

第1章 绪论

一、选择题

1.B	2.C	3.1C	3.2B	4.B	5.D	6.C	7.C	8.D	9.D	10.A	11.C
12.D	13.D	14.A	15.C	16.A	17.C						

二、判断题

1. ×	2. ×	3. ×	4. ×	5. √	6. ×	7. ×	8. √	9. ×	10. ×	11. ×	12. √
13. ×											

三、填空题

1. 数据元素 数据元素间关系 2. 集合 线性结构 树形结构 图状结构或网状结构。

3. 数据的组织形式，即数据元素之间逻辑关系的总体。而逻辑关系是指数据元素之间的关联方式或称“邻接关系”。

4. 表示（又称映像）。 5. (1) 逻辑特性 (2) 在计算机内部如何表示和实现 (3) 数学特性。

6. 算法的时间复杂度和空间复杂度。 7. (1) 逻辑结构 (2) 物理结构 (3) 操作（运算） (4) 算法。

8. (1) 有穷性 (2) 确定性 (3) 可行性。

9. (1) $n+1$ (2) n (3) $n(n+3)/2$ (4) $n(n+1)/2$ 。

10. $1 + (1+2) + (1+2+3) + \dots + (1+2+\dots+n) = n(n+1)(n+2)/6$ $O(n^3)$

11. $\log_2 n$ 12. $n \log_2 n$ 13. $\log_2 n^2$ 14. $(n+3)(n-2)/2$ 15. $O(n)$

16. ① (1) 1 (2) 1 (3) $f(m, n-1)$ (4) n ② 9 17. $n(n-1)/2$

四、应用题

1. 数据结构是一门研究在非数值计算的程序设计问题中，计算机的操作对象及对象间的关系和施加于对象的操作等的学科。

2. 四种表示方法

(1) 顺序存储方式。数据元素顺序存放，每个存储结点只含一个元素。存储位置反映数据元素间的逻辑关系。存储密度大，但有些操作（如插入、删除）效率较差。

(2) 链式存储方式。每个存储结点除包含数据元素信息外还包含一组（至少一个）指针。指针反映数据元素间的逻辑关系。这种方式不要求存储空间连续，便于动态操作（如插入、删除等），但存储空间开销大（用于指针），另外不能折半查找等。

(3) 索引存储方式。除数据元素存储在一地址连续的内存空间外，尚需建立一个索引表，索引表中索引指示存储结点的存储位置（下标）或存储区间端点（下标），兼有静态和动态特性。

(4) 散列存储方式。通过散列函数和解决冲突的方法，将关键字散列在连续的有限的地址空间内，并将散列函数的值解释成关键字所在元素的存储地址，这种存储方式称为散列存储。其特点是存取速度快，只能按关键字随机存取，不能顺序存取，也不能折半存取。

3. 数据类型是程序设计语言中的一个概念，它是一个值的集合和操作的集合。如 C 语言中的整型、实型、字符型等。整型值的范围（对具体机器都应有整数范围），其操作有加、减、乘、除、求余等。实际上数据类型是厂家提供给用户的已实现了的数据结构。“抽象数据类型（ADT）”指一个数学模型及定义在该模型上的一组操作。“抽象”的意义在于数据类型的数学抽象特性。抽象数据类型的定义仅取决于它的逻辑特性，而与其在计算机内部如

何表示和实现无关。无论其内部结构如何变化，只要它的数学特性不变就不影响它的外部使用。抽象数据类型和数据类型实质上是一个概念。此外，抽象数据类型的范围更广，它已不再局限于机器已定义和实现的数据类型，还包括用户在设计软件系统时自行定义的数据类型。使用抽象数据类型定义的软件模块含定义、表示和实现三部分，封装在一起，对用户透明（提供接口），而不必了解实现细节。抽象数据类型的出现使程序设计不再是“艺术”，而是向“科学”迈进了一步。

4. (1) 数据的逻辑结构反映数据元素之间的逻辑关系（即数据元素之间的关联方式或“邻接关系”），数据的存储结构是数据结构在计算机中的表示，包括数据元素的表示及其关系的表示。数据的运算是数据定义的一组操作，运算是定义在逻辑结构上的，和存储结构无关，而运算的实现则是依赖于存储结构。

(2) 逻辑结构相同但存储不同，可以是不同的数据结构。例如，线性表的逻辑结构属于线性结构，采用顺序存储结构为顺序表，而采用链式存储结构称为线性链表。

(3) 栈和队列的逻辑结构相同，其存储表示也可相同（顺序存储和链式存储），但由于其运算集合不同而成为不同的数据结构。

(4) 数据结构的评价非常复杂，可以考虑两个方面，一是所选数据结构是否准确、完整的刻划了问题的基本特征；二是是否容易实现（如对数据分解是否恰当；逻辑结构的选择是否适合于运算的功能，是否有利于运算的实现；基本运算的选择是否恰当。）

5. 评价好的算法有四个方面。一是算法的正确性；二是算法的易读性；三是算法的健壮性；四是算法的时空效率（运行）。

6. (1) 见上面题 3 (2) 见上面题 4 (3) 见上面题 3

(4) 算法的时间复杂性是算法输入规模的函数。算法的输入规模或问题的规模是作为该算法输入的数据所含数据元素的数目，或与此数目有关的其它参数。有时考虑算法在最坏情况下的时间复杂度或平均时间复杂度。

(5) 算法是对特定问题求解步骤的描述，是指令的有限序列，其中每一条指令表示一个或多个操作。算法具有五个重要特性：有穷性、确定性、可行性、输入和输出。

(6) 频度。在分析算法时间复杂度时，有时需要估算基本操作的原操作，它是执行次数最多的一个操作，该操作重复执行的次数称为频度。

7. 集合、线性结构、树形结构、图形或网状结构。 8. 逻辑结构、存储结构、操作（运算）。

9. 通常考虑算法所需要的存储空间量和算法所需要的时间量。后者又涉及到四方面：程序运行时所需输入的数据总量，对源程序进行编译所需时间，计算机执行每条指令所需时间和程序中指令重复执行的次数。

10. D 是数据元素的有限集合，S 是 D 上数据元素之间关系的有限集合。

11. “数据结构”这一术语有两种含义，一是作为一门课程的名称；二是作为一个科学的概念。作为科学概念，目前尚无公认定义，一般认为，讨论数据结构要包括三个方面，一是数据的逻辑结构，二是数据的存储结构，三是对数据进行的操作（运算）。而数据类型是值的集合和操作的集合，可以看作是已实现了的数据结构，后者是前者的一种简化情况。

12. 见上面题 2。

13. 将学号、姓名、平均成绩看成一个记录（元素，含三个数据项），将 100 个这样的记录存于数组中。因一般无增删操作，故宜采用顺序存储。

```
typedef struct
{int num;//学号
char name[8];//姓名
float score;/平均成绩
```

```

    }node;
    node student[100];

```

14. 见上面题 4 (3)。

15. 应从两方面进行讨论：如通讯录较少变动（如城市私人电话号码），主要用于查询，以顺序存储较方便，既能顺序查找也可随机查找；若通讯录经常有增删操作，用链式存储结构较为合适，将每个人的情况作为一个元素（即一个结点存放一个人），设姓名作关键字，链表安排成有序表，这样可提高查询速度。

16. 线性表中的插入、删除操作，在顺序存储方式下平均移动近一半的元素，时间复杂度为 $O(n)$ ；而在链式存储方式下，插入和删除时间复杂度都是 $O(1)$ 。

17. 对算法 A1 和 A2 的时间复杂度 T1 和 T2 取对数，得 $n \log^2$ 和 $2 \log^n$ 。显然，算法 A2 好于 A1。

18. **struct** node

```

    {int year, month, day; };
    typedef struct
    {int num; //帐号
      char name[8]; //姓名
      struct node date; //开户年月日
      int tag; //储蓄类型, 如: 0- 零存, 1- 一年定期.....
      float put; //存入累加数;
      float interest; //利息
      float total; //帐面总数
    }count;

```

19. (1) n (2) $n+1$ (3) n (4) $(n+4)(n-1)/2$ (5) $(n+2)(n-1)/2$ (6) $n-1$

这是一个递归调用，因 k 的初值为 1，由语句 (6) 知，每次调用 k 增 1，故第 (1) 语句执行 n 次。(2) 是 FOR 循环语句，在满足 (1) 的条件下执行，该语句进入循环体 (3) n 次，加上最后一次判断出界，故执行了 $n+1$ 次。(4) 也是循环语句，当 $k=1$ 时判断 $n+1$ 次（进入循环体 (5) n 次）， $k=2$ 时判断 n 次，最后一次 $k=n-1$ 时判断 3 次，故执行次数是 $(n+1) + n + \dots + 3 = (n+4)(n-1)/2$ 次。语句 (5) 是 (4) 的循环体，每次比 (4) 少一次判断，故执行次数是 $n + (n-1) + \dots + 2 = (n+2)(n-1)/2$ 次。注意分析时，不要把 (2) 分析成 n 次，更不是 1 次。

20. 4（这时 $i=4$ ， $s=100$ ） REPEAT 语句先执行循环体，后判断条件，直到条件为真时退出循环。

21. 算法在最好情况下，即二进制数的最后一位为零时，只作一次判断，未执行循环体，赋值语句 $A[i]$ 执行了一次；最坏情况出现在二进制数各位均为 1（最高位为零，因题目假设无溢出），这时循环体执行了 $n-1$ 次，时间复杂度是 $O(n)$ ，循环体平均执行 $n/2$ 次，时间复杂度仍是 $O(n)$ 。

22. 该算法功能是将原单循环链表分解成两个单循环链表：其一包括结点 h 到结点 g 的前驱结点；另一个包括结点 g 到结点 h 的前驱结点。时间复杂度是 $O(n)$ 。

23. 第一层 FOR 循环判断 $n+1$ 次，往下执行 n 次，第二层 FOR 执行次数为 $(n+(n-1)+(n-2)+\dots+1)$ ，第三层循环体受第一层循环和第二层循环的控制，其执行次数如下表：

$i=$	1	2	3	...	n
$j=n$	n	n	n	...	n
$j=n-1$	$n-1$	$n-1$	$n-1$...	
...		

j=3 3 3
 j=2 2 2
 j=1 1

执行次数为 $(1+2+\dots+n)+(2+3+\dots+n)+\dots+n=n*(n+1)/2-n(n^2-1)/6$ 。在 $n=5$ 时, $f(5)=55$, 执行过程中, 输出结果为: $sum=15, sum=29, sum=41, sum=50, sum=55$ (每个 $sum=$ 占一行, 为节省篇幅, 这里省去换行)。

24. $O(n^2)$, m 的值等于赋值语句 $m:=m+1$ 的运行次数, 其计算式为 $\sum_{i=1}^{n/2} (n-2i+1) = \frac{n^2}{4}$

25. (1) $O(1)$ (2) $O(n^2)$ (3) $O(n^3)$ 26. (1) $O(n)$ (2) $O(n^2)$

27. (1) 由斐波那契数列的定义可得:

$$\begin{aligned} F_n &= F_{n-1} + F_{n-2} \\ &= 2F_{n-2} + F_{n-3} \\ &= 3F_{n-3} + 2F_{n-4} \\ &= 5F_{n-4} + 3F_{n-5} \\ &= 8F_{n-5} + 5F_{n-6} \\ &\dots\dots \\ &= pF_1 + qF_0 \end{aligned}$$

设 F_m 的执行次数为 B_m ($m=0, 1, 2, \dots, n-1$), 由以上等式可知, F_{n-1} 被执行一次, 即 $B_{n-1}=1$; F_{n-2} 被执行两次, 即 $B_{n-2}=2$; 直至 F_1 被执行 p 次、 F_0 被执行 q 次, 即 $B_1=p, B_0=q$ 。 B_m 的执行次数为前两等式第一因式系数之和, 即 $B_m=B_{m-1}+B_{m-2}$, 再有 $B_{n-1}=1$ 和 $B_{n-2}=2$, 这也是一个斐波那契数列。可以解得:

$$B_m = \frac{\sqrt{5}}{5} \left[\left(\frac{1+\sqrt{5}}{2} \right)^{n-m+2} - \left(\frac{1-\sqrt{5}}{2} \right)^{n-m+2} \right] \quad (m=0, 1, 2, \dots, n-1)$$

(2) 时间复杂度为 $O(n)$

28. 从小到大排列为: $\log n, n^{1/2} + \log n, n, n \log n, n^2 + \log n, n^3, n - n^3 + 7n^5, 2^{n/2}, (3/2)^n, n!, \binom{2n}{n}$