数据结构与算法

- 课程编号：**CS32131**
- 授课学时：**40（7至16周，周4学时，2次）**
- 实验学时：**16（第10/12/14/16/17周，3学时/1次）**
- 课程分类：专业（技术）基础
- 先修课程：集合论与图论、高级语言程序设计
- 答疑地点：**格物楼203，每周1次**
- 课程资源：
  - **IP:10.160.3.21:8080?**
- 考核形式：
  - **笔试70%+作业10%+实验20%**
- 主讲教师：李秀坤
  - **联系方式：电话13796628867**
  - **email: lixiukun@hit.edu.cn**

数据结构与算法

第1章 绪论

数据结构与算法

教学目的:

(1)学会分析和研究计算机处理的数据对象的特性,掌握常用数据结构内在的逻辑关系、在机内的存储表示,掌握常用数据结构上的运算操作的动态性质和执行算法.

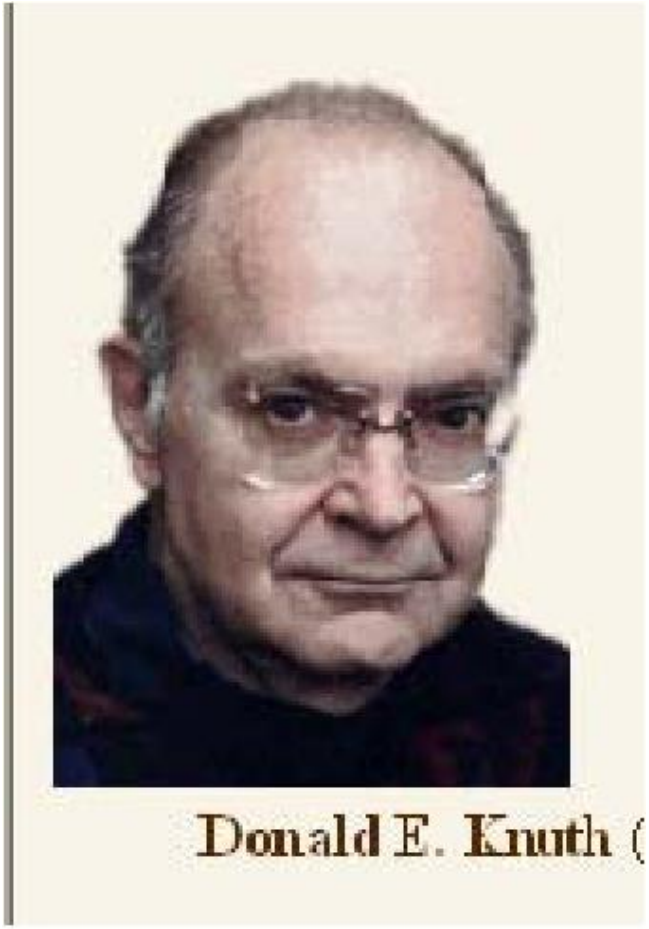
(2)能够为实际应用选择适当的数据结构、存储结构和相应算法;

(3)初步掌握算法性能的分析方法。

参考书目

- [美]Sartaj Sahni 著，汪诗林 孙晓东等译：《数据结构、算法与应用》C++语言描述，机械工业出版社，2000年1月
- 高质量C++/C编程指南 http://man.chinaunix.net/develop/c&c++/c/c.htm#_Toc520634058

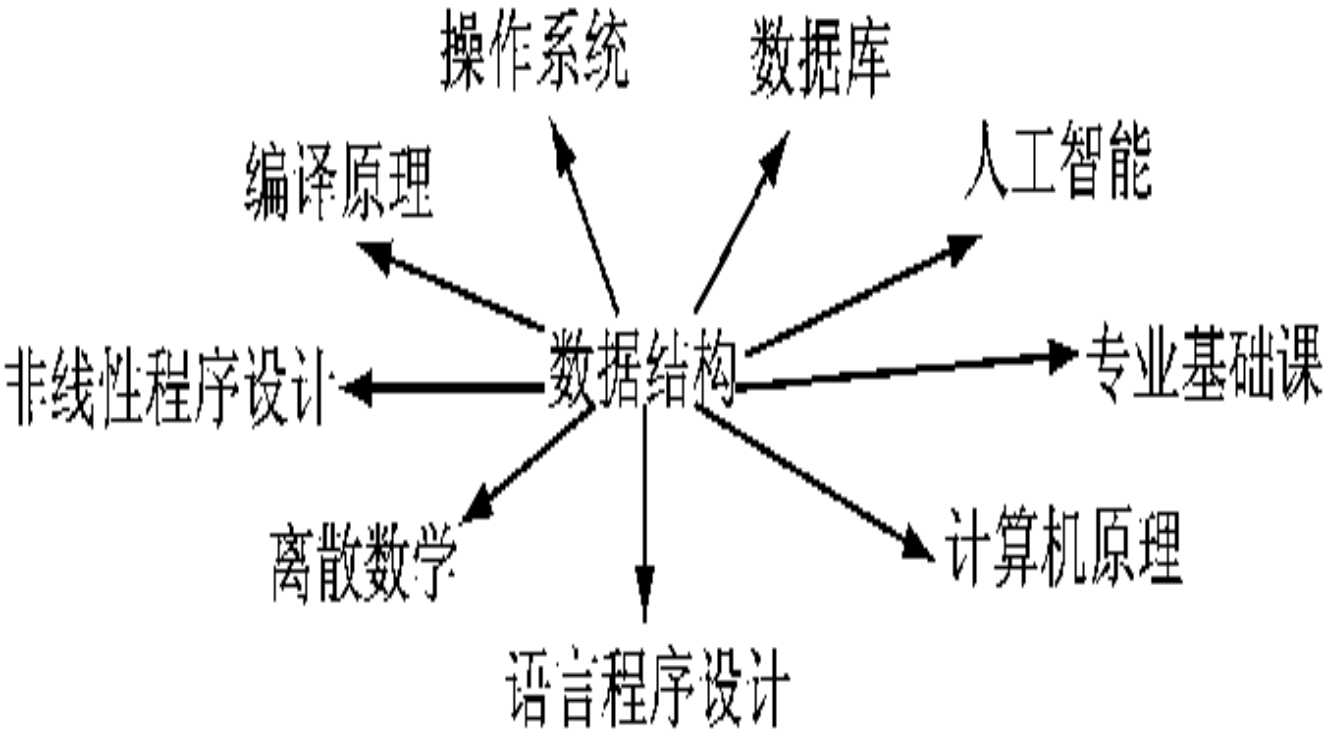
数据结构的创始人—Donald. E. Knuth

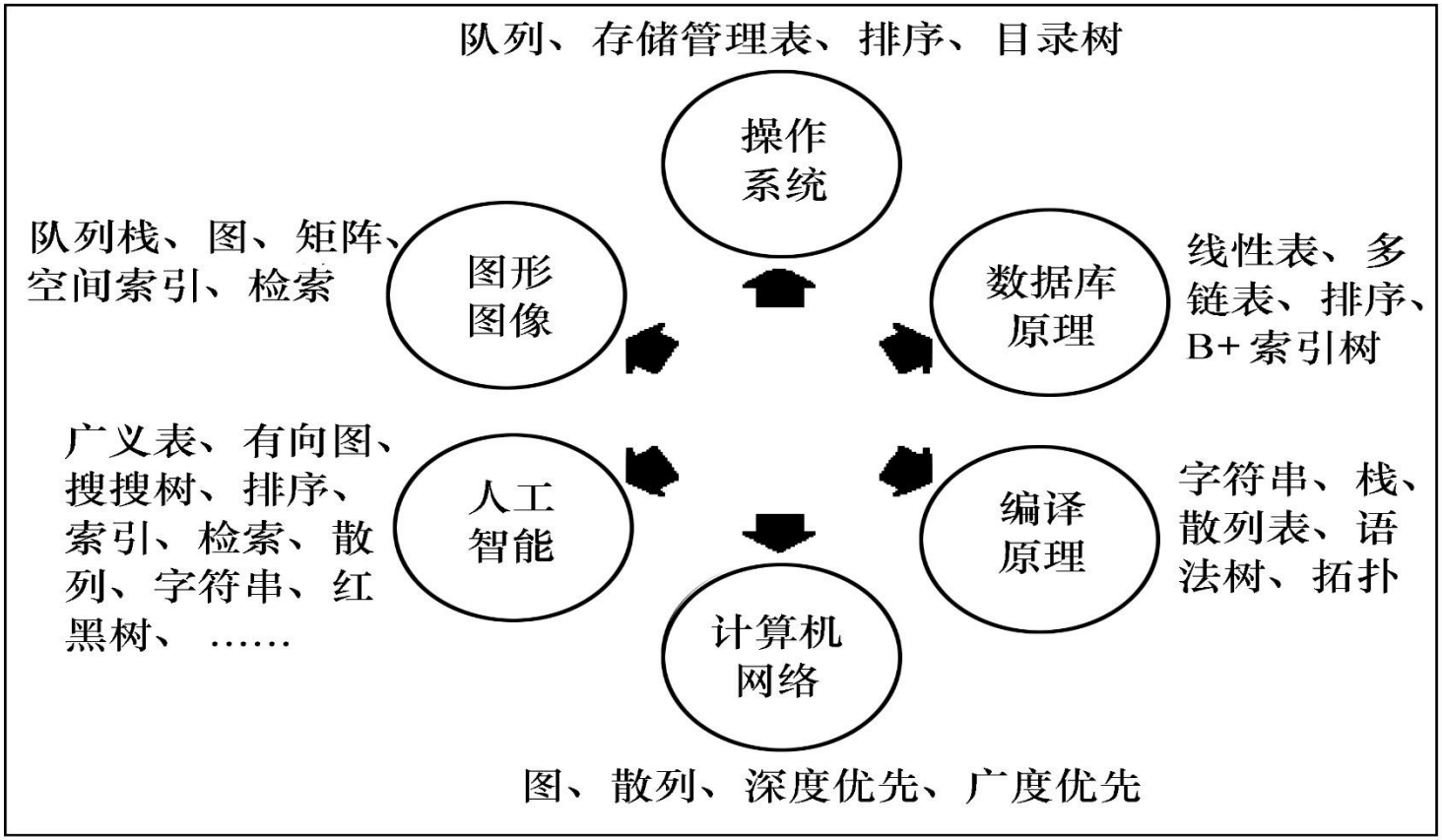


1938年出生，**25**岁毕业于加州理工学院数学系，博士毕业后留校任教，**28**岁任副教授。**30**岁时，加盟斯坦福大学计算机系，任教授。从**31**岁起，开始出版他的历史性经典巨著：

The Art of Computer Programming

他计划共写**7**卷，然而出版三卷之后，已震惊世界，使他获得计算机科学界的最高荣誉图灵奖，此时，他年仅**36**岁。


操作系统
数据库
编译原理
人工智能
非线性程序设计
数据结构
专业基础课
离散数学
计算机原理
语言程序设计


队列、存储管理表、排序、目录树
操作系统
图形图像
队列栈、图、矩阵、空间索引、检索
数据库原理
线性表、多链表、排序、B+索引树
人工智能
广义表、有向图、搜搜树、排序、索引、检索、散列、字符串、红黑树、……
编译原理
字符串、栈、散列表、语法树、拓扑
计算机网络
图、散列、深度优先、广度优先

大学遗憾排行榜

序号	遗憾之事	遗憾指数
1	没有把握好那些可以让自己变得更好的时间	★★★★★
2	没早点开始做职业规划	★★★★★
3	没有练就一项让自己立足于社会的本领	★★★★★
4	没有深入了解自己的专业	★★★★
5	没有主动参加任何一个社团组织	★★★★
6	没有常回家看看父母	★★★★
7	不喜欢自己的专业，也没有勇气转系	★★★
8	没有不为学分纯为兴趣旁听过任何一门课程	★★★
9	没有谈一场恋爱	★★★
10	没有毕业旅行	★★★
11	没有坚持运动，变成了颓废的胖子	★★★
12	没有听过图书馆的闭馆音乐	★★
13	没有一次青春的疯狂	★★
14	没有和室友推心置腹地谈过	★★
15	没有利用学生证半价去看展览、看风景	★★
16	没有走遍校园的每一个角落	★★

主要内容

- 1.1数据结构的定义
- 1.2数据结构的抽象形式
- 1.3算法定义
- 1.4算法性能分析与度量
- 本章小结


指针
指针
区块1
区块2
区块3
区块链中的内容
上个区块的hash值
区块2中的内容
上个区块的hash值
区块3中的内容


0
15
16
31
4位
版本
4位首部
长度
8位服务类型
(TOS)
16位总长度(字节数)
16位标识
3位
标志
13位片偏移
8位生存时间
(TTL)
8位协议
16位首部检验和
20字节
32位源IP地址
32位目的IP地址
选项(如果有)
数 据

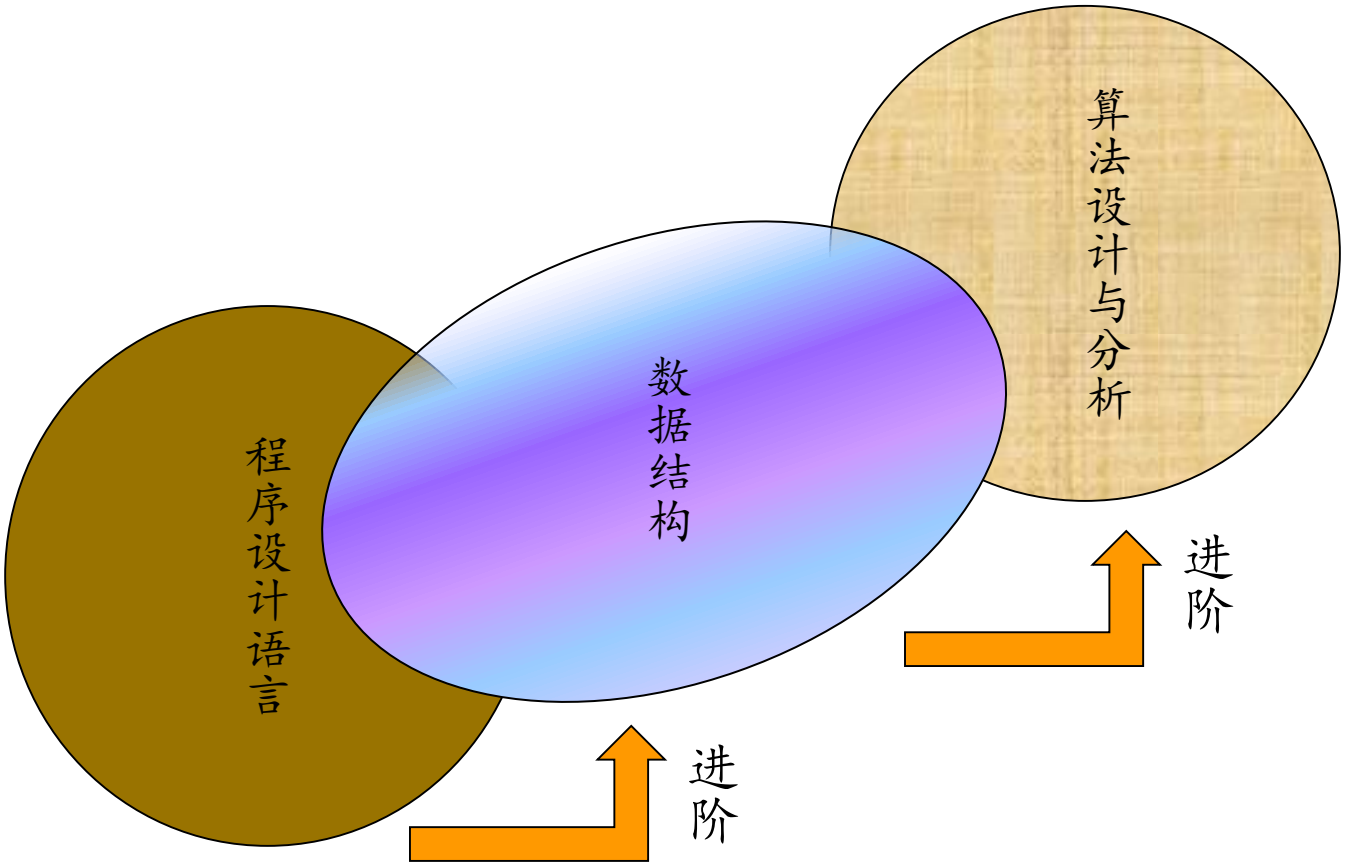
问题： 统计

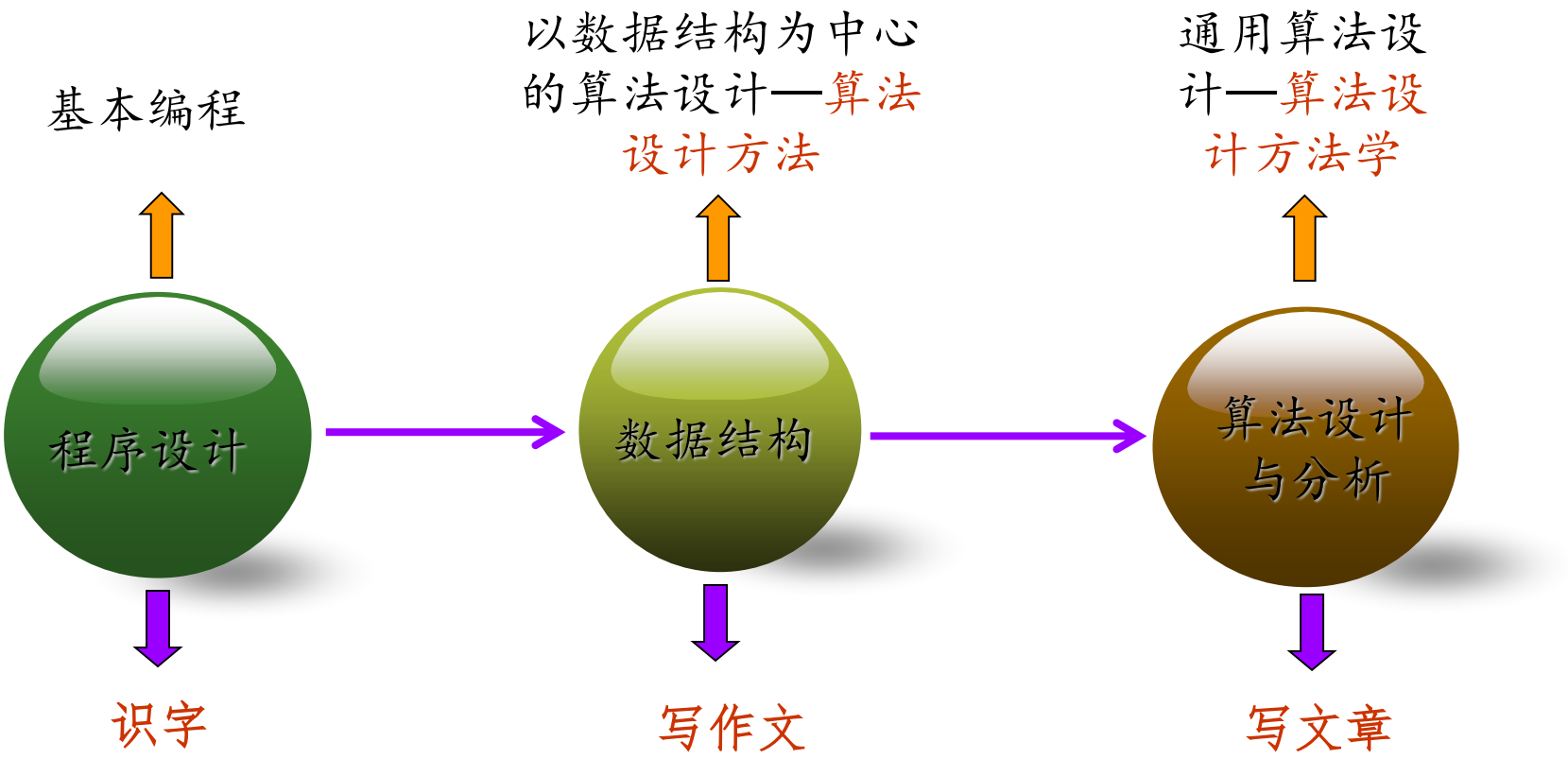
- 表1是5次C语言测试成绩，编写程序计算这5次测试成绩的总分。

```c
void main()
{
    int t1,t2,t3,t4,t5;/*各次成绩*/
    int sum;
    t1=98;t2=87;t3=65;t4=54;t5=76;
    sum=t1+t2+t3+t4+t5;
    printf("输出总分: %d\n",sum);
}
```

```c
void main()
{
    int t[5]={98,87,65,54,76};
    int sum=0;
    int i;
    for(i=0;i<5;i++)
    sum+=t[i];
    printf("输出总分: %d\n",sum);
}
```

从程序设计到算法设计的课程体系




基本编程
以数据结构为中心的算法设计—算法设计方法
通用算法设计—算法设计方法学
程序设计
数据结构
算法设计与分析
识字
写作文
写文章

数据结构和程序设计课程的侧重点

程序设计		数据结构
注重变量定义	⟷	注重数据组织
注重控制流程	⟷	注重数据处理方法
强调程序设计	⟷	强调“好程序”的设计

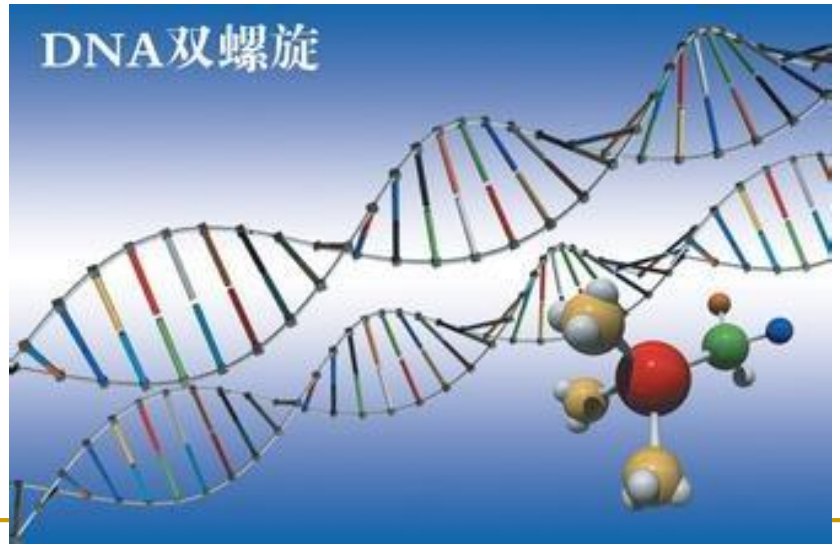
数据结构和算法设计与分析课程的侧重点

数据结构		算法设计与分析
掌握各种常用的数据结构	⟷	掌握各种常用的求解策略
↑		↑
基于数据结构的算法设计，如栈、队列、二叉树和图算法等。	⟷	基于求解策略的算法设计，如贪心法、分治法、回溯法、动态规划和分枝限界法等。

为什么学习数据结构课程：数据是有结构的

引子：万事万物都是有结构的。

❶ 微观世界—DNA结构



❷ 宏观世界—建筑物的结构

1.1 数据结构的定义

✓信息是客观世界在人脑中的反映

✓数据是信息的载体

怎样在计算机中存储和组织数据？

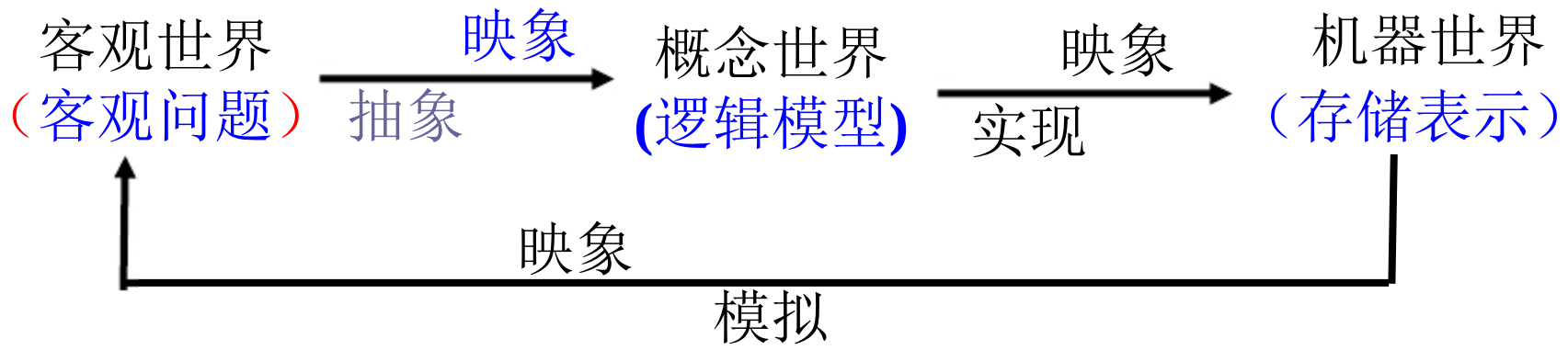
- 程序设计的实质是什么？
  - 数据表示：将数据存储在计算机中
  - 数据处理：处理数据，求解问题
  - 数据结构问题起源于程序设计

- 数据结构随着程序设计的发展而发展
  - 1. 无结构阶段：在简单数据上作复杂运算
  - 2. 结构化阶段：数据结构＋算法＝程序
  - 3. 面向对象阶段：（对象＋行为）＝程序

- 数据结构的发展并未终结……

■ 客观世界与计算机世界的关系

- ✓ 计算机科学是研究信息表示和信息处理的科学。
- ✓ 信息在计算机内是用数据表示的。
- ✓ 用计算机解决实际问题的实质可以用下图表示：



客观世界与计算机的关系

例1　学籍管理问题——表结构

完成什么功能?各表项之间是什么关系？

学号	姓名	性别	出生日期	政治面貌
0001	王晓东	男	**1990/09/02**	团员
0002	李明远	男	**1989/12/25**	党员
0003	张蔷薇	女	**1991/03/26**	团员
…	…	…	…	…

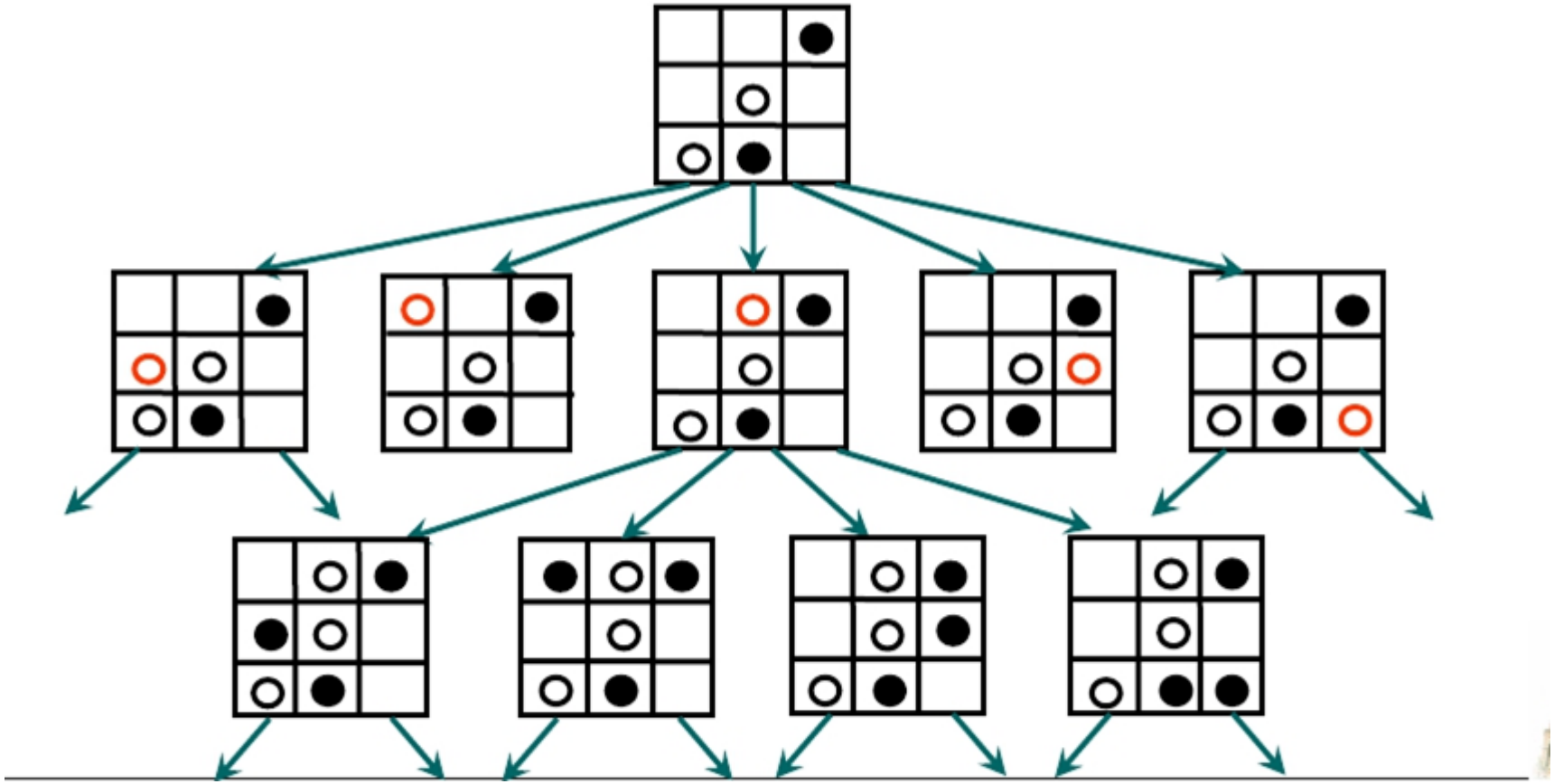
1997 年 5 月 11 日

计算机获得世界国际象棋冠军

（深蓝）　　（加里 卡斯帕罗夫）

例2 人机对弈问题——树结构

如何实现对弈?各格局之间是什么关系？



例3：在任意6人的集合上，有3个人以前彼此认识，或者有3个人以前彼此不认识。这二者必居其一。

1. 如何证明？
2. 人与人的关系如何表示？

逻辑推理问题：皇帝不是穷人，在守财奴中也有穷人，所以有一些————并不是————

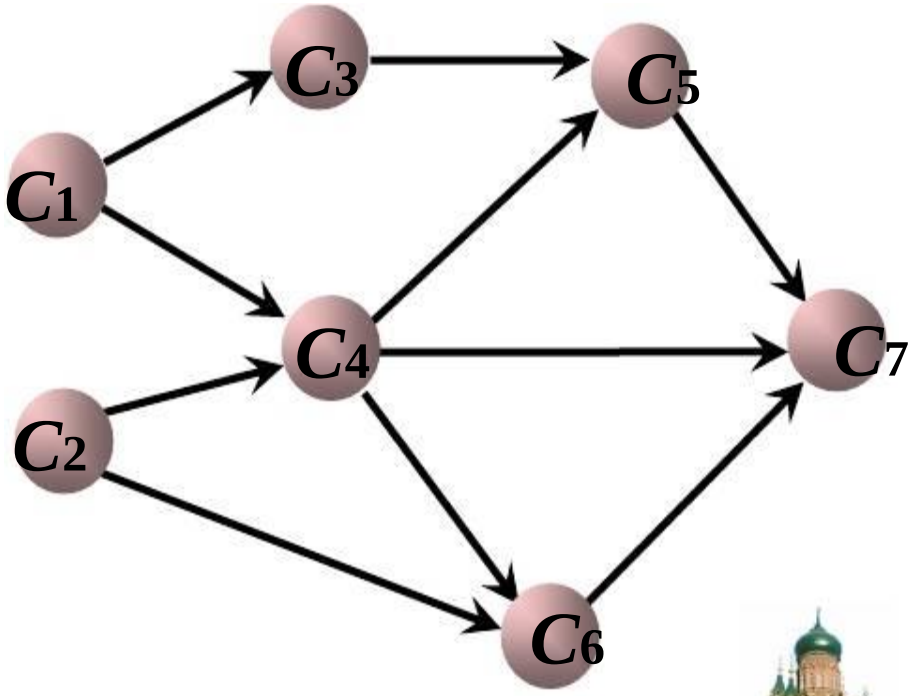
A. 皇帝 皇帝　　B.皇帝 守财奴

C. 守财奴 守财奴　D.守财奴　皇帝

例4 教学计划编排问题——图结构

如何表示课程之间的先修关系？

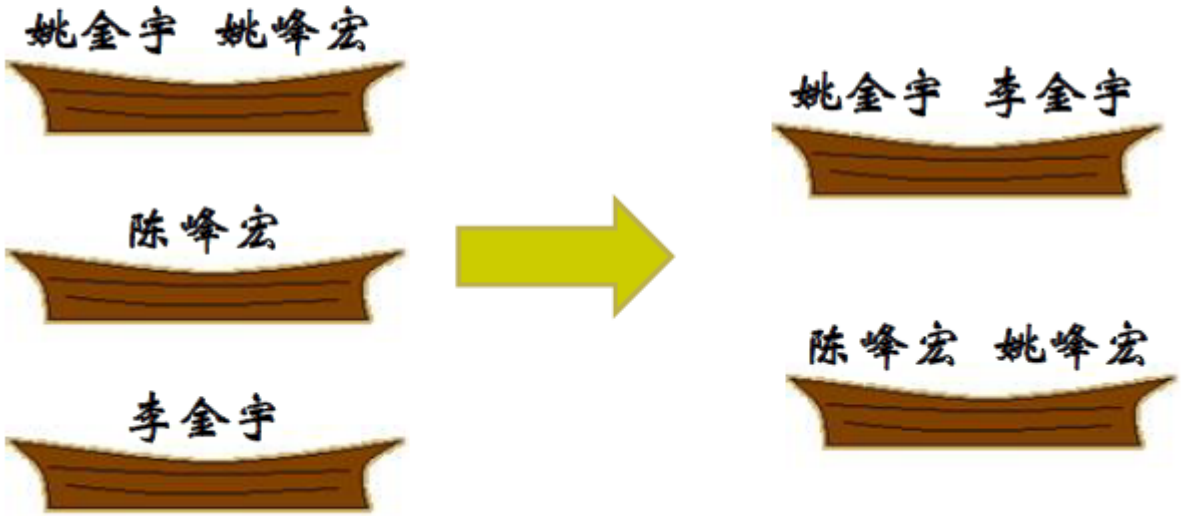
编号	课程名称	先修课
C1	高等数学	无
C2	计算机导论	无
C3	离散数学	C1
C4	程序设计	C1,C2
C5	数据结构	C3,C4
C6	计算机原理	C2，C4
C7	数据库原理	C4,C5,C6



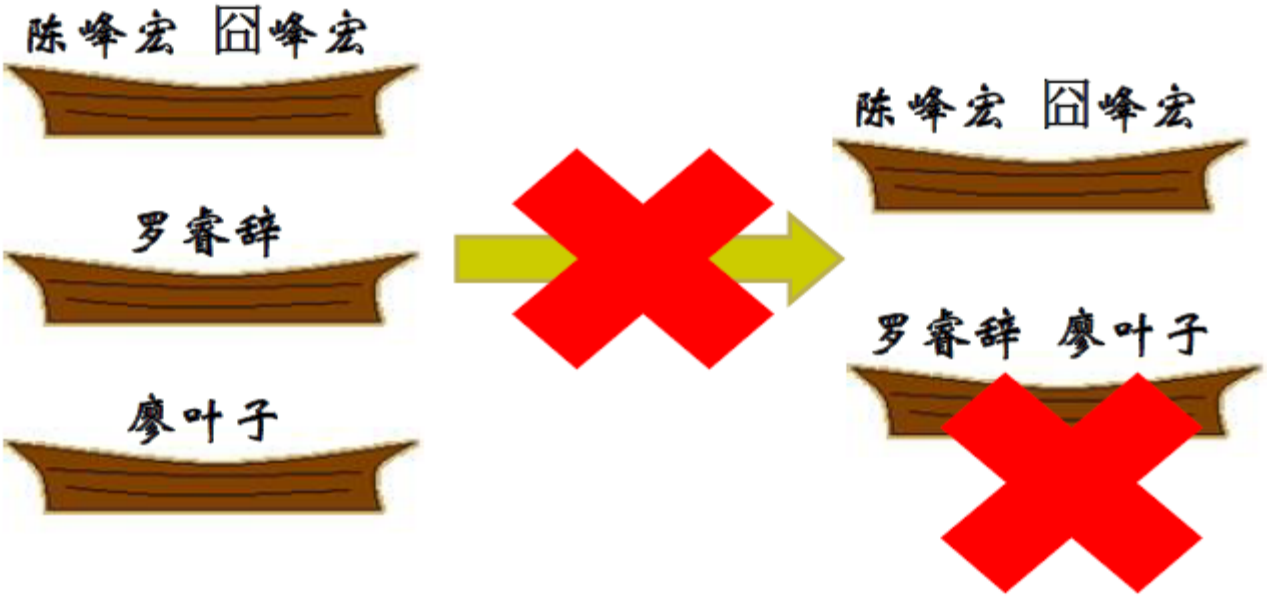
例5 坐船问题（改编自湖南省信息学省队选拔赛试题）

- ✓ 某高校有n个学生去公园划船；
- ✓ 一只船最多坐2个人；
- ✓ 出于娱乐目的，大家决定同船的2个人 要么同姓要么同名；
- ✓ 每个人都必须上船，且一人不能上多只船；
- ✓ 问最少需要几只？

- 姚金宇，李金宇，姚峰宏，陈峰宏

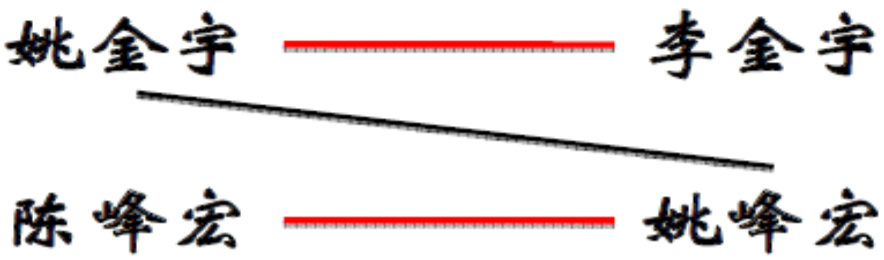


- 陈峰宏，囧峰宏，罗睿辞，廖叶子



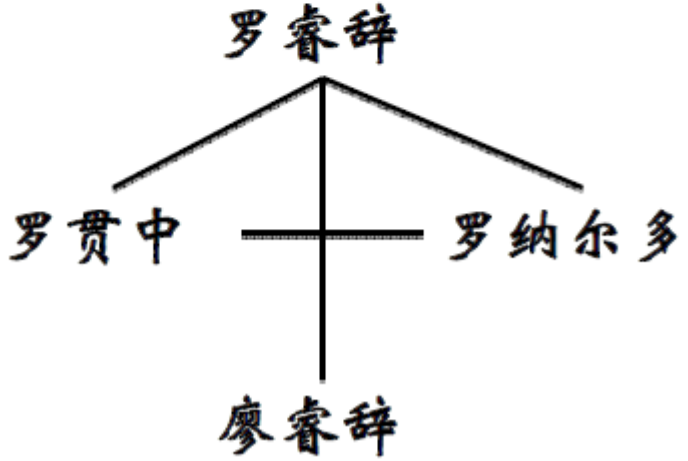
图论问题

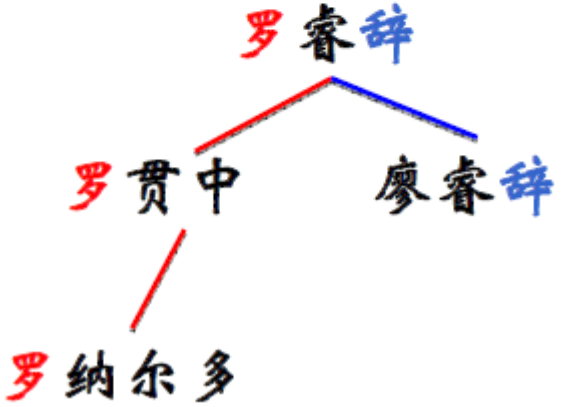
- 一条边代表一种坐船的搭配方式
- 用最少的边覆盖图中的点
  - 一般图的最小边覆盖问题
- 一般图最大匹配问题，算法复杂，实现麻烦。



图转换成二叉树

- 树中一个结点的左孩子跟其同姓；
- 一个结点的右孩子跟其同名
- 对于原图中的每一个连通分量，一定可以转换成一棵二叉树





基本概念：

数据 (Data):

- 数据是信息的载体，是描述客观事物的数、 字符、以及所有能输入到计算机中，被计算机程序识别和处理的符号的集合。

  - ✓数值性数据
  - ✓非数值性数据

数据元素 (Data Element):

- ✓数据的基本单位，在计算机程序中常作为 一个整体进行考虑和处理。
- ✓有时一个数据元素可以由若干数据项(Data Item)组成。数据项是具有独立含义的最小标识单位。
- ✓数据元素又称为元素、结点、记录。

数据对象 (Data Object):

- 数据的子集，具有相同性质的数据成员（数据元素）的集合。
  - 整数数据对象　$N=\{0, \pm1, \pm2, \ldots\}$
  - 学生数据对象
- 数据对象中所有成员之间存在某种关系，如学生按学号的排序；按性别的分类等。
- 数据成员及其之间关系，是数据结构研究的主要内容。


学号
姓名
语文
数学
C语言
6201001
张三
85
54
92
6201002
李四
92
84
64
6201003
王五
87
74
73
6201004
...
一个数据元素
一个数据项
整个表记录的是学生成绩数据，单个学生的成绩是其中的一个数据元素。

什么是数据结构

数据以及数据之间的相互关系，即计算机中存储和组织数据的形式。

通常可以用一个二元组<D，R>来表示。

或写成DS=<D，R>。

其中D是数据集合（数据对象），R是D中数据元素之间所存在的关系的有限集合。

数据结构（技术）就是根据各种不同的数据集合和数据元素之间的关系，研究如何表示、存储和操作（查找、插入、删除、修改、排序）这些数据的技术。

DS第一个重要部分：

数据的逻辑结构 —人为定义的，用户可以看到

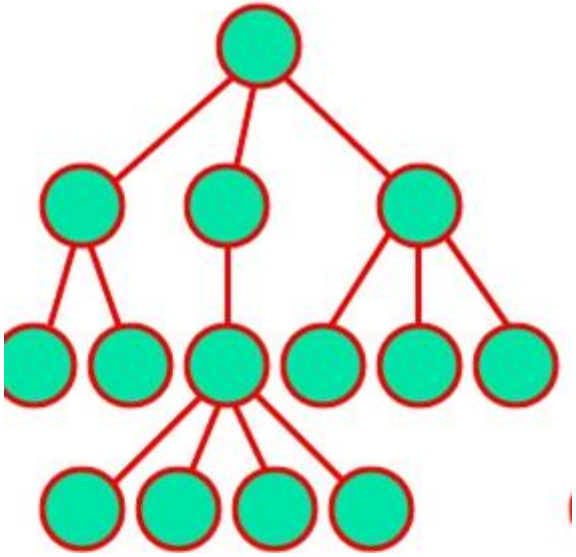
- ✓数据的逻辑结构从逻辑关系上描述数据，与数据的存储无关；
- ✓数据的逻辑结构可以看作是从具体问题抽象出来的数据模型；
- ✓数据的逻辑结构与数据元素的相对（存储）位置无关。

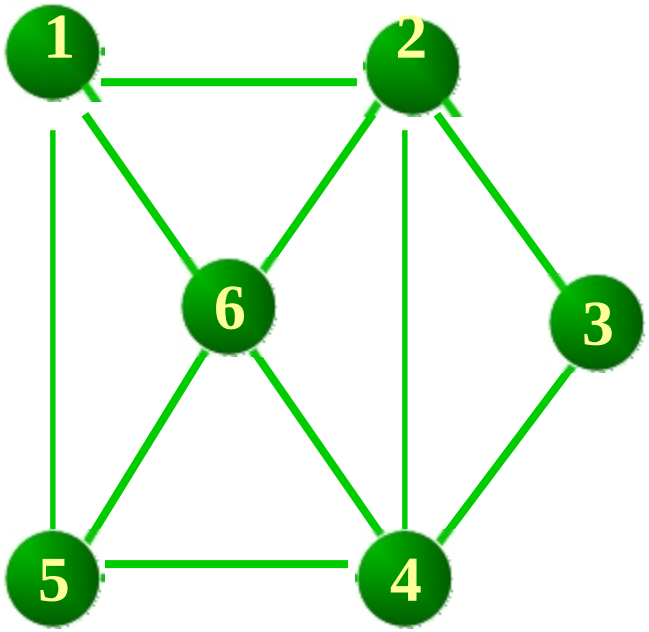
数据的逻辑结构分类

- 集合
- 线性结构
- 非线性结构
  - 树
  - 图（或网络）

线性结构：

树形结构：





图结构

线性结构

- 关系r 是一种线性关系，或称为“前后关系”，有时也称为“大小关系”。关系 r 是有向的，且满足全序性和单索性等约束条件
  - 全序性是指，线性结构的全部结点两两皆可以比较前后（关系 r）
  - 单索性是指，每一个结点 x 都存在唯一的一个直接后继结点 y。如果其他结点 z 在 y 之前，则这个 z 也一定在 x 之前，不会在x，y之间

e.g. linearity=(K,R)
K={1,2,3,4,5,6,7,8,9,10}
R={<5,1>,<1,3>,<3,8>,<8,2>,<2,7>,<7,4>,<4,6>,<6,9>,<9,10>}

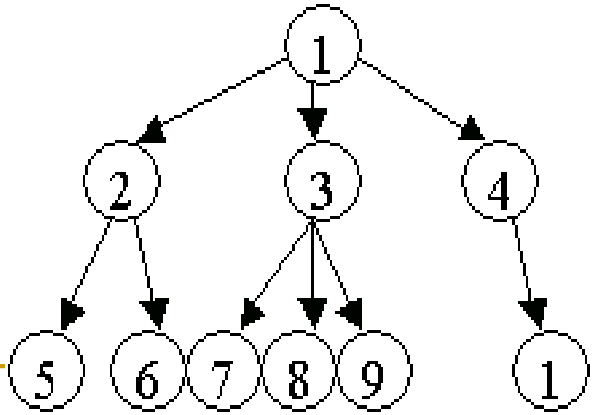
对应的图形：
⑤→①→③→⑧→②→⑦→④→⑥→⑨→⑩

特点：数据元素之间是1对1（1：1）联系。

树型结构

- 简称树结构，或层次结构。其关系 r 称为层次关系，或称“父子关系”、“上下级关系”等
- 每一个结点可以有多于一个的“直接下级”，但是它只能有唯一的“直接上级”。
  - 树型结构的最高层次的结点称为根（root）结点。只有它没有父结点
- 树型结构存在着很多变种，如二叉树结构，堆结构等，都有各自独特的有效应用

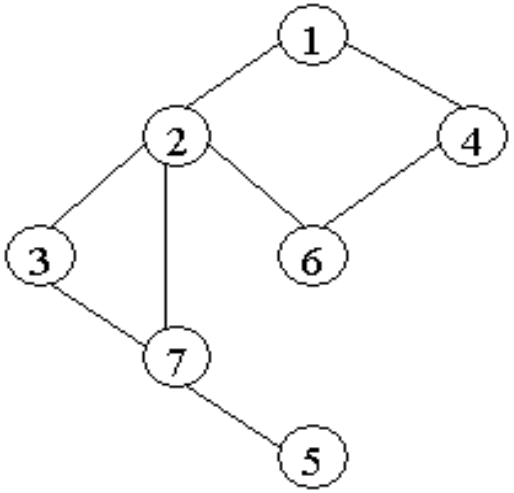
e.g.tree=(K,R)
K={1,2,3,4,5,6,7,8,9,10}
R={1,2>,<1,3>,<1,4>,<2,5>,<2,6>,<3,7>,<3,8>,<3,9>,<4,10>}
对应的图形:



图结构

- 有时称为结点互联的网络结构，因特网的网页链接关系就是一个非常复杂的图结构
- 对于图结构的关系 r 没有加任何约束。这样也就无法象线性结构及树结构那样，利用关系 r 的约束来设计图结构的存储结构
- 在日常应用中图结构往往只是层次结构的一种扩展——允许结点具有多个“直接上级结点”，关系 r 表现为树型结构约束的放松
- 从数学上看，树型结构和图结构的基本区别就是“每个结点是否仅仅从属一个直接上级”。而线性结构和树型结构的基本区别是“每个结点是否仅仅有一个直接后继”

e.g.graph=(K,R)
K={1,2,3,4,5,6,7}
R={r}
R={<1,2>,<2,1>,<1,4>,<4,1>,<2,3>,<3,2>,<2,6>,<6,2>,<2,7
<4,6>,<6,4>,<5,7>,<7,5>}



对应的图形:

特点：数据元素之间是M对N（M:N）(M≥0,N≥0)

树结构和图（网状）结构统称为非线性结构。

DS第二个重要部分：

数据的存储结构-计算机如何存储（结点及结点关系）

- ✓数据的存储结构是逻辑结构用计算机语言的实现；
- ✓数据的存储结构依赖于计算机语言；
- ✓计算机处理问题时是一条一条的，不能像人一样整体处理，根据存储结构找到下一条数据。

数据的存储结构

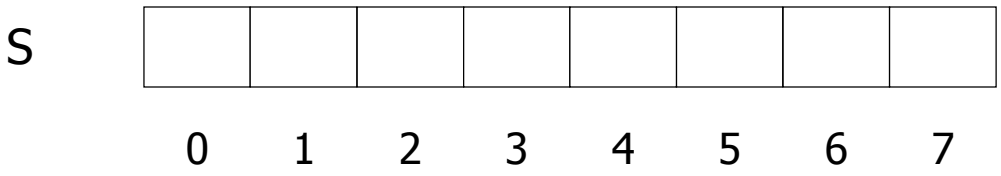
- 数据的存储结构是建立一种映射，对于数据逻辑结构（K，r），其中r∈R
  - 对结点集合 K 建立一个从K到存储器M的单元的映射：K→M，对于每一个结点 j∈K 都对应一个唯一的连续存储区域c ∈ M
  - 每一个关系元组（j_1，j_2）∈r（其中j_1，j_2∈K是结点），亦即j_1，j_2的逻辑后继关系应映射为存储单元的地址之间的顺序关系（或指针的地址指向关系）
- 基本存储映射方法：顺序、非顺序（链接）

其中顺序可以延伸到索引，索引可以扩展为散列

- 顺序存储表示：逻辑相邻则物理相邻
- 链接存储表示：逻辑相邻未必物理相邻
- 索引存储表示：为数据建立索引表
- 散列存储表示：根据数据的关键码用散列函数计算出该数据的存储地址

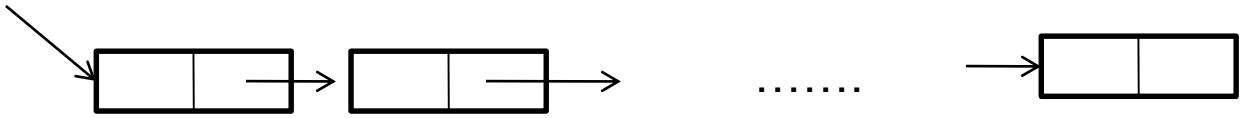
顺序方法

- 用一块连续的存储区域存储数据称为顺序存储
- 顺序存储把一组结点存储在按地址相邻的顺序存储单元里，结点间的逻辑后继关系用存储单元的自然顺序关系来表达
- 顺序存储法为使用整数编码来访问数据结点提供了便利



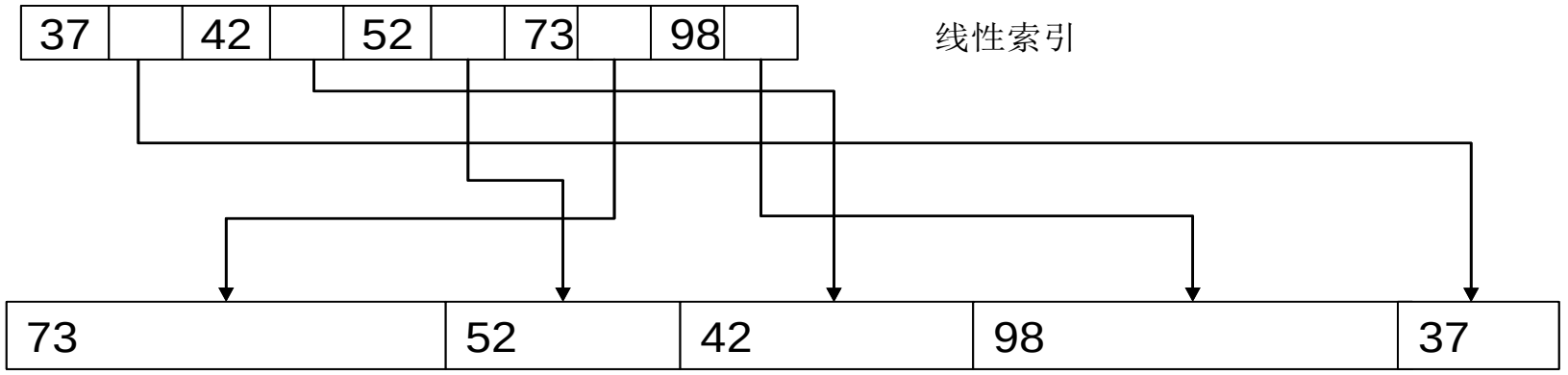
链接方法

- 利用指针，在结点的存储结构中附加指针字段称为链接法。两个结点的逻辑后继关系用指针来表达
- 任意的逻辑关系 r，均可使用这种指针地址来表达。一般的做法是将数据结点分为两部分:
  - 一部分存放结点本身的数据，称为数据字段
  - 一部分存放指针，称指针字段，链接到某个后继结点，指向它的存储单元的开始地址。多个相关结点的依次链接就会形成链索



索引方法

- 顺序存储法的一种推广，也使用整数编码来访问数据结点位置
- 建造一个由整数域 Z 映射到存储地址域D 的函数　Y：Z→D，把结点的整数索引值 z∈Z 映射到结点的存储地址 d∈D 。Y称为索引函数



注：一般而言，索引函数并不象数组那样，是简单的线性函数。当数据结点长度不等的情况下，索引函数就无法用线性表达式给出

- 为了构造任意的索引函数，可为索引函数提供附加的存储空间，称为索引表S
- 索引表中每一元素是指向数据结点的指针。因为索引表 S 由等长元素（指针）组成，故可进行线性的索引计算：

  始址(元素S[i]) ＝ 始址(元素S[0]) ＋ i •（指针尺寸）

  通过上述公式，由索引号 i 可以计算出索引表中的单元S[i]的始址，再通过读出S[i]元素的内容（指针），访问真正需要访问的数据结点

散列方法

- 索引方法的一种延伸和扩展
- 利用一种称为散列函数(hash functions)的机制来进行索引值的计算，然后通过索引表求出结点的指针地址
- 散列函数是将字符串 s（ 或关键码）映射到非负整数 z 的一类函数 h: S → Z ，

  对任意的 s∈ S，散列函数 h(s)=z，z∈ Z

实例：银行帐号共 100000 个 如图所示，组成一个顺序存储的结构，存于计算机之中。插入新帐号 75怎样进行呢？

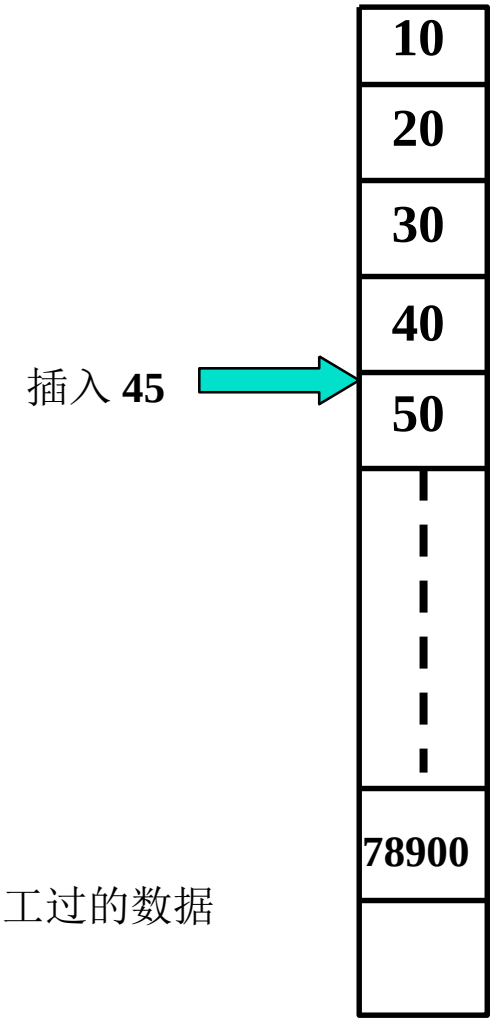
插入新帐号 75：1、查找位置

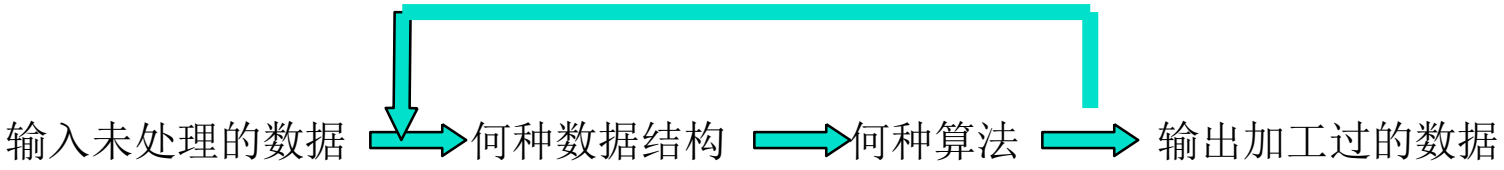
2、移表

3、插入合适位置

移动一个结点，需 100us, 移动 100000 个结点需 100us ×100000 ＝ 10 秒。每天处理 10000 个帐号，需 30 小时，无法接受。

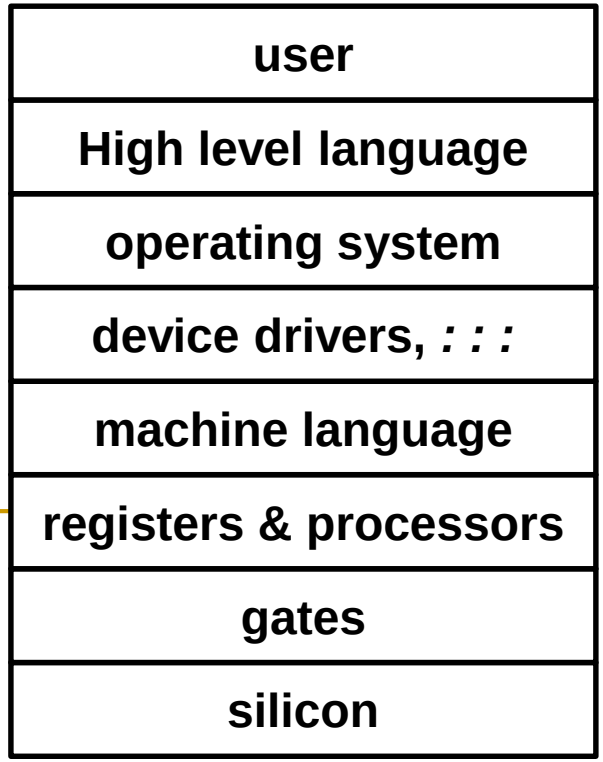
如何快速地进行插入? 节省访问外存的时间，是一个很重要的问题。





1.2 数据结构的抽象形式

- 计算机科学本身就是抽象的科学— 为问题建立适当的模型并设计相应的技术解决它
  - 相对于物理学
- 计算机本身的一些抽象
- 抽象的本质？
  - 简化
  - 忽略非本质的部分



抽象数据类型

- 模块化的思想的发展，为模块的划分提供了理论依据，简称ADT (Abstract Data Type)
- 目的
  - 隐藏运算实现的细节和内部数据结构
  - 提高复用的力度和粒度

数据类型： 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。

- C语言中的数据类型

char　int　float　double　void　***

字符型　整型　浮点型　双精度型　无值　地址

int取值范围[-32768, 32767];

每个数据类型对应一组操作

int(6/4)=1　(float)(6.0/4.0)=1.5

- 数据类型由
  - 基本数据类型 或
  - 构造数据类型组成

■构造数据类型由不同成分类型构成。

■基本数据类型可以看作是计算机中已实现的数据结构。

■数据类型就是数据结构，不过是从编程者的角度来使用。

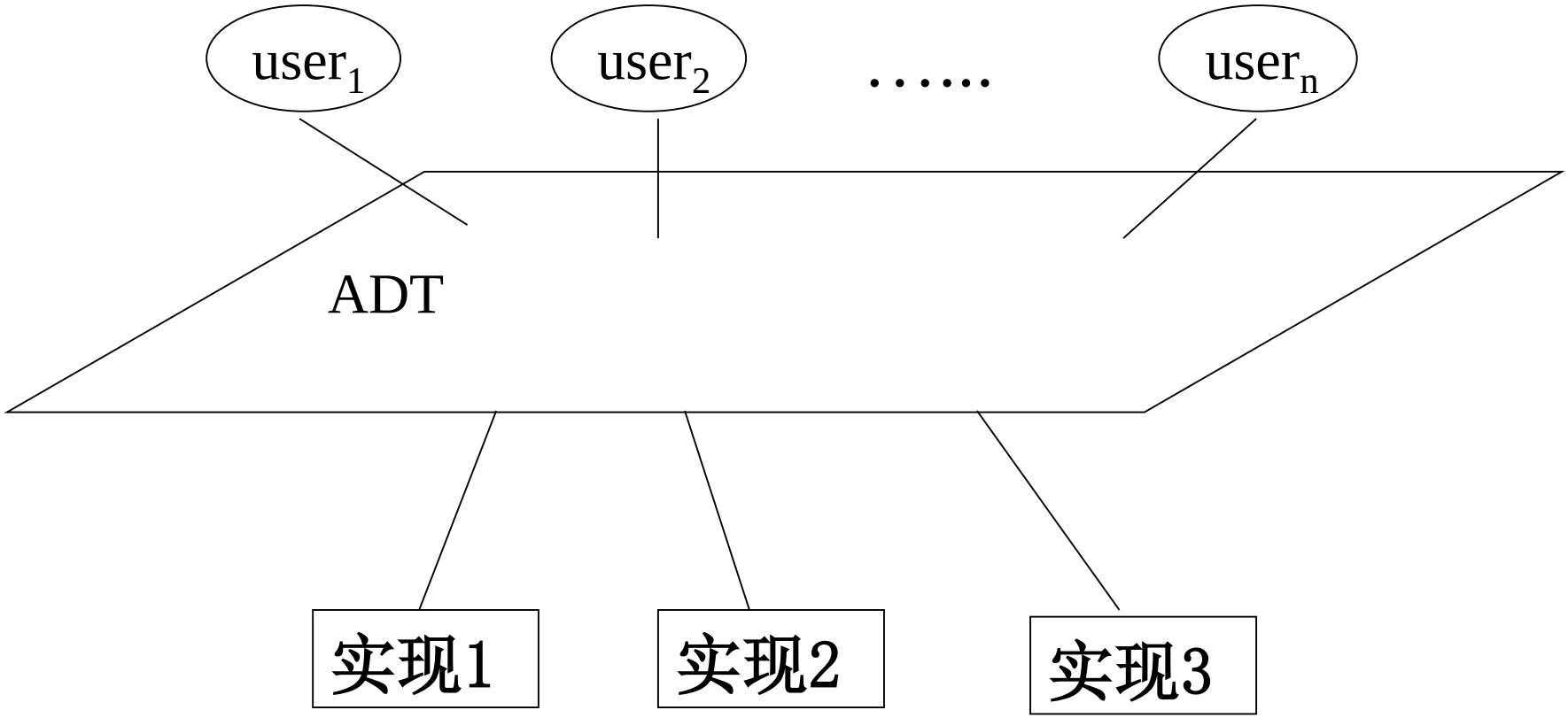
抽象数据类型ADT (Abstract Data Type)

✓由用户定义，用以表示应用问题的数据模型。

✓**ADT**：由基本的数据类型组成，并包括一组相关的服务（或称操作）。

✓信息隐蔽和数据封装，使用与实现相分离。

抽象数据类型



抽象数据类型

- 用数学方法定义对象集合和运算集合，仅通过运算的性质刻画数据对象，而独立于计算机中可能的表示方法
  - 可看作是定义了一组操作的一个抽象模型
    - 例如，集合与集合的并、交、差运算就可定义为一个的抽象数据类型
  - 一个抽象数据类型要包括哪些操作，这一点由设计者根据需要确定
    - 例如，对于集合，如果需要，也可以把判别一个集合是否为空集或两个集合是否相等作为集合上的操作

1.3 算法定义

- 对特定问题求解过程的描述，是指令的有限序列，也即，为解决某一特定问题而采取的具体有限的操作步骤。
- 程序是算法的一种实现，计算机按照程序逐步执行算法，实现对问题的求解。

算法Algorithm=程序Program？

- 算法是有穷性： 算法应在执行有穷步后结束。

- 程序可能持续运行，直到系统退出，
  例如操作系统wait函数。
- 算法是面向问题的。
- 程序是算法的具体语言实现。

- 计算：1+2+3+…+100?

算法的特性

- 输入Input 有0个或多个输入；
- 输出Output 有一个或多个输出（处理结果）；
- 确定性 每步定义都是确切无歧义的；
- 有穷性 算法应在执行有穷步后结束；
- 有效性 每一条运算应足够基本。

算法分类

- 算法设计与算法分析是计算机科学的核心问题
- 常用的设计方法
  - 穷举法　(百钱买百鸡)
  - 贪心法　(Huffman树、Prim等)
  - 递归法, 分治法(二分检索、快速排序等)
  - 回溯法(树、图等的深度优先搜索）
  - 动态规划法(最佳二叉排序树)
  - α-β裁剪和分枝界限法
  - 并行算法

- 评价算法效率的方法
  - 事后统计：将算法实现，测算其时间和空间开销。
  - 优点：准确的实测值
  - 缺点：
    - 编写程序实现算法将花费较多的时间和精力；
    - 所得实验结果依赖于计算机的软、硬件等环境因素。
  - 事前分析：对算法所消耗资源的一种估算方法。
- 算法分析：对算法所需要的计算机资源——时间和空间进行估算。
  - 时间复杂性（**Time Complexity**）
  - 空间复杂性（**Space Complexity**）

1.4 算法的执行效率及其度量

需要明确：

- 如何表达一个算法的复杂性
- 怎样计算一个算法的复杂性

算法分析----时间复杂度分析

算法的执行时间＝每条语句执行时间之和

每条语句执行次数之和

基本语句的执行次数

执行次数×执行一次的时间

单位时间

指令系统、编译的代码质量

例：
```
for (i=1; i<=n; i++)
    for (j=1; j<=n; j++)
        x++;
```

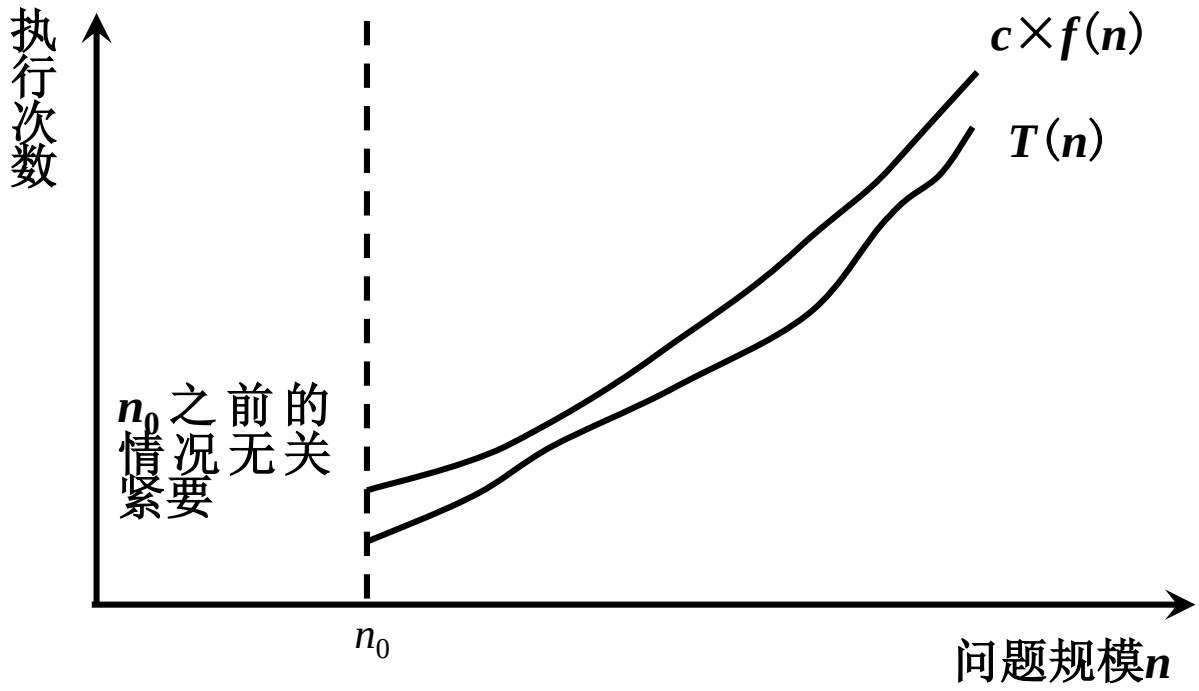
- 算法分析----算法的时间复杂度（性）
  - 算法的执行时间，是基本（操作）语句重复执行的次数，它是问题规模的一个函数。我们把这个函数的渐近阶称为该算法的时间复杂度。
  - 问题规模：输入量的多少。
  - 基本语句：是执行次数与整个算法的执行次数成正比的操作指令。

```
例：for (i=1; i<=n; i++)
        for (j=1; j<=n; j++)
            x++;
```

  - 问题规模：**n**
  - 基本语句：**x++**
  - 时间复杂性：**O（n^2）**

算法分析----大O符号

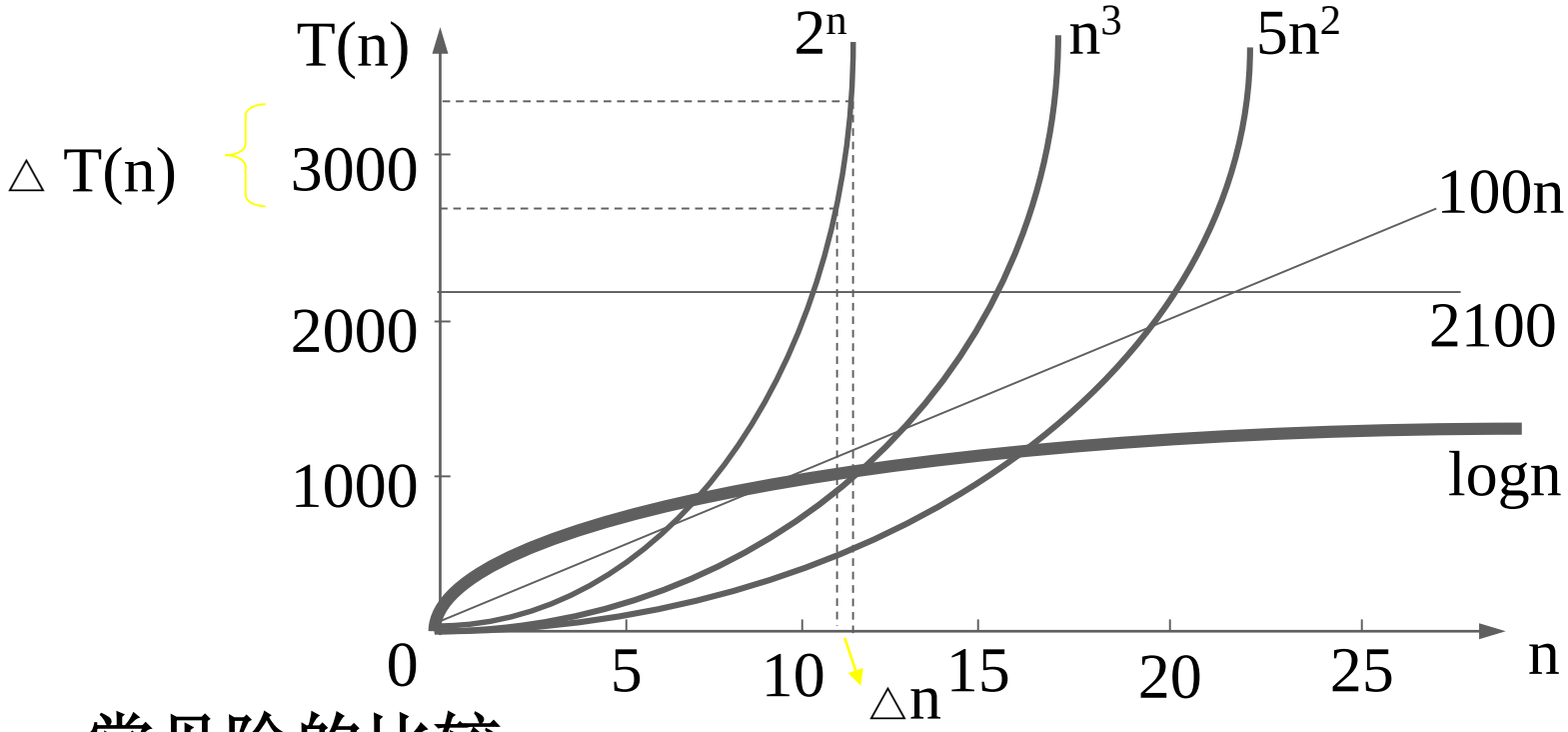
- 定义　若存在两个正的常数c和n_0，对于任意$n \geqslant n_0$，都有$T(n) \leqslant c \times f(n)$，则称$T(n)=O(f(n))$



当问题规模充分大时在渐近意义下的阶

算法分析----常见的时间复杂度

程序运行时间比较　T（n）=O（f（n））



- 常见阶的比较：
- $O(1)<O(\log_2 n)<O(n)<O(n\log_2 n)<O(n^2)<O(n^3)<\ldots<O(2^n)<O(n!)$

- 算法分析-----时间复杂性分析的基本方法
  - 时间复杂性的运算法则

设$T_1(n)=O(f(n))$，$T_2(n)=O(g(n))$，则

  - ①加法规则:$T_1(n)+T_2(n)=O(\max\{f(n), g(n)\})$
  - ②乘法规则:$T_1(n)*T_2(n)=O(f(n)\cdot g(n))$

  - 时间复杂性的分析方法
    - 首先求出程序中各语句、各模块的运行时间,
    - 再求整个程序的运行时间。
    - 各种语句和模块分析应遵循的规则是:

- 算法分析---各种语句和模块分析应遵循的规则
  - (1)赋值语句或读/写语句:
    - 运行时间通常取O(1) .有函数调用的除外，此时要考虑函数的执行时间。
  - (2)语句序列:
    - 运行时间由加法规则确定，即该序列中耗时最多的语句的运行时间。
  - (3)分支语句:
    - 运行时间由条件测试（通常为O(1) ）加上分支中运行时间最长的语句的运行时间

- 算法分析-----各种语句和模块分析应遵循的规则
  - (4)循环语句:
    - 运行时间是对输入数据重复执行n次循环体所耗时间的总和每次重复所耗时间包括两部分：一是循环体本身的运行时间；二是计算循环参数、测试循环终止条件和跳回循环头所耗时间。后一部分通常为O(1)。
    - 通常，将常数因子忽略不计，可以认为上述时间是循环重复次数n和m的乘积，其中m是n次执行循环体当中时间消耗最多的那一次的运行时间(乘法规则)
    - 当遇到多重循环时，要由内层循环向外层逐层分析。因此，当分析外层循环的运行时间是，内层循环的运行时间应该是已知的。此时，可以把内层循环看成是外层循环的循环体的一部分。

- 算法分析-----各种语句和模块分析应遵循的规则
  - (5)函数调用语句:
    - ①若程序中只有非递归调用，则从没有函数调用的被调函数开始，计算所有这种函数的运行时间。然后考虑有函数调用的任意一个函数P，在P调用的全部函数的运行时间都计算完之后，即可开始计算P的运行时间。
    - ②若程序中有递归调用，则令每个递归函数对应于一个未知的时间开销函数T(n)，其中n是该函数参数的大小，之后列出关于T的递归方程并求解之。

- 算法分析--例:分析下述“气泡”排序程序的时间复杂性。

```
    void BubbleSort( int A[], int n )
    {     int i, j, temp;
(1)       for (i=0; i<n-1; i++)
(2)           for (j=n-1; j>=i+1; j--)
(3)               if (A[j-1]>A[j]) {
(4)                   temp=A[j-1];
(5)                   A[j-1]=A[j];
(6)                   A[j]=temp;
                  }
    }
```

$$O(\sum_{i=0}^{n-2}(n-i-1)) \le O(n(n-1)/2) = O(n^2)$$

- 算法分析---例:编写求n!的程序，并分析其时间复杂性。

●求n!的递归算法

```
long  fact ( int  n)
{   if ( n==0 ) || ( n ==1 )
        return( 1 );
    else
        return( n * fact(n – 1));
}
```

●时间复杂性的递归方程

$$T(n)=\begin{cases} C & \text{当 } n=0,\ n=1 \\ G+T(n-1) & \text{当 } n>1 \end{cases}$$

●解递归方程：

$$\begin{aligned} T(n) &= G+T(n-1) \\ T(n-1) &= G+T(n-2) \\ T(n-2) &= G+T(n-3) \\ &\cdots\cdots \\ T(2) &= G+T(1) \\ +\ T(1) &= C \end{aligned}$$

$$T(n)=G(n-1)+C$$

⇩

取 $f(n)=n$

$\therefore T(n)=O(f(n))$

$=O(n)$

各时间量级在现实中对应的物体速度

米/秒	英制度量衡	现实世界的例子
10^{-11}	1.2 英寸/世纪	钟乳石的生长速度
10^{-10}	1.2英寸/十年	大陆板块的漂移速度
10^{-9}	1.2 英寸/年	指甲的生长速度
10^{-8}	1 英尺/年	头发的生长速度
10^{-7}	1英尺/月	杂草的生长速度
10^{-6}	3.4英寸/天	冰河的流动速度
10^{-5}	1.4英寸/时	表的分针转动速度
10^{-4}	1.2英尺/时	胃肠的蠕动速度
10^{-3}	2英寸/分	蜗牛的速度
10^{-2}	2英尺/分	蚂蚁的速度
10^{-1}	20英尺/分	巨龟的速度
1	2.2英里/时	人类的散步速度
10^{1}	22英里/时	人类疾跑速度
10^{2}	220英里/时	螺旋桨飞机的速度
10^{3}	37英里/分	喷气式飞机的速度
10^{4}	370英里/分	宇宙飞船的速度
10^{5}	3700英里/分	流星撞击地球的速度
10^{6}	620英里/秒	地球自转速度
10^{7}	6200英里/秒	卫星从洛杉矶到纽约的速度
10^{8}	62,000英里/秒	光速的三分之一

最差、最佳、平均情况

- 在进行算法增长率估计时，有些算法，即使问题规模相同，若输入数据不同，其时间复杂度也不同
  - 由于算法实际执行的操作往往依赖于分支条件的走向，而输入数据的取值又影响这些分支走向，因此很多算法都无法得出独立于输入数据的渐进估计
- 针对这一情况，提出了最差情况估计、平均情况估计、最佳情况估计等三种方法

示例

- 求一个数组的所有有序子数组中最长的一个：

```
for (i = 0, length=1; i<n-1; i++)  {
    for (j1 = j2 = k = i;  k < n-1 && a[k] < a[k+1]; k++, j2++);
      if (length < j2 - j1 - 1)
              length = j2 - j1 + 1;
}
```

譬如，在数组[1, 8, 1, 2, 5, 0, 11, 9]中，这个最长的有序子数组为[1，2，5]，长度为3。

时间代价和数组 a 中元素的实际取值状态很相关

示例：最佳

- 数组a的所有元素是以降序方式输入
  - 外层循环执行n-1次，
  - 每次内层循环只执行1次
  - 整个的时间开销为O(n)

min{ complexity(size(y)) | y∈ Input}

示例：最差

- 数组 **a** 的所有元素是以升序方式输入
  - 外层循环执行**n-1**次，
  - 对每个 **i**，内层循环需要执行 **(n - 1 - i)** 次
  - 整个的时间开销为**O(n^2)**

max {complexity(size(y)) | y∈ Input}

示例：平均

- 随机情况下，数组的元素是无序的，既非升序也非降序
- 计算平均情况的复杂度应该考虑算法的所有输入情况，确定针对每种输入情况算法所需的操作数目
  - 简单情况：每种输入出现的概率相同
  - 复杂情况：每种输入的出现概率并非相同，

$$C_{avg} = \sum_i p(input_i)steps(input_i) \qquad \sum_i P(input_i) = 1$$

需要了解算法的实际输入在所有可能的输入集合中的分布状况

最差、最佳、平均情况

- 对于时间开销，一般不注意算法的“最好估计”。特别是处理应急事件，计算机系统必须在规定的响应时间内做完紧急事件处理。这时，最坏估计是唯一的选择
- 对于多数算法而言，最坏情况和平均情况估计两者，它们的时间开销的公式虽然不同，但是往往只是常数因子大小的区别，或者常数项的大小区别。因此不会影响渐进分析的增长率函数估计

空间资源开销

- 对于空间开销，也可以实行类似的渐进分析方法
  - 很多算法使用的数据结构是静态的存储结构，即存储空间在算法执行过程中并不发生变化
  - 使用动态数据结构算法的存储空间是变化的，在算法运行过程中有时会有数量级的增大或缩小。对于这种情况，空间开销的分析和估计是十分必要的

时空资源的折中原理

- 同一个问题求解，一般会存在多种算法，这些算法在时空开销上的优劣往往表现出“时空折中”（trade-off）的性质
  - 即，为了改善一个算法的时间开销，往往以增大空间开销为代价，而设计出一个新算法来
  - 有时，为了缩小算法的空间开销，也可以牺牲计算机的运行时间，通过增大时间开销来换取存储空间的节省

数据结构的选择和评价

- 仔细分析所要解决的问题，特别是求解问题所涉及的数据类型和数据间逻辑关系
- 数据结构的初步设计往往在算法设计之先
- 注意数据结构的可扩展性。包括考虑当输入数据的规模发生改变时，数据结构是否能够适应 。同时，数据结构应该适应求解问题的演变和扩展
- 数据结构的设计和选择也要比较算法的时空开销的优劣

本章小结

- 数据结构的定义
- 数据结构的主要研究内容
- 抽象数据类型的概念
- 算法及其特点
- 算法的有效性度量

例. 按要求编写算法完成下列问题：

1. 假设红、白、蓝每种颜色的花盆至少有一盆，亦即 n >= 3，杂乱的排在一起，编写一个高效算法使花盆按红、蓝、白的顺序排序。
   假设每组输入为：一个整数N，表示有N个花盆。然后N个整数，每个整数为1（若为红花盆）、2（若为白花盆）或3（若为蓝花盆），每个整数用空格分开。输入N = 0时程序结束。

2. 约瑟夫环问题（Josephus problem）是如下一个游戏：编号为1到N的N个人围坐成一个圈，从第一个人开始，传递一个热马铃薯；在M次传递后，拥有热马铃薯的人离开圈子，圈缩小，游戏继续，离开的人后面的人捡起热马铃薯继续传递；最后留下的人获胜。设计一个高效算法，给出一个出列的序列。（通常M是一个常数）

习题

- **1.**设字符集为字符和数字的集合，字符的顺序为**A，B，C，…，Z，0，1，2，…，9，**请将下列字符串按字典顺序排列、存储： **PAB ， 5C，PABC ， CXY，CRSI,7,B899,B9,** 并分析可以采取的存储方案。
- **2.**有一个包括**100**个元素的数组，每个元素的值都是实数，请写出求最大元素的值及其位置的算法，讨论它可能采取的存储结构。
- **3.** 在有**n** 个选手$P_1,P_2,P_3,\ldots,P_n$参加的单循环赛中，每对选手之间非胜即负。现要求求出一个选手序列：$P_1',P_2',P_3',\ldots,P_n'$,其满足 P_i'胜$P_{i+1}'(i=1,\ldots,n-1)$。

- 4. 参加比赛有n个学院，学院编号为1……n。比赛分成m个男子项目，和w个女子项目。项目编号为男子1……m，女子m+1……m+w。不同的项目取前五名或前三名积分；取前五名的积分分别为：7、5、3、2、1，前三名的积分分别为：5、3、2；哪些取前五名或前三名由学生自己设定。（m<=20,n<=20）

  功能要求：

  1)可以输入各个项目的前三名或前五名的成绩；

  2)能统计各学院总分，

  3)可以按学院编号或名称、学校总分、男女团体总分排序输出；

  4)可以按学院编号查询学院某个项目的情况；可以按项目编号查询取得前三或前五名的学院。

  5)数据存入文件并能随时查询

  6)规定：输入数据形式和范围：可以输入学院的名称，运动项目的名称

  输出形式：有中文提示，各学院分数为整形

  界面要求：有合理的提示，每个功能可以设立菜单，根据提示，可以完成相关的功能要求。

  存储结构：学生自己根据系统功能要求自己设计，但是要求运动会的相关数据要存储在数据文件中。

  测试数据：**1**、全部合法数据；**2**、整体非法数据；**3**、局部非法数据。进行程序测试，以保证程序的稳定。测试数据及测试结果请在上交的资料中写明；