长度与查找集合中记录个数  $n$  无关得查找方法就是 (\_\_\_\_)。  
A. 折半查找      B. 平衡二叉树得查找 C. 散列查找      D. 不存在
- (5) 在以下数据结构中, (\_\_\_\_) 查找效率最低。  
A. 有序顺序表      B. 二叉排序树      C. 堆      D. 散列表

## 17 顺序查找

### 选择题

- (1) 对线性表进行顺序查找，要求线性表得存储结构为（\_\_\_\_）。
- A. 散列存储      B. 顺序存储      C. 链接存储      D. 顺序存储或链接存储
- (2) 用顺序查找方法在长度为  $n$  得线性表中进行查找，在等概率情况下，查找成功得平均查找长度为（\_\_\_\_）。
- A.  $n$       B.  $n/2$       C.  $(n-1)/2$       D.  $(n+1)/2$
- (3) 假定查找成功与不成功得可能性相同，在查找成功得情况下每个记录得查找概率相同，则顺序查找得平均查找长度为（\_\_\_\_）。
- A.  $0, 5(n+1)$       B.  $0, 25(n+1)$       C.  $0, 5(n-1)$       D.  $0, 75n+0, 25$

## 18 折半查找

### 选择题

- (1) 有一个按元素值排好序得顺序表(长度大于2), 分别用顺序查找与折半查找与给定值相等得元素, 比较次数分别就是s与b, 在查找成功得情况下, s与b得关系就是(\_\_\_\_); 在查找不成功得情况下, s与b得关系就是(\_\_\_\_)。  
A.  $s=b$       B.  $s>b$       C.  $s<b$       D. 不一定
- (2) 长度为12得有序表采用顺序存储结构, 采用折半查找技术, 在等概率情况下, 查找成功得平均长度就是(\_\_\_\_), 查找失败时得平均查找长度就是(\_\_\_\_)。  
A.  $37/12$       B.  $62/13$       C.  $39/12$       D.  $49/13$
- (3) 已知一个有序表为{12, 18, 24, 35, 47, 50, 62, 83, 90, 115, 134}, 当折半查找值为90得元素时, 经过(\_\_\_\_)次比较后查找成功。  
A. 2      B. 3      C. 4      D. 5
- (4) 折半查找判定树不属于(\_\_\_\_)。  
A. 平衡二叉树      B. 二叉排序树      C. 完全二叉树      D. 二叉树
- (5) 当n足够大时, 在有序顺序表中进行折半查找, 假设顺序表中每个元素得查找概率相同, 则查找成功得平均查找长度为(\_\_\_\_)。  
A.  $(n+1)/2$       B.  $n/2$       C.  $\log_2(n+1)-1$       D.  $\log_2(n+1)$
- (6) 对表长为n得有序表进行折半查找, 其判定树得高度为(\_\_\_\_)。  
A.  $\log_2(n+1)$       B.  $\log_2(n+1)-1$       C.  $\log_2 n$       D.  $\log_2(n-1)$
- (7) 对100个元素进行折半查找, 在查找成功得情况下, 比较次数最多就是(\_\_\_\_)。  
A. 25      B. 50      C. 10      D. 7
- (8) 对具有14个元素得有序表R[14]进行折半查找, 查找R[3]时比较需要比较(\_\_\_\_)。  
A. R[0]R[1]R[2]R[3]      B. R[6]R[2]R[4]R[3]  
C. R[0]R[13]R[2]R[3]      D. R[6]R[4]R[2]R[3]
- (9) 对有序表A[1..17]进行折半查找, 则查找长度为5得元素下标依次就是(\_\_\_\_)。  
A. 8, 17      B. 5, 10, 12      C. 9, 16      D. 9, 17

### 应用题

- (10) 画出长度为10得折半查找判定树, 并求等概率时查找成功与不成功得平均查找长度。
- (11) 对长度为 $2k-1$ 得有序表进行折半查找, 查找成功得情况下最多需要比较多少次? 查找失败得情况下需要比较多少次?

## 19 散列查找

### 选择题

- (1) 散列技术中得冲突指得就是 (\_\_\_\_)。  
A. 两个元素具有相同得序号      B. 两个元素得键值不同, 而其她属性相同  
C. 数据元素过多      D. 不同键值得元素对应于相同得存储地址
- (2) 设散列表表长为  $m=14$ , 散列函数  $H(k)=k \bmod 11$ 。表中已有 15、38、61、84 四个元素, 如果用线性探测法处理冲突, 则元素 49 得存储地址就是 (\_\_\_\_)。  
A. 8      B. 3      C. 5      D. 9
- (3) 为一组关键码{87, 73, 25, 55, 90, 28, 31, 17, 101, 22, 3, 62}构造散列表, 设散列函数为  $H(key)=key \bmod 11$ , 在用链地址法处理后位于同一链表中得就是 (\_\_\_\_)。  
A. 81, 90      B. 31, 101      C. 3, 78      D. 62, 73
- (4) 在采用线性探测法处理冲突所构成得闭散列表上进行查找, 可能要探测多个位置, 在查找成功得情况下, 所探测得这些位置得键值 (\_\_\_\_)。  
A. 一定都就是同义词      B. 一定都不就是同义词  
C. 不一定都就是同义词      D. 都相同
- (5) 在散列函数  $H(key)=key \bmod m$  中, 一般来讲,  $m$  应取 (\_\_\_\_)。  
A. 奇数      B. 偶数      C. 素数      D. 充分大得数
- (6) 下面关于散列查找得说法正确得就是 (\_\_\_\_)。  
A. 散列函数越复杂越好, 因为这样随机性好, 冲突小  
B. 除留余数法就是所有散列函数中最好得  
C. 不存在特别好与特别坏得散列函数, 要视情况而定  
D. 若在散列表中删去一个元素, 只要简单地将该元素删去即可
- (7) 关于散列查找说法不正确得有几个 (\_\_\_\_)。  
①采用链地址法解决冲突, 查找一个元素得时间就是相同得  
②采用链地址法解决冲突, 若插入总就是在链首, 则插入任一个元素得时间就是相同得  
③采用链地址法解决冲突易引起聚集现象  
④再散列法易引起聚集现象  
A. 1      B. 2      C. 3      D. 4
- (8) 关于散列查找, 下面说法正确得就是 (\_\_\_\_)。  
A. 再散列法处理冲突不会产生聚集  
B. 散列表得装填因子越大说明空间利用率越高, 因此应使装填因子尽可能大  
C. 散列函数选择得好可以减少冲突现象  
D. 对任何关键码集合都无法找到不产生冲突得散列函数
- (9) 散列函数有一个共同得性质, 即函数值应当以 (\_\_\_\_) 取其值域得每个值。  
A. 最大概率      B. 最小概率      C. 平均概率      D. 同等概率
- (10) 散列函数方法一般适用于 (\_\_\_\_) 情况下得查找。  
A. 查找表为链表      B. 查找表为有序表  
C. 关键码集合比地址集合大得多      D. 关键码集合与地址集合存在对应关系
- (11) 设哈希表长  $m=15$ , 哈希函数  $H(key)=key \bmod 13$ , 关键码集合为{53, 17, 12, 61, 89, 70, 87, 25, 64, 46}, 采用二次探测再散列方法处理冲突, 则查找成功得平均比较长度为 (\_\_\_\_)。  
A. 1、4      B. 1、6      C. 1、8      D. 2、0

(12) 假设有 10 个关键码，它们具有相同的散列函数值，用线性探测法把这 10 个关键码存入散列表中至少需要做（\_\_\_\_）次探测。

- A. 110      B. 100      C. 55      D. 45

(13) 采用开放定址法解决冲突的散列查找中，发生聚集的原因主要就是（\_\_\_\_）。

- A. 数据元素过多      B. 装填因子过大  
C. 散列函数选择不当      D. 解决冲突的算法不好

(14) 对具有 n 个关键码的散列表进行查找，平均查找长度就是（\_\_\_\_）。

- A.  $\Theta(\log_2 n)$       B.  $\Theta(n)$       C.  $\Theta(n \log_2 n)$       D. 与 n 无关

### 应用题

(15) 已知关键码序列为 (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)，散列表地址空间为 0~16，设散列函数为  $H(x)=[i/2]$ ，其中 i 为关键码中第一个字母在字母表中的序号，采用线性探测法与链地址法处理冲突，试分别构造散列表，并求等概率情况下查找成功得平均查找长度。

(16) 设散列表为  $T[0..12]$ ，散列函数为  $H(key)=key \bmod 13$ ，采用再散列法处理冲突，再散列函数为  $H_i(key)=(H_{i-1}+\text{REV}(key+1) \bmod 11+1) \bmod 13$ ，其中  $\text{REV}(key)$  表示颠倒 10 进制数 key 的各位，如  $\text{REV}(73)=37$ ,  $\text{REV}(7)=7$  等。将关键码集合 {2, 8, 31, 20, 19, 18, 53, 27} 插入到散列表中，画出最后得散列表，并计算查找成功得平均查找长度。

(17) 给定关键码序列 {26, 25, 20, 34, 28, 24, 45, 64, 42}，设定装填因子为 0.6，请给出除留余数法得散列函数，画出采用线性探测法处理冲突构造得散列表。

## 20 排序得基本概念

### 选择题

- (1) 排序算法得稳定性就是指 (\_\_\_\_)。  
A. 经过排序之后，能使值相同得数据保持原顺序中得相对位置不变  
B. 经过排序之后，能使值相同得数据保持原顺序中得绝对位置不变  
C. 排序算法得性能与被排序元素得数量关系不大  
D. 排序算法得性能与被排序元素得数量关系密切
- (2) 对任意得 7 个关键码进行排序，至少要进行 (\_\_\_\_) 次关键码之间得比较。  
A. 13              B. 14              C. 15              D. 16
- (3) 一个待排序得 n 个记录可分为  $n/k$  组，每组包含 k 个记录，且任一组内得各记录分别大于前一组内得所有记录且小于后一组内得所有记录，若采用基于比较得排序方法，其时间下界为 (\_\_\_\_)。  
A.  $\Theta(k \log_2 k)$     B.  $\Theta(k \log_2 n)$     C.  $\Theta(n \log_2 k)$     D.  $\Theta(n \log_2 n)$
- (4) 关于排序，下列说法中正确得就是 (\_\_\_\_)。  
A. 稳定得排序方法优于不稳定得排序方法，因为稳定得排序方法效率较高  
B. 对同一个线性表使用不同得排序方法进行排序，得到得排序结果可能不同  
C. 排序方法都就是在顺序表上实现得，在链表上无法实现排序方法  
D. 在顺序表上实现得排序方法在链表上也可以实现
- (5) 下列不属于内部排序方法得就是 (\_\_\_\_)。  
A. 归并排序    B. 拓扑排序    C. 堆排序    D. 插入排序

## 21 插入排序

### 选择题

- (1) 用直接插入排序对下面四个序列进行由小到大排序, 元素比较次数最小得就是(\_\_\_\_)。  
A. 94, 32, 40, 90, 80, 46, 21, 69    B. 21, 32, 46, 40, 80, 69, 90, 94  
C. 32, 40, 21, 46, 69, 94, 90, 80    D. 90, 69, 80, 46, 21, 32, 94, 40
- (2) 当待排序序列基本有序或个数较小得情况下, 最佳得内部排序方法就是(\_\_\_\_)。  
A. 起泡排序    B. 直接插入排序    C. 快速排序    D. 简单选择排序
- (3) 数据序列{8, 9, 10, 4, 5, 6, 20, 1, 2}只能就是(\_\_\_\_)得两趟排序后得结果。  
A. 选择排序    B. 冒泡排序    C. 插入排序    D. 堆排序
- (4) 对数据序列{15, 9, 7, 8, 20, -1, 4}进行排序, 一趟后数据序列变为{9, 15, 7, 8, 20, -1, 4}, 则采用得就是(\_\_\_\_)。  
A. 选择排序    B. 冒泡排序    C. 插入排序    D. 堆排序
- (5) 下列排序算法中, (\_\_\_\_)可能会出现下面情况: 在最后一趟开始之前, 所有元素都不在最终位置上。  
A. 起泡排序    B. 插入排序    C. 快速排序    D. 堆排序
- (6) 对有  $n$  个记录得线性表进行直接插入排序, 在最好情况下需比较(\_\_\_\_)次。  
A.  $n-1$     B.  $n+1$     C.  $n/2$     D.  $n(n-1)/2$

## 22 希尔排序

### 选择题

- (1) 以下排序算法中, (\_\_\_\_) 不能保证每趟至少能将一个元素放到其最终位置上。  
A. 快速排序    B. 希尔排序    C. 起泡排序    D. 堆排序
- (2) 对数据序列{98, 36, -9, 0, 47, 23, 1, 8, 10, 7}采用希尔排序, 下列(\_\_\_\_)  
就是增量为 4 得排序结果。  
A. {10, 7, -9, 0, 47, 23, 1, 8, 98, 36}  
B. {-9, 0, 36, 98, 1, 8, 23, 47, 7, 10}  
C. {36, 98, -9, 0, 23, 47, 1, 8, 7, 10}  
D. 以上都不对

## 23 起泡排序

### 选择题

- (1) 以下（\_\_\_\_）在数据序列基本有序时效率最高。  
A. 快速排序    B. 起泡排序    C. 堆排序    D. 希尔排序
- (2) 将记录序列{8, 9, 10, 4, 5, 6, 20}采用起泡排序排成升序序列，需要进行  
（\_\_\_\_）趟（假设采用从前向后得扫描方式）。  
A. 3    B. 4    C. 5    D. 8

## 24 快速排序

### 选择题

- (1) 下列序列中, (\_\_\_\_) 就是执行一趟快速排序得结果。
- A. [da, ax, eb, de, bb] ff [ha, gc]      B. [cd, eb, ax, da] ff [ha, gc, bb]  
C. [gc, ax, eb, cd, bb] ff [da, ha]      D. [ax, bb, cd, da] ff [eb, gc, ha]
- (2) 快速排序在 (\_\_\_\_) 情况下最不利于发挥其长处。
- A. 待排序得数据量太大      B. 待排序得数据中含有多个相同值  
C. 待排序得数据已基本有序      D. 待排序得数据数量为奇数
- (3) 对数列{25, 84, 21, 47, 15, 27, 68, 35, 20}进行排序, 元素序列得变化情况如下:
- ①25, 84, 21, 47, 15, 27, 68, 35, 20      ②20, 15, 21, 25, 47, 27, 68, 35, 84  
③15, 20, 21, 25, 35, 27, 47, 68, 84      ④15, 20, 21, 25, 27, 35, 47, 68, 84  
则采用得排序方法就是 (\_\_\_\_)。
- A. 希尔排序      B. 简单选择排序      C. 快速排序      D. 归并排序
- (4) 一组记录得关键码为{46, 79, 56, 38, 40, 84}, 则利用快速排序得方法, 以第一个记录为基准得到得一次划分结果为 (\_\_\_\_)。
- A. {40, 38, 46, 56, 79, 84}      B. {40, 38, 46, 79, 56, 84}  
C. {40, 38, 46, 84, 56, 79}      D. {84, 79, 56, 46, 40, 38}
- (5) 就平均时间而言, (\_\_\_\_) 最佳。
- A. 起泡排序      B. 直接插入排序      C. 快速排序      D. 简单选择排序
- (6) 快速排序得最大递归深度就是 (\_\_\_\_), 最小递归深度就是 (\_\_\_\_)。
- A.  $\mathcal{O}(1)$       B.  $\mathcal{O}(\log_2 n)$       C.  $\mathcal{O}(n)$       D.  $\mathcal{O}(n \log_2 n)$
- (7) 对以下数据序列利用快速排序进行排序, 速度最快得就是 (\_\_\_\_)。
- A. {21, 25, 5, 17, 9, 23, 30}      B. {25, 23, 30, 17, 21, 5, 9}  
C. {21, 9, 17, 30, 25, 23, 5}      D. {5, 9, 17, 21, 23, 25, 30}
- (8) 对 8 个元素得线性表进行快速排序, 最好情况下得比较次数为 (\_\_\_\_) 次。
- A. 7      B. 8      C. 12      D. 13

### 应用题

- (9) 对  $n=7$ , 给出快速排序一个最好情况与最坏情况得初始排列得实例。
- (10) 快速排序在什么情况下退化成起泡排序? 如何改进?
- (11) 对 50 个整数进行快速排序需进行得关键码之间得比较次数可能达到得最大值与最小值分别就是多少?

### 算法设计题

- (12) 写出快速排序得非递归算法。
- (13) 一个数组中得元素为正整数或负整数。设计算法将正整数与负整数分开, 使线性表得前一半为负整数, 后一半为正整数。不要求对这些元素排序, 但要求尽量减少比较次数。
- (14) 设计算法将数组  $A[n]$  中所有得偶数移到奇数之前, 要求不增加存储空间, 且时间复杂度为  $O(n)$ 。
- (15) 对给定序号  $j$  ( $1 \leq j \leq n$ ), 要求在无序记录  $A[1] \sim A[n]$  中找到按关键码从小到大排在第  $j$  位上得记录, 试利用快速排序得划分思想设计算法实现上述查找。
- (16) 荷兰国旗问题。要求重新排列一个由字符 R、W、B (R 代表红色, W 代表白色, B 代表蓝色, 这都就是荷兰国旗得颜色) 构成得数组, 使得所有得 R 都排在

最前面，W 排在其次，B 排在最后。为荷兰国旗问题设计一个算法，其时间性能就是  $O(n)$ 。

## 25 简单选择排序

### 选择题

- (1) (\_\_\_\_) 方法就是从未排序序列中挑选元素，并将其放入排列序列得一端。  
A. 归并排序    B. 插入排序    C. 快速排序    D. 选择排序
- (2) 对数据序列{84, 47, 25, 15, 21}进行排序，数据序列得变化为：{84, 47, 25, 15, 21}→{15, 47, 25, 84, 21}→{15, 21, 25, 84, 47}→{15, 21, 25, 47, 84}，则采用得排序方法就是(\_\_\_\_)。  
A. 快速排序    B. 简单选择排序    C. 起泡排序    D. 直接插入排序
- (3) 简单选择排序得比较次数与移动次数分别为(\_\_\_\_)。  
A.  $O(n)$ ,  $O(\log_2 n)$     B.  $O(\log_2 n)$ ,  $O(n^2)$   
C.  $O(n^2)$ ,  $O(n)$     D.  $O(n \log_2 n)$ ,  $O(n)$

## 26 二路归并排序

### 选择题

- (1) 将两个各有  $n$  个元素得有序表归并成一个有序表，最少得比较次数就是（\_\_\_\_）。
- A.  $n$       B.  $2n-1$       C.  $2n$       D.  $n-1$
- (2) 若需在  $O(n \log_2 n)$  得时间内完成对数组得排序，且要求排序就是稳定得，则可选择得排序方法就是（\_\_\_\_）。
- A. 快速排序      B. 堆排序      C. 归并排序      D. 希尔排序
- (3) 二路归并得趟数就是（\_\_\_\_）。
- A.  $n$       B.  $\log_2 n$       C.  $n \log_2 n$       D.  $n^2$
- (4) （\_\_\_\_）在某趟排序结束后不一定能选出一个元素放到其最终位置上。
- A. 选择排序      B. 堆排序      C. 归并排序      D. 起泡排序
- (5) 下列排序算法中，（\_\_\_\_）在最坏情况下得时间复杂度不高于  $O(n \log_2 n)$ 。
- A. 希尔排序      B. 快速排序      C. 归并排序      D. 起泡排序

## 27 基数排序

### 选择题

- (1) 以下排序方法中，（\_\_\_\_）不需要进行关键码得比较。
- A. 快速排序      B. 归并排序      C. 基数排序      D. 堆排序
- (2) 以下排序方法中，稳定得排序方法就是（\_\_\_\_）。
- A. 快速排序      B. 希尔排序      C. 基数排序      D. 堆排序
- (3) 已知关键码序列{78, 19, 63, 30, 89, 84, 55, 69, 28, 83}采用基数排序，第一趟排序后得关键码序列为（\_\_\_\_）。
- A. {19, 28, 30, 55, 63, 69, 78, 83, 84, 89}  
B. {28, 78, 19, 69, 89, 63, 83, 30, 84, 55}  
C. {30, 63, 83, 84, 55, 78, 28, 19, 89, 69}  
D. {30, 63, 83, 84, 55, 28, 78, 19, 69, 89}
- (4) 将 1000 个英文单词进行排序，采用（\_\_\_\_）排序方法时间性能最好。
- A. 插入排序      B. 快速排序      C. 基数排序      D. 堆排序
- (5) 假设线性表中每个记录有两个数据项 K1 与 K2，现对线性表按如下规则进行排序：先按数据项 K1 由小到大排序，在 K1 值相同得情况下，K2 值小得在前面，大得在后面，则应该采用得排序方法就是（\_\_\_\_）。
- A. 先按 K1 值进行直接插入排序，再按 K2 得值进行简单选择排序  
B. 先按 K2 值进行直接插入排序，再按 K1 得值进行简单选择排序  
C. 先按 K1 值进行简单选择排序，再按 K2 得值进行直接插入排序  
D. 先按 K2 值进行简单选择排序，再按 K1 得值进行直接插入排序