



数据结构与算法

Data Structure and Algorithm

第1章 绪论



目 录

1.1 什么是数据结构

1.2 相关的基本术语

1.3 算法及其评价

1.4 退出

1.1 什么是数据结构

众所周知，计算机的程序是对信息进行加工处理。在大多数情况下，这些信息并不是没有组织，信息（数据）之间往往具有重要的结构关系，这就是数据结构的内容。那么，什么是数据结构呢？先看以下几个例子。

例1、电话号码查询系统

设有一个电话号码簿，它记录了 N 个人的名字和其相应的电话号码，假定按如下形式安排：

$(a_1, b_1)(a_2, b_2) \dots (a_n, b_n)$

其中 $a_i, b_i (i=1, 2 \dots n)$ 分别表示某人的名字和对应的电话号码要求设计一个算法，当给定任何一个人的名字时，该算法能够打印出此人的电话号码，如果该电话簿中根本就没有这个人，则该算法也能够报告没有这个人的标志。

1.1 什么是数据结构

算法的设计，依赖于计算机如何存储人的名字和对应的电话号码，或者说依赖于名字和其电话号码的结构。

数据的结构，直接影响算法的选择和效率。

上述的问题是一种数据结构问题。可将名字和对应的电话号码设计成：二维数组、表结构、向量。

假定名字和其电话号码逻辑上已安排成N元向量的形式，它的每个元素是一个数对 (a_i, b_i) ， $1 \leq i \leq n$

崔文靖	7214156
何文颖	6562890
李淑芳	2348293
刘 丽	2356767
石宝国	7892345
魏永鸣	5674967
吴承志	7233445
赵胜利	4665555
张会有	7455566

1.1 什么是数据结构

例2 书目自动检索系统

线性表

书目文件

001	高等数学	樊映川	S01
002	理论力学	罗远祥	L01
003	高等数学	华罗庚	S01
004	线性代数	栾汝书	S02
.....	作者名:..

索引表

按书名

分类号:

按作者名

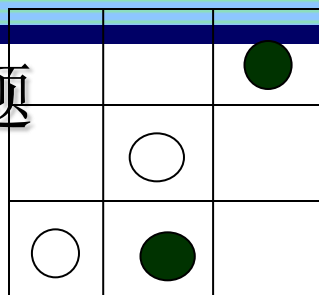
按分类号

高等数学	001, 003.	出版单位:	樊映川	001, ...
理论力学	002,	出版时间:	华罗庚	002,
线性代数	004,	价格:	栾汝书	004,
.....

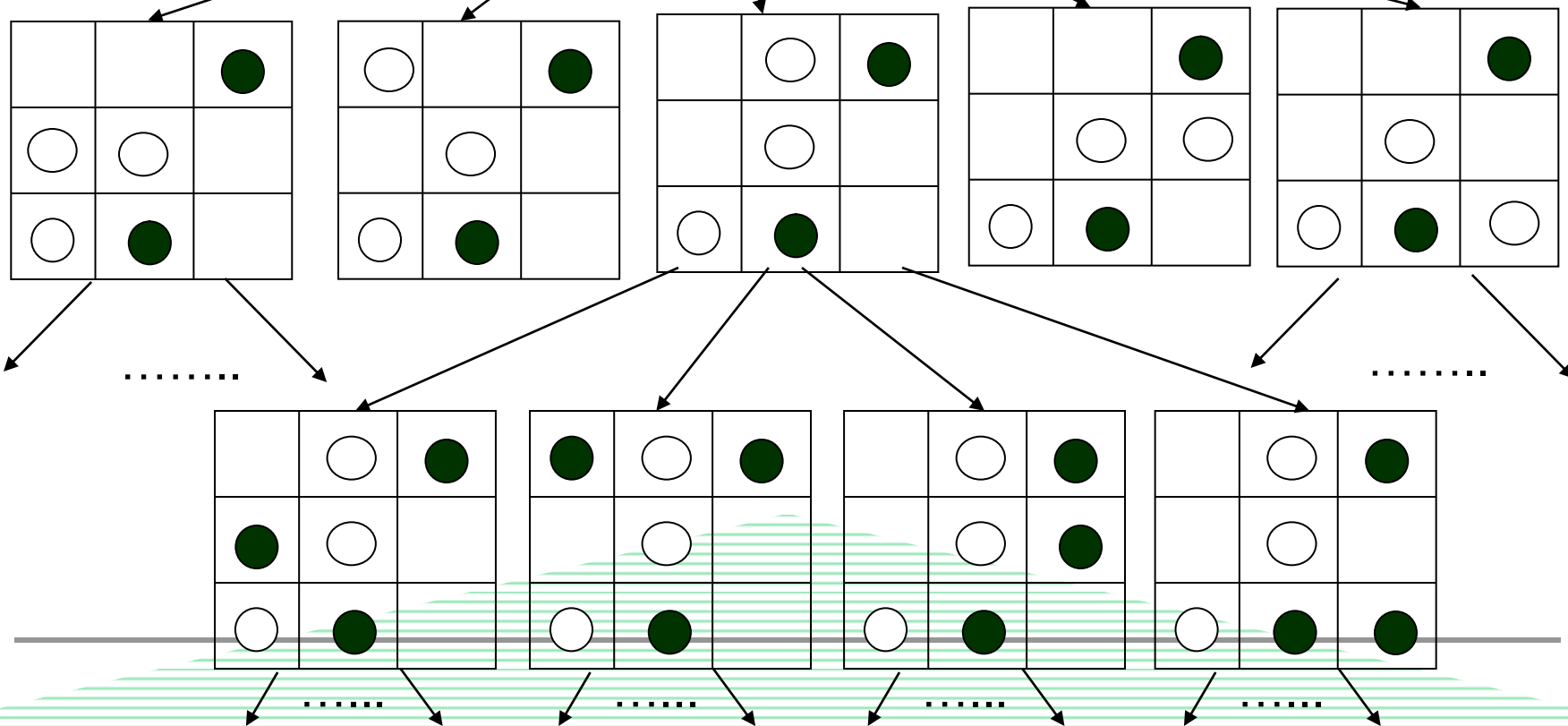
L	002, ...
S	001, 003,
.....

1.1 什么是数据结构

— 例3 人机对奕问题



树

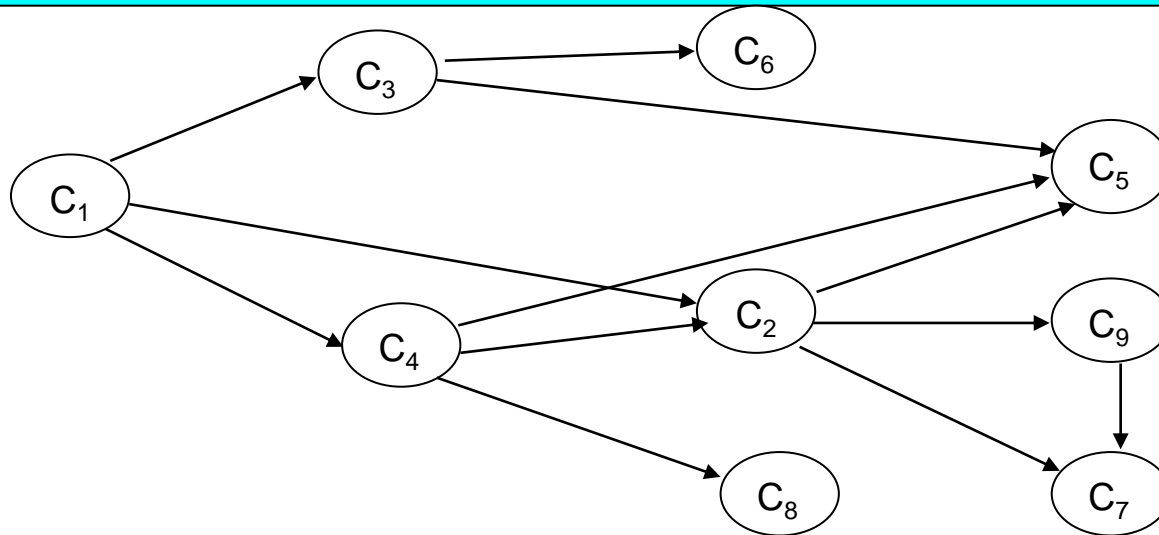


例4、教学计划编排问题

图

课程编号	课程名称	先修课程
C ₁	计算机导论	无
C ₂	数据结构	C ₁ , C ₄
C ₃	汇编语言	C ₁
C ₄	C程序设计语言	C ₁
C ₅	计算机图形学	C ₂ , C ₃ , C ₄
C ₆	接口技术	C ₃
C ₇	数据库原理	C ₂ , C ₉
C ₈	编译原理	C ₄
C ₉	操作系统	C ₂

(a) 计算机专业的课程设置



(b) 表示课程之间优先关系的有向图

图1.3 教学计划编排问题的数据结构

1.1 什么是数据结构

1.1.1 定义:

指同一数据元素类中各数据元素之间存在的关系.

1.1.2 组成部分:

数据的逻辑结构:简称数据结构

数据的存储结构

数据的运算



数据的逻辑结构

定义：**对数据之间关系的描述**，相互之间存在一种或多种特定关系的数据元素的集合。

数据的逻辑结构的形式定义为：是一个二元组， $\text{Data_Structure} = (D, S)$

其中：D是数据元素的有限集

S是D上关系的有限集。

例：复数的数据结构定义如下：

$$\text{Complex} = (C, R)$$

其中：C是含两个实数的集合 $\{C1, C2\}$ ，分别表示复数的实部和虚部。

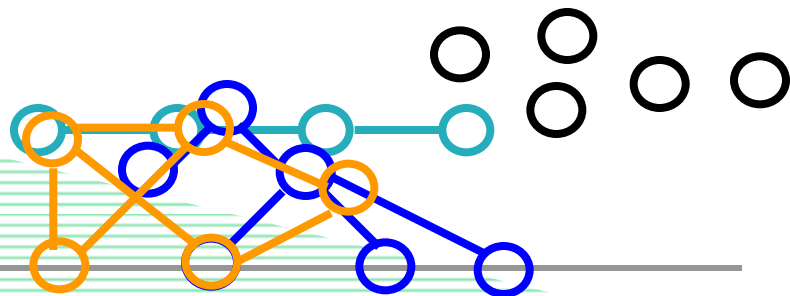
$R = \{P\}$ ，P是定义在集合上的一种关系 $\{\langle C1, C2 \rangle\}$ 。



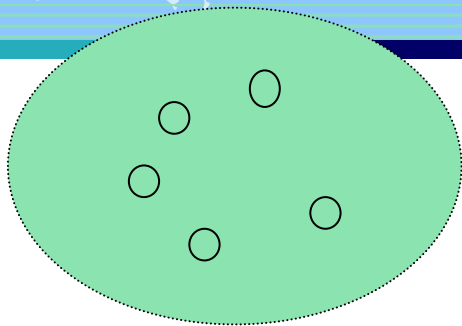
数据的逻辑结构

根据数据元素间关系的不同特性，通常有下列四类基本的结构：

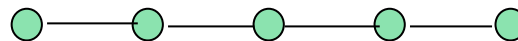
- (1) **集合结构**。在集合结构中，数据元素间的关系是“属于同一个集合”，别无其他关系。集合是元素关系极为松散的一种结构。
- (2) **线性结构**。该结构的数据元素之间存在着一对一的关系。
(第2—5章)
- (3) **树型结构**。该结构的数据元素之间存在着一对多的关系。
(第6章)
- (4) **图形结构**。该结构的数据元素之间存在着多对多的关系，图形结构也称作网状结构。(第7章)



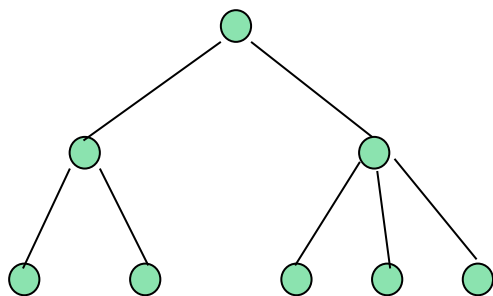
数据的逻辑结构



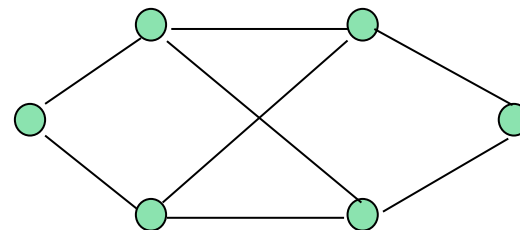
(a) 集合结构



(b) 线性结构



(c) 树型结构



(d) 图形结构

四类基本结构的示意图



数据的存储结构

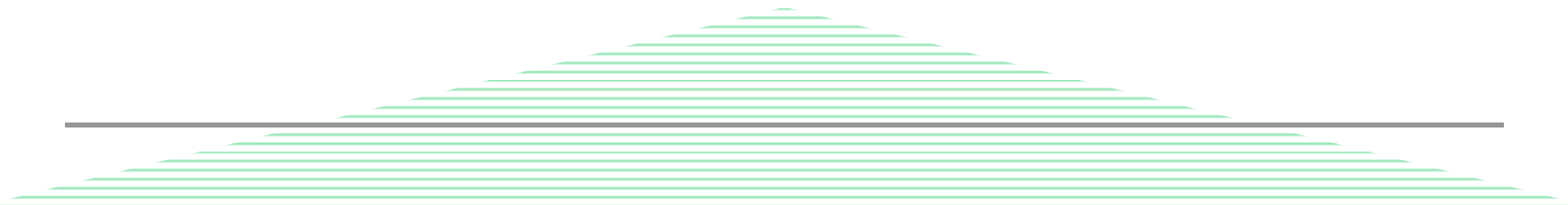
定义:数据的逻辑结构在**计算机存储器**中的实现。

与逻辑结构的**区别**:

逻辑结构是从逻辑关系上观察数据,与数据在计算机中的存储没有任何关系,是**独立于计算机**的,而存储结构是**依赖于计算机**的.

与逻辑结构的**联系**:

两者**密切相关**,任何一个算法的设计取决于选定的逻辑结构,而算法的实现依赖于采用的存储结构.



数据的存储结构

分类: 顺序存储结构、链式存储结构

- **顺序存储结构:** 是把逻辑上相邻的元素存储在物理位置相邻的存储单元中，由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法，通常借助于程序设计语言中的数组来实现。
- **链式存储结构:** 对逻辑上相邻的元素不要求其物理位置相邻，元素间的逻辑关系通过附设的指针字段来表示，由此得到的存储表示称为链式存储结构，链式存储结构通常借助于程序设计语言中的指针类型来实现。

又可分为: 顺序方法、链式方法、索引方法、散列方法，后三中为非顺序方法。



1.2 相关的基本术语

1. 数据（data）

数据是对信息的一种符号表示。在计算机科学中是指所有能输入到计算机中并被计算机程序处理的符号的总称。

例如：数字、字母、汉字、图形、图像、声音都称为数据。

2. 数据元素（data element）

数据元素是组成数据的基本单位，不是组成数据的最小单位。

一个数据元素可由若干个数据项（Data Item）组成，例如，学籍管理系统中学生信息表的每一个数据元素就是一个学生记录。它包括学生的学号、姓名、性别、籍贯、出生年月、成绩等数据项。这些数据项可以分为两种：一种叫做初等项，如学生的性别、籍贯等，这些数据项是在数据处理时不能再分割的最小单位；另一种叫做组合项，如学生的成绩，它可以再划分为数学、物理、化学等更小的项。

3. 数据对象 (data object)

是性质相同的数据元素组成的集合，是数据的一个子集。

例如，整数数据对象的集合可表示为 $N=\{0, \pm 1, \pm 2, \dots\}$ ，字母字符数据对象的集合可表示为 $C=\{'A', 'B', \dots, 'Z'\}$ 。

4. 数据类型 (data type)

4.1 定义：是一组性质相同的值的集合以及定义于这个值集合上的一组操作的总称。

例如，C语言中用到的整数数据类型(int a;)，是指由-32768到32767中值构成的集合及一组操作（加、减、乘、除、乘方等）的总称。

4.2 分类：可分为两类，一类是原子类型，另一类则是结构类型。

原子类型的值是不可分解的。如C语言中整型、字符型、浮点型、双精度型等基本类型，分别用保留字int、char、float、double标识。

结构类型的值是由若干成分按某种结构组成的，因此是可分解的，并且它的成分可以是非结构的，也可以是结构的。例如，数组的值由若干分量组成，每个分量可以是整数，也可以是数组等，其他还有结构、联合、指针、枚举型、自定义。

5. 抽象数据类型 (Abstract Data Type)

是指一个数学模型以及定义在该模型上的一组操作。

5.1 定义方式

用三元组描述如下：(D, S, P)

在本书中，采用如下书写格式：

ADT 抽象数据类型名{

数据对象: <对数据对象的定义>

数据关系: <数据对象的定义>

基本操作: <操作声明>

}ADT 抽象数据类型名

基本操作的定义:

基本操作名(参数表)

初始条件: <初始条件描述>

操作条件: <操作结果描述>



5. 抽象数据类型 (Abstract Data Type)

例:抽象数据类型三元组的定义 P9

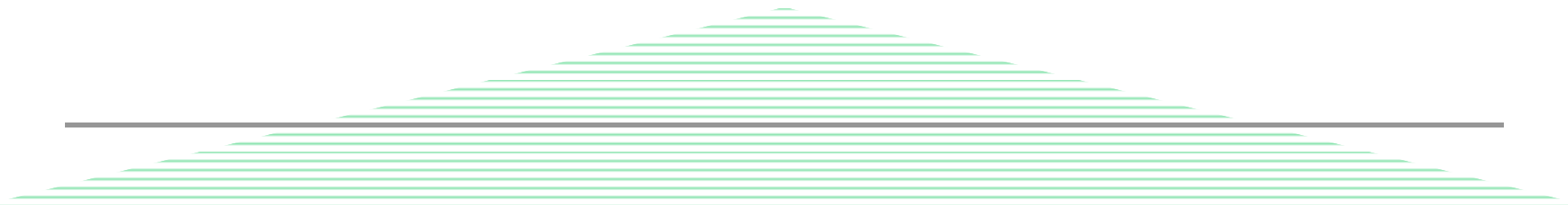
说明:

a. **ElemSet**:定义了关系运算的某个集合，即有个确定的，将由用户自行定义的，含某个关系运算的数据对象。

b. 参数表:

赋值参数: 提供输入;

引用参数: "&"打头, 不仅提供输入, 而且返回操作结果。



5.2 表示和实现方式-----类C语言

三个注意：

- 类C与C语言表示的区别之处（见书第10页）
- 赋值语句、if分支语句和while(或for)循环语句是最基本的三种语句，仅此三种语句就足以对付一切算法的设计了，不仅“足够”且“最好”。Switch语句为广义if语句，在分支条件反复时可用，提高可读性，鼓励使用。
- 动态存储分配函数

calloc() 分配n个数据项的内存空间，每个数据项的大小为size

例如： `p=(char *)calloc(100,sizeof(char));`

free() 释放指针p所指向的内存空间，p所指向的内存空间必须是用**calloc,malloc,realloc**所分配的内存，如果p为NULL或指向不存在的内存块则不做任何操作。

例如： `free(p);`

malloc() 分配size字节的存储区

例如： `p=(char *)malloc(100);`

realloc() 改变已分配内存空间的大小

例如： `p=(char *)realloc(p,256);`



数据结构的三个方面：

数据的逻辑结构

线性结构

线性表

栈

队

非线性结构

树形结构

图形结构

数据的存储结构

顺序存储

链式存储

数据的运算：检索、排序、插入、删除、修改等

1.3 算法及其评价

1. **算法**：是对特定问题求解步骤的一种描述

算法是指令的有限序列，其中每一条指令表示一个或多个操作。

2. 算法和程序的关系

算法的含义与程序十分相似，但二者是有区别的。**程序中的指令必须是机器可执行的，而算法中的指令则无此限制。**一个算法若用计算机语言来书写，则它就可以是一个程序。

3. 基本特性：

- a. 有穷性 一个算法必须总是在执行有穷步之后结束，且每一步都在有穷时间内完成。
- b. 确定性 算法中每一条指令必须有确切的含义。不存在二义性。且算法只有一个入口和一个出口。
- c. 可行性 一个算法是可行的。即算法描述的操作都是可以通过已经实现的基本运算执行有限次来实现的。
- d. 输入 一个算法有零个或多个输入，这些输入取自于某个特定的对象集合。
- e. 输出 一个算法有一个或多个输出，这些输出是同输入有着某些特定关系的量。

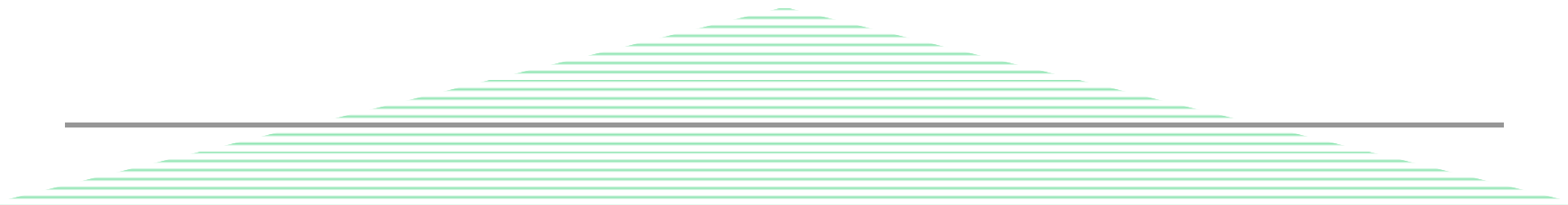


1.3 算法及其评价

算法输入和输出可以通过三种途径现：

- a.通过scanf和printf语句实现
- b.以算法头中参数表里显式列出的参量作为输入/出的媒介
- c.通过全局变量或外部变量隐式传递信息（避免使用）

b和c特点： 实现了算法与其调用者之间的信息交换



1.3 算法及其评价

3. 评价算法

四个方面：正确性、运行时间、占用空间、简单性

1.3.1 时间复杂度

指算法中包含简单操作次数, 一般不必精确计算出算法的时间, 只要大致计算出相应数量级, 如 $O(1)$ 、 $O(n)$ 、 $O(n\log_2 n)$ 、 $O(n^2)$ 、 $O(n^3)$ 、 $O(2n)$ 等。

```
for (i=1, i<=n; ++i) //两个N*N矩阵相乘
    for (j=1; j<=n; ++j)
    {
        c[i][j]=0;
        for (k=1; k<=n; ++k)
            c[i][j] += a[i][k] * b[k][j];
    }
```

由于是一个三重循环, 每个循环从1到n, 则总次数为: $n \times n \times n = n^3$
时间复杂度为 $T(n) = O(n^3)$

1.3 算法及其评价

频度：是指该语句重复执行的次数

算法的运行时间：一个算法中所有语句的频度之和。

例1、`{++x ; s=0;}`

将`x`自增看成是基本操作，则语句频度为 1，即时间复杂度为 $O(1)$

如果将`s=0`也看成是基本操作，则语句频度为 2，其时间复杂度仍为 $O(1)$ ，即常量阶。

例2、`for (i=1; i<=n; ++i)`

`{++x; s+=x;}`

语句频度为： $2n$ 其时间复杂度为： $T(n)=O(n)$

即时间复杂度为线性阶。

例3、`i=1;`

`while (i<=n)`

`i=i*10;`

语句1： 频度=1

语句2： 频度 $\leq \lg(n)$

其时间复杂度为： $T(n)=1+\log n=O(\lg(n))$



1.3 算法及其评价

按数量级递增排列，常见的时间复杂度有：

常数阶 $O(1)$,对数阶 $O(\log_2 n)$,线性阶 $O(n)$,

线性对数阶 $O(n\log_2 n)$,平方阶 $O(n^2)$ ，立方阶 $O(n^3)$,...,

k 次方阶 $O(n^k)$,指数阶 $O(2^n)$ 。随着问题规模 n 的不断增大，上述时间复杂度不断增大，算法的执行效率越低。



1.3.2 空间复杂度

与时间复杂度类似，空间复杂度是指算法在计算机内执行时所占用的内存开销规模。但我们一般所讨论的是除正常占用内存开销外的额外空间。



本章作业

数据结构题集 P8:

1.8(2)(4)(6)(8)

1.9