数据结构

教材

主教材

王红梅，胡明，王涛. 数据结构 （C＋＋版）. 清华大学出版社.

参考教材

王红梅. 数据结构学习辅导与实验指导. 清华大学出版社.

严蔚敏. 数据结构. 清华大学出版社.

严蔚敏, 吴伟民. 数据结构题集. 清华大学出版社.

课程性质

- 数据结构是计算机专业的专业基础课

  公共基础课、专业基础课、专业方向课、专业选修课

- 在教学计划中的地位：核心、承上启下

  前导课：高等数学、离散数学、程序设计语言

  后续课：数据库、操作系统、编译原理……

- 属于武术中的“练功”科目

  “练武不练功，到头一场空”

- 考研

学习目标

- ✓掌握基本的数据结构

  工具箱→复用、修改、重组

- ✓培养算法设计能力、程序设计能力

  算法——程序的灵魂

  问题求解过程：问题→想法→算法→程序

  程序设计研究的层次：算法→方法学→语言→工具

- ✓培养算法分析能力

  评价算法、改进算法

学习要求

- 循序渐进，切忌心浮气躁

  提高课外学习的时间和内容

  理解科学而不是背诵科学→读书

  正确对待考试

- 作习题

  华罗庚：“学数学不做习题等于入宝山而空返”

- 作实验

  计算机学科是一门科学性与工程性并重的学科，表现为理论和实践紧密结合的特征。

成绩组成

学期成绩=平时成绩（30%）+期末成绩（70%）

说明：

平时成绩=上课出勤（30%）+作业/测验（ 70% ）

上课出勤：上课点名，请假要有假条。

要求大家课后务必复习！切记！！

实验项目一

- **顺序表** 掌握线性表的顺序存储结构和操作特性，实现基于顺序表的基本操作。
- **链表** 掌握线性表的链式存储结构和操作特性，实现基于链表的基本操作。
- **栈** 掌握栈的顺序存储结构和操作特性，实现基于顺序栈的基本操作。
- **队列** 掌握队列的链式存储结构和操作特性，实现基于链队列的基本操作。

实验项目 二

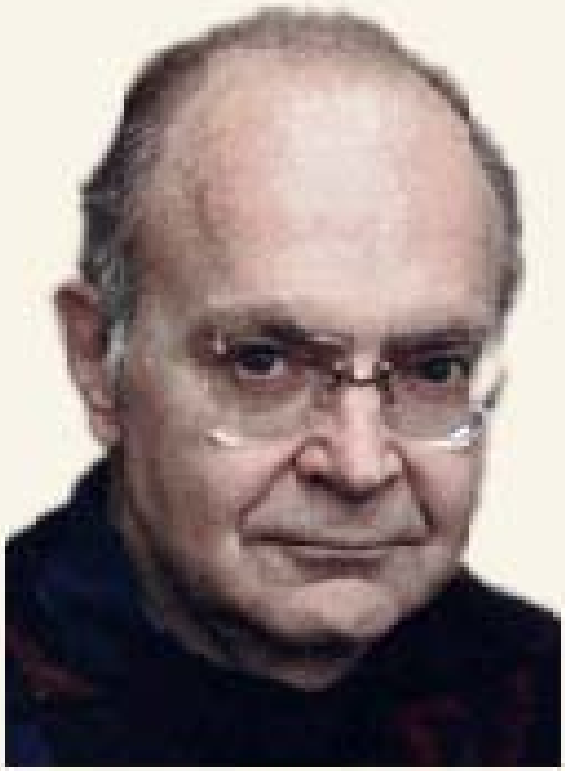
- **二叉树** 掌握二叉树的二叉链表存储结构，实现二叉树的先、中、后序遍历。
- **图** 建立无向图的邻接矩阵存储结构和有向图的邻接表存储结构，分别对其实现图的广/深度优先遍历。
- **顺序查找、折半查找** 实现顺序查找、折半查找算法，在一数据表中查找一元素，若没有此元素则进行插入，并保持此数据表有序。
- **直接插入、起泡、简单选择排序** 实现直接插入、起泡、简单选择排序算法。

第1章 绪 论

本章的基本内容是：

- 数据结构的兴起和发展
- 数据结构的研究对象
- 数据结构的基本概念
- 算法及算法分析

数据结构的创始人——克努思



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。

1.1 数据结构的兴起和发展

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

1.1 数据结构的兴起和发展

- 数据结构随着程序设计的发展而发展

1. 无结构阶段
2. 结构化阶段：数据结构＋算法＝程序
3. 面向对象阶段：（数据结构＋算法）＝程序

- 数据结构的发展并未终结

1.2 数据结构的研究对象

- 计算机求解问题:

  问题→抽象出问题的模型→求模型的解

- 问题——数值问题、非数值问题

  数 值 问 题→数学方程

  非数值问题→数据结构

1.2 数据结构的研究对象

例1 学籍管理问题——表结构

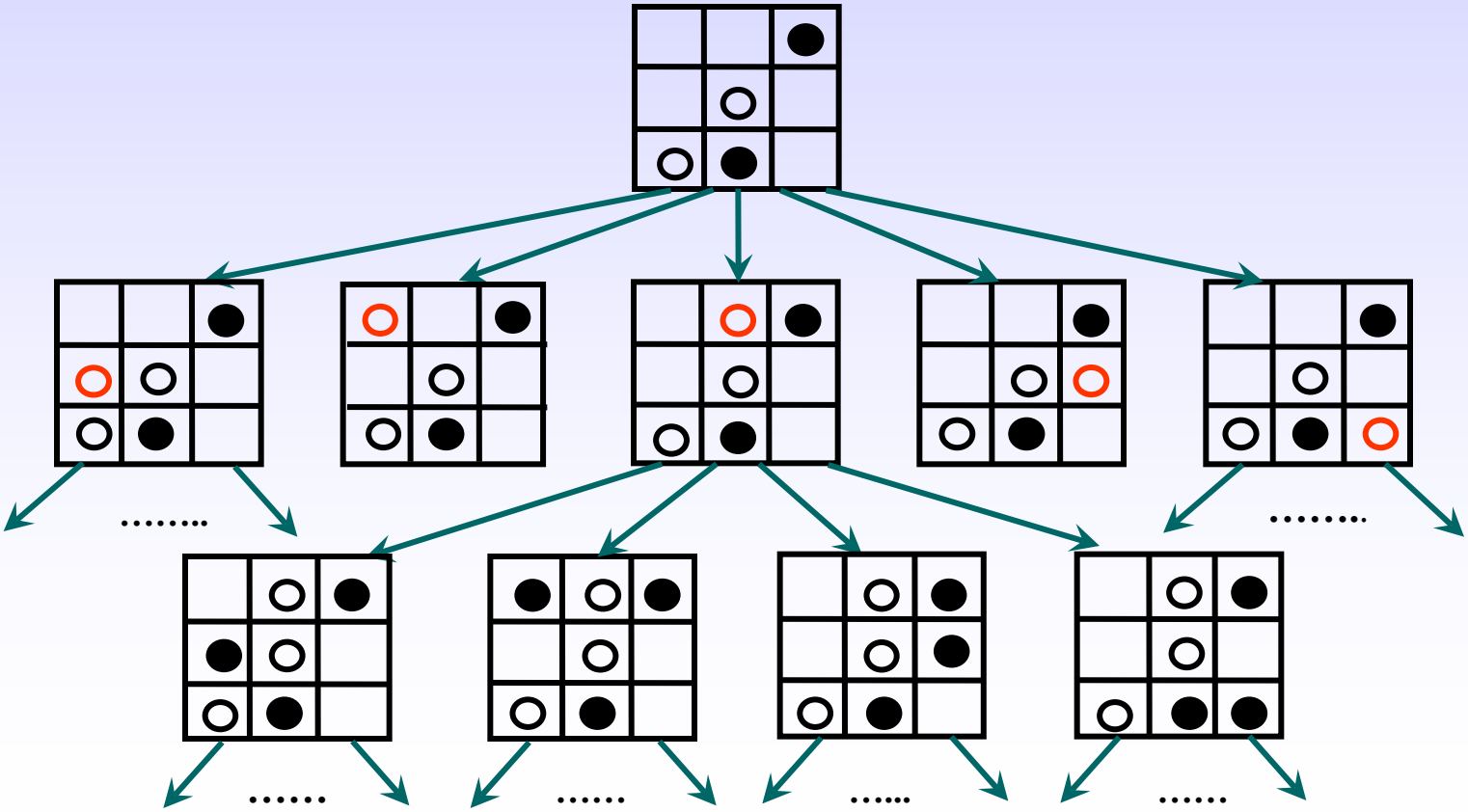
完成什么功能?各表项之间是什么关系？

学号	姓名	性别	出生日期	政治面貌
0001	王 军	男	**1983/09/02**	团员
0002	李 明	男	**1982/12/25**	党员
0003	汤晓影	女	**1984/03/26**	团员
…	…	…	…	…

1.2 数据结构的研究对象

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?

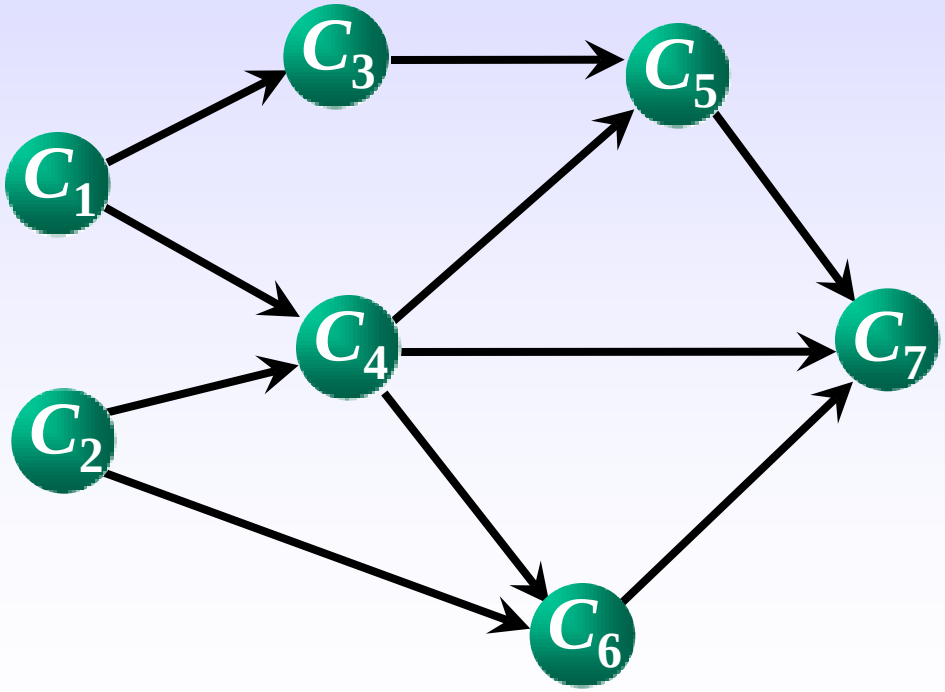


1.2 数据结构的研究对象

例3 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
C_1	高等数学	无
C_2	计算机导论	无
C_3	离散数学	C_1
C_4	程序设计	C_1, C_2
C_5	数据结构	C_3, C_4
C_6	计算机原理	C_2, C_4
C_7	数据库原理	C_4, C_5, C_6



1.2 数据结构的研究对象

数据结构是研究非数值问题中计算机的操作对象以及它们之间的关系和操作的学科。

1.3 数据结构的基本概念

数据结构的基本概念

数据：所有能输入到计算机中并能被计算机程序识别和处理的符号集合。

数值数据：整数、实数等

非数值数据：图形、图象、声音、文字等

数据元素：数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。

数据项：构成数据元素的不可分割的最小单位。

数据对象：具有相同性质的数据元素的集合。

1.3 数据结构的基本概念

数据、数据元素、数据项之间的关系

- ✓包含关系：数据是由数据元素组成，数据元素是由数据项组成。

- ✓数据元素是讨论数据结构时涉及的最小数据单位，其中的数据项一般不予考虑。

1.3 数据结构的基本概念

数据结构的基本概念

数据结构：相互之间存在一定**关系**的数据元素的集合。
按照视点的不同，数据结构分为逻辑结构和存储结构。

➢逻辑结构：指数据元素之间**逻辑关系**的整体。

⬇

关联方式或邻接关系

学籍管理问题中，表项之间的逻辑关系指的是什么？

人机对弈问题中，格局之间的逻辑关系指的是什么？

教学计划编排问题中，课程之间的逻辑关系指的是什么？

数据的逻辑结构是从具体问题抽象出来的**数据模型**

1.3 数据结构的基本概念

数据结构的基本概念

数据结构：相互之间存在一定关系的数据元素的集合。
按照视点的不同，数据结构分为逻辑结构和存储结构。

- 逻辑结构：指数据元素之间逻辑关系的整体。
- 存储结构：又称为物理结构，是数据及其逻辑结构在计算机中的表示。

⬇

内存

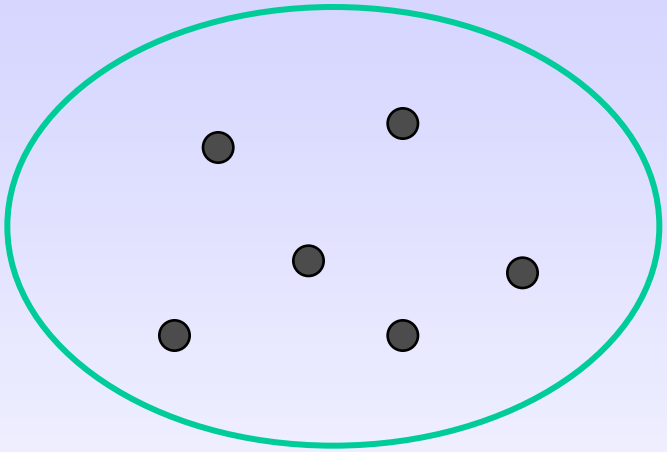
存储结构实质上是内存分配，
在具体实现时，依赖于计算机语言。

1.3 数据结构的基本概念

数据结构的基本概念

数据结构从逻辑上分为四类：

(1) 集合：数据元素之间就是“属于同一个集合”；



1.3 数据结构的基本概念

数据结构的基本概念

数据结构从逻辑上分为四类：

(1) 集合：数据元素之间就是“属于同一个集合”；

(2) 线性结构：数据元素之间存在着一对一的线性关系；

1.3 数据结构的基本概念

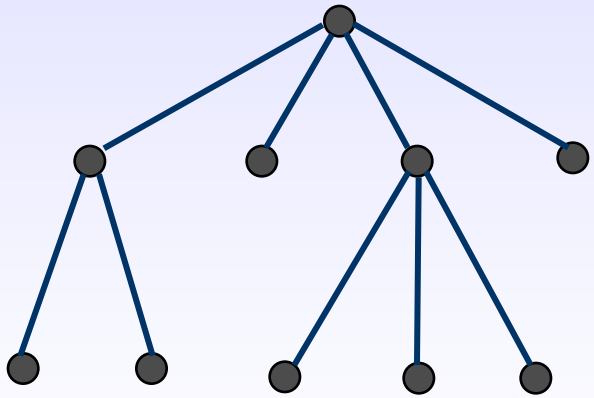
数据结构的基本概念

数据结构从逻辑上分为四类：

(1) 集合：数据元素之间就是“属于同一个集合”；

(2) 线性结构：数据元素之间存在着一对一的线性关系；

(3) 树结构：数据元素之间存在着一对多的层次关系；



1.3 数据结构的基本概念

数据结构的基本概念

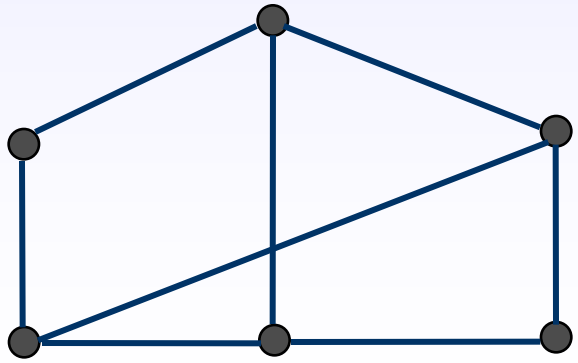
数据结构从逻辑上分为四类：

(1) 集合：数据元素之间就是“属于同一个集合”；

(2) 线性结构：数据元素之间存在着一对一的线性关系；

(3) 树结构：数据元素之间存在着一对多的层次关系；

(4) 图结构：数据元素之间存在着多对多的任意关系。



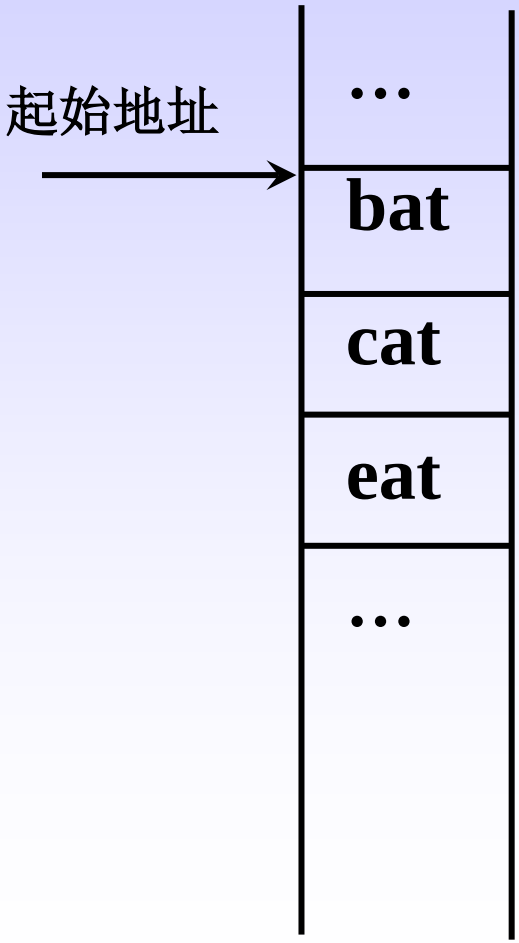
1.3 数据结构的基本概念

数据结构的基本概念

通常有两种存储结构：

1. 顺序存储结构：用一组连续的存储单元依次存储数据元素，数据元素之间的逻辑关系由元素的存储位置来表示。

例：（bat, cat, eat）



1.3 数据结构的基本概念

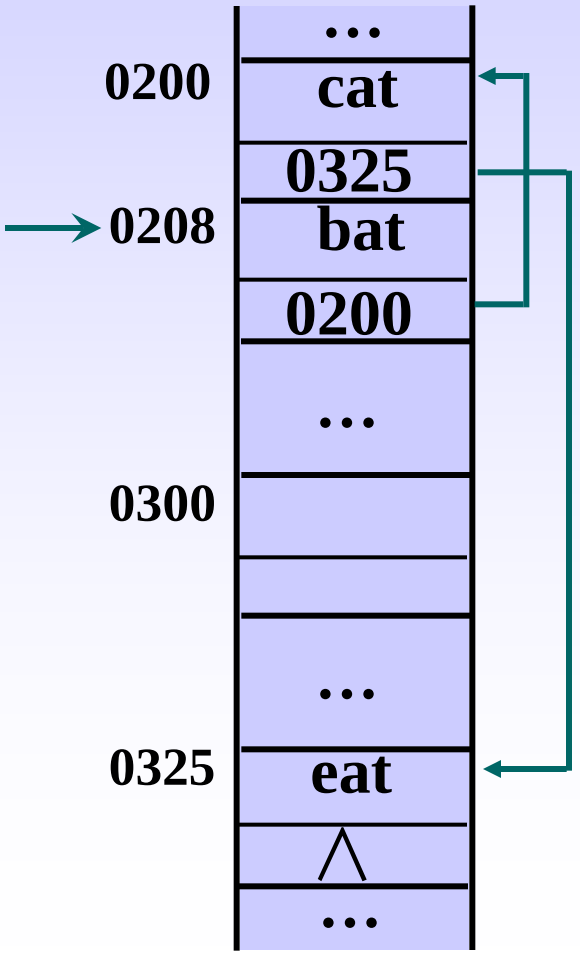
数据结构的基本概念

通常有两种存储结构：

1. 顺序存储结构：用一组连续的存储单元依次存储数据元素，数据元素之间的逻辑关系由元素的存储位置来表示。

2. 链接存储结构：用一组任意的存储单元存储数据元素，数据元素之间的逻辑关系用指针来表示 。

例：（bat, cat, eat）



1.3 数据结构的基本概念

逻辑结构和存储结构之间的关系

- 数据的逻辑结构属于用户视图，是面向问题的，反映了数据内部的构成方式；数据的存储结构属于具体实现的视图，是面向计算机的。

- 一种数据的逻辑结构可以用多种存储结构来存储，而采用不同的存储结构，其数据处理的效率往往是不同的。

1.3 数据结构的基本概念

数据结构的访问接口

- 对数据结构的访问是指对数据的读取、修改、加工、处理等操作。
- 数据结构的基本操作：各种应用都能通过这些操作实现对数据结构的各种访问。

  基本操作的特性：抽象性、基本性、完备性、一般性
- 访问接口：操作的调用形式与规范（例如形参表、返回类型等）。
- 数据结构的访问接口：数据结构基本操作接口的全体。

1.3 数据结构的基本概念

抽象数据类型

1. 数据类型（Data Type）：一组值的集合以及定义于这个值集上的一组操作的总称。

例如：C++中的整型变量

2. 抽象（Abstract）:抽出问题本质的特征而忽略非本质的细节。

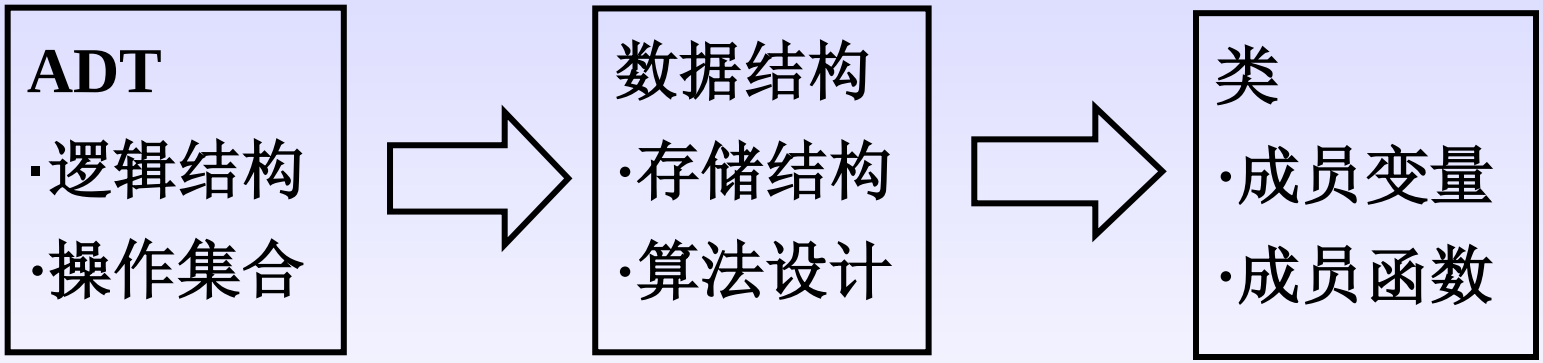
例如： 地图、驾驶汽车

3. 抽象数据类型（Abstract Data Type，ADT）:一个数据结构以及定义在该结构上的一组操作的总称。

1.3 数据结构的基本概念

ADT是对数据类型的进一步抽象

ADT的不同视图



(a)使用视图
ADT的定义

(b) 设计视图
ADT的设计

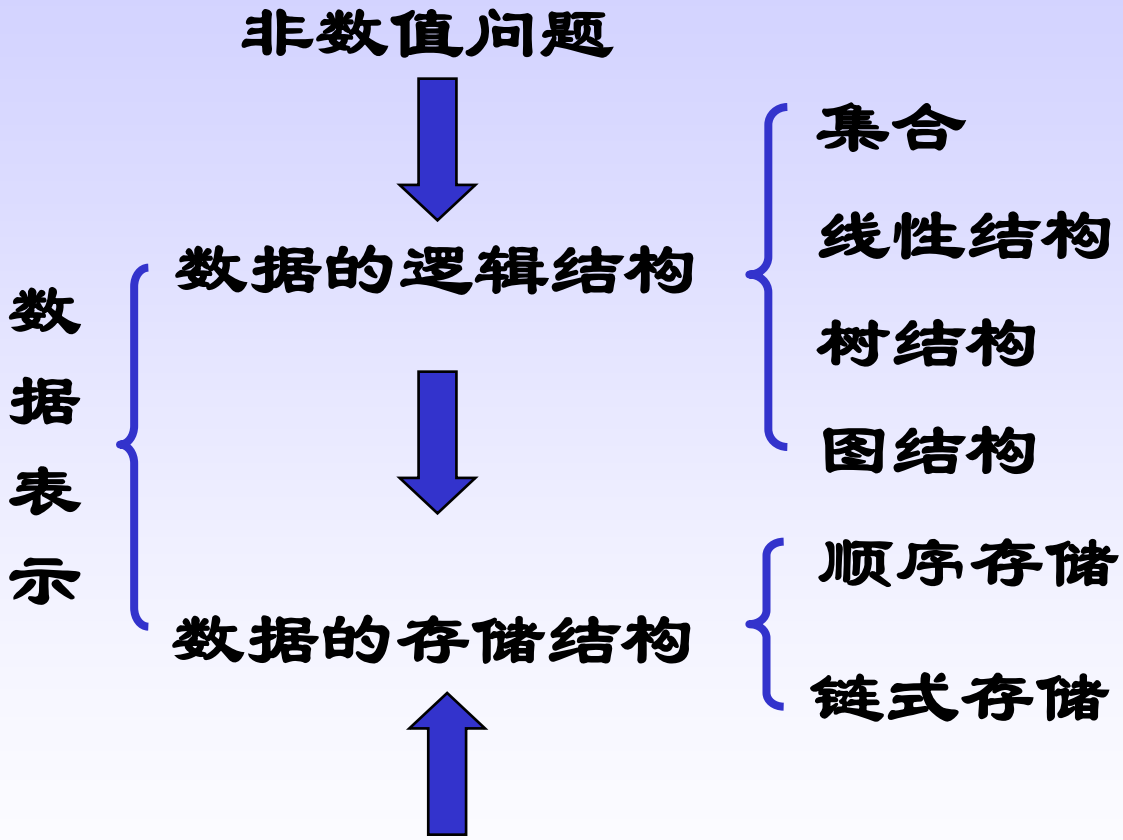
(c) 实现视图
ADT的实现

1.3 数据结构的基本概念

ADT的定义形式

ADT 抽象数据类型名
Data
　　数据元素之间逻辑关系的定义
Operation
　　操作1
　　　　前置条件：执行此操作前数据所必须的状态
　　　　输　　入：执行此操作所需要的输入
　　　　功　　能：该操作将完成的功能
　　　　输　　出：执行该操作后产生的输出
　　　　后置条件：执行该操作后数据的状态
　　操作2
　　　　……
　　……
　　操作n
　　　　……
endADT

1.3 数据结构的基本概念（小结）



数据的操作：插入、删除、修改、检索、排序等

1.4 算法及算法分析

算法的相关概念

1.算法（Algorithm）:是对特定问题求解步骤的一种描述，是指令的有限序列。

2. 算法的五大特性：

(1) 输入：一个算法有零个或多个输入。

(2) 输出：一个算法有一个或多个输出。

(3) 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。

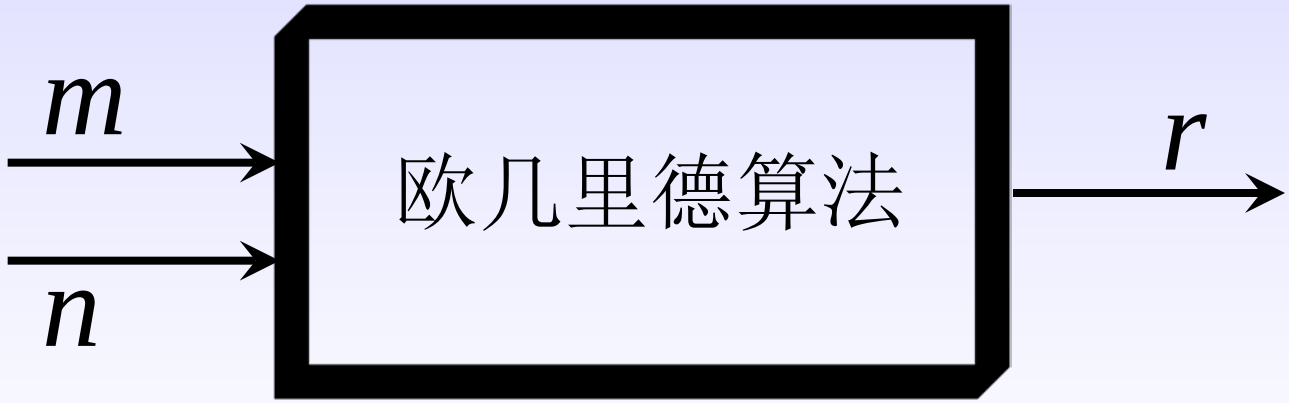
(4) 确定性：算法中的每一条指令必须有确切的含义，对于相同的输入只能得到相同的输出。

(5) 可行性：算法描述的操作可以通过已经实现的基本操作执行有限次来实现。

1.4 算法及算法分析

例：欧几里德算法——辗转相除法

求两个自然数 m 和 n 的最大公约数



1.4 算法及算法分析

算法的描述方法——自然语言

优点：容易理解

缺点：冗长、二义性

使用方法：粗线条描述算法思想

注意事项：避免写成自然段

1.4 算法及算法分析

例：欧几里德算法

自然语言

① 输入m和n；

② 求m除以n的余数r；

③ 若r等于0，则n为最大公约数，算法结束；否则执行第④步；

④ 将n的值放在m中，将r的值放在n中；

⑤ 重新执行第②步。

1.4 算法及算法分析

算法的描述方法——流程图

优点：流程直观

缺点：缺少严密性、灵活性

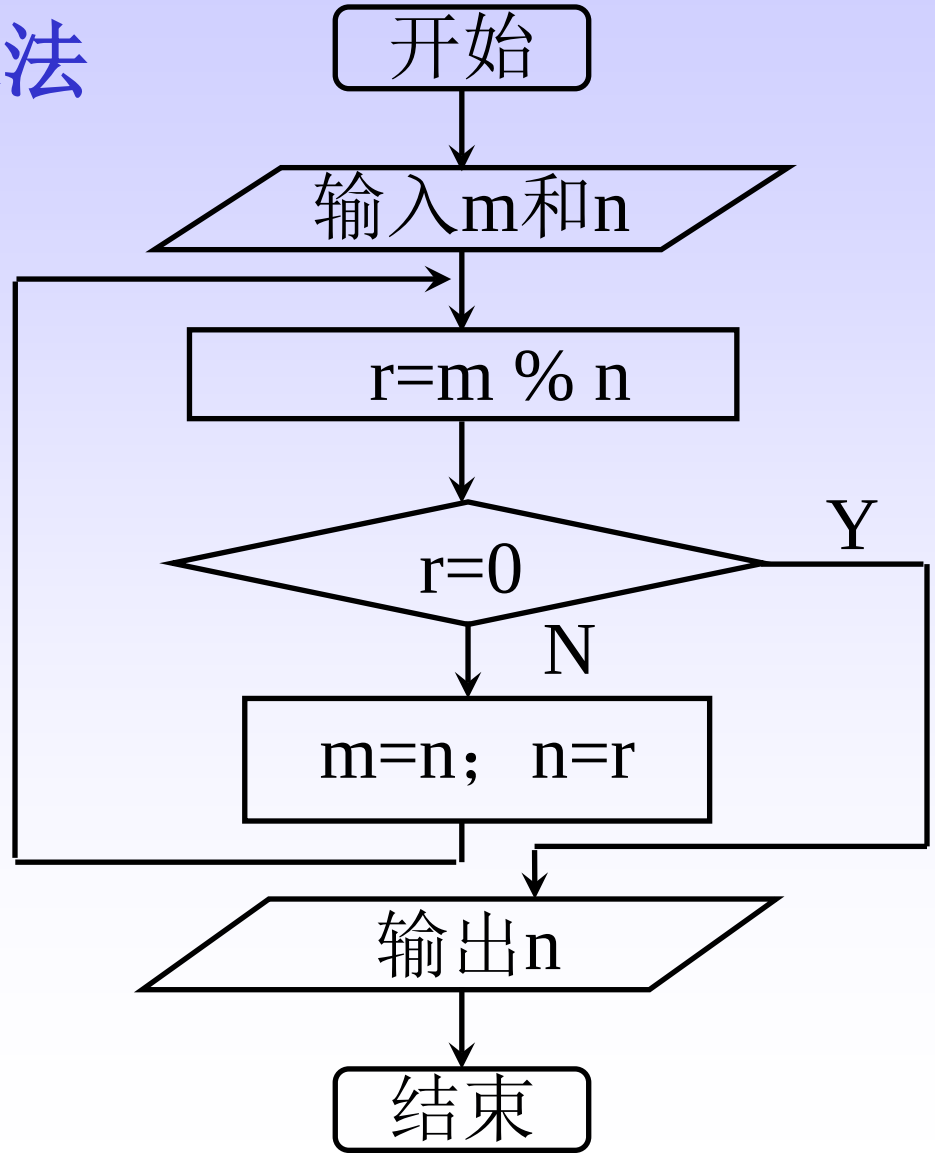
使用方法：描述简单算法

注意事项：注意抽象层次

1.4 算法及算法分析

例：欧几里德算法

流程图



1.4 算法及算法分析

算法的描述方法——程序设计语言

优点：能由计算机执行

缺点：抽象性差，对语言要求高

使用方法：算法需要验证

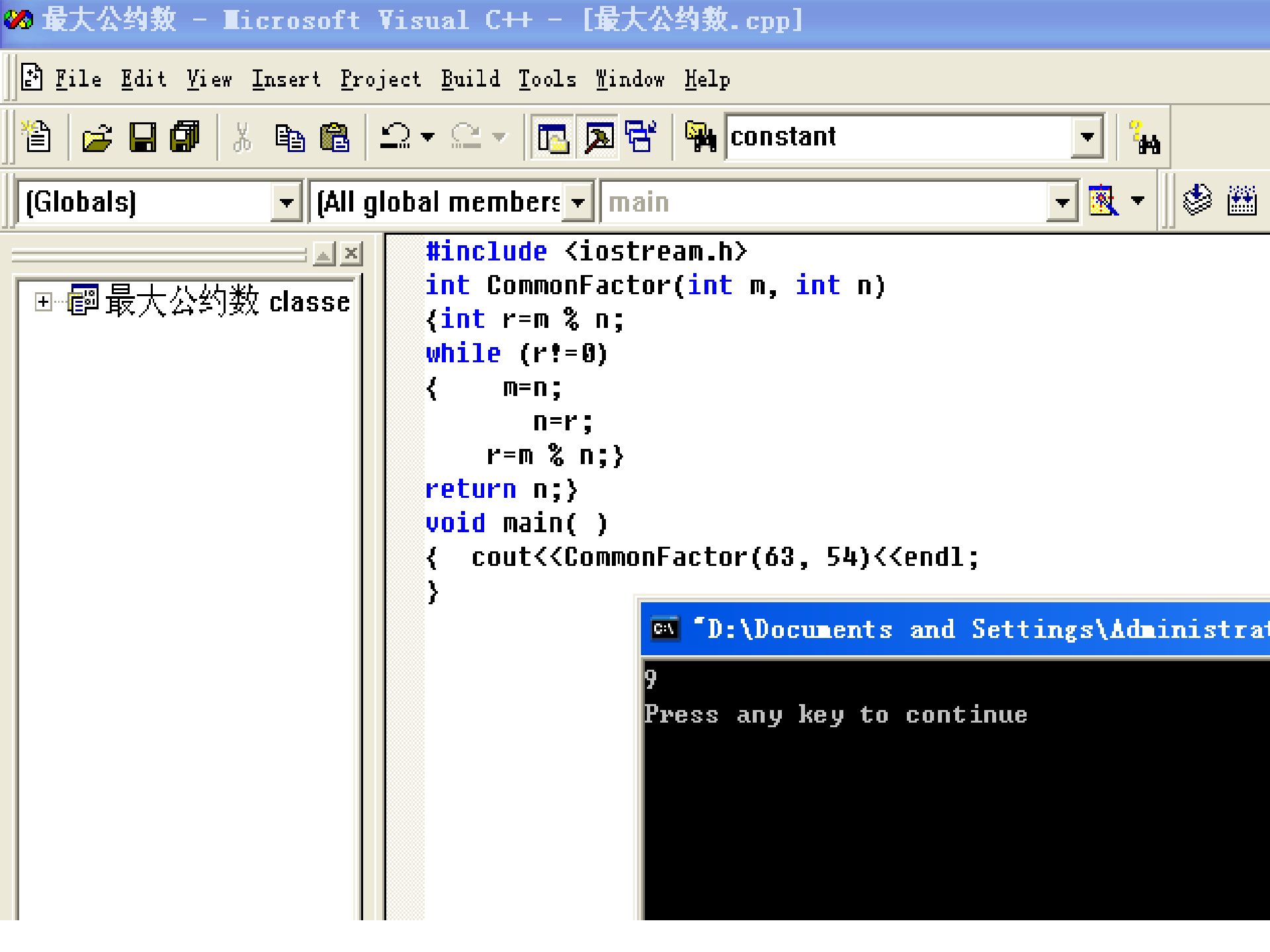
注意事项：将算法写成子函数

1.4 算法及算法分析

例：欧几里德算法

程序设计语言

```
#include <iostream.h>
int CommonFactor(int m, int n)
{
    int r=m % n;
    while (r!=0)
    {
       m=n;
       n=r;
       r=m % n;
    }
    return n;
}
void main( )
{
   cout<<CommonFactor(63, 54)<<endl;
}
```


最大公约数 - Microsoft Visual C++ - [最大公约数.cpp]
File Edit View Insert Project Build Tools Window Help
constant
(Globals)
(All global members
main
最大公约数 classe
#include <iostream.h>
int CommonFactor(int m, int n)
{int r=m % n;
while (r!=0)
{ m=n;
n=r;
r=m % n;}
return n;}
void main( )
{ cout<<CommonFactor(63, 54)<<endl;
}
"D:\Documents and Settings\Administrat
9
Press any key to continue

1.4 算法及算法分析

算法的描述方法——伪代码

伪代码（Pseudocode）：介于自然语言和程序设计语言之间的方法，它采用某一程序设计语言的基本语法，操作指令可以结合自然语言来设计。

优点：表达能力强，抽象性强，容易理解

使用方法：7 ± 2

1.4 算法及算法分析

例：欧几里德算法

伪代码

1. r = m % n;
2. 循环直到 r 等于0
   2.1 m = n;
   2.2 n = r;
   2.3 r = m % n;
3. 输出 n ;

上述伪代码再具体一些，用C++的函数来描述。请同学们自行完成！

1.4 算法及算法分析

算法分析

度量算法效率的方法：

- 事后统计：将算法实现，测算其时间和空间开销。

缺点：(1) 编写程序实现算法将花费较多的时间和精力；

(2) 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。

1.4 算法及算法分析

算法分析

算法分析（Algorithm Analysis）：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 算法及算法分析

算法分析

算法的时间复杂度分析

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

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

1.4 算法及算法分析

算法分析

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。

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

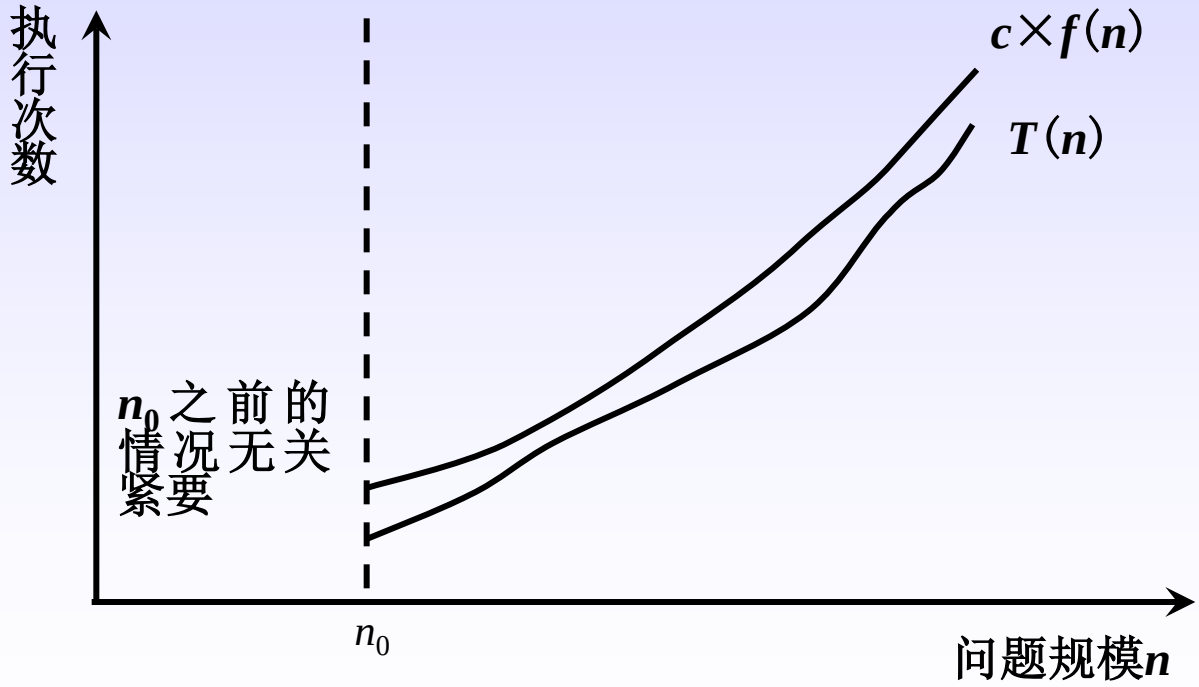
问题规模：n

基本语句：x++

1.4 算法及算法分析

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

1.4 算法及算法分析

算法分析——大O符号

定理：若$A(n)=a_m n^m+a_{m-1}n^{m-1}+\ldots+a_1 n+a_0$是一个$m$次多项式，则$A(n)=O(n^m)$。

说明：在计算算法时间复杂度时，可以忽略所有低次幂和最高次幂的系数。

1.4 算法及算法分析

算法分析

例1-5
```
++x;
```

例1-6
```
for（i=1; i<=n; ++i）
    ++x;
```

例1-7
```
for (i=1; i<=n; ++i)
    for (j=1; j<=n; ++j)
        ++x;
```

例1-8
```
for (i=1; i<=n; ++i)
    for (j=1; j<=i-1; ++j)
        ++x;
```

1.4 算法及算法分析

算法分析

例1-9
```
for (i=1; i<=n; ++i)
    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];
    }
```

例1-10
```
for (i=1; i<=n; i=2*i)
    ++x;
```

$$O(1)<O(\log_2 n)<O(n)<O(n\log_2 n)<O(n^2)<O(n^3)$$
$$<\ldots<O(2^n)<O(n!)$$

1.4 算法及算法分析

最好情况、最坏情况、平均情况

例：在一维整型数组A[n]中顺序查找与给定值k相等的元素（假设该数组中有且仅有一个元素值为k）。

```
int Find(int A[ ], int n)
{
    for (i=0; i<n; i++)
        if (A[i]= =k) break;
    return i;
}
```

基本语句的执行次数是否只和问题规模有关？

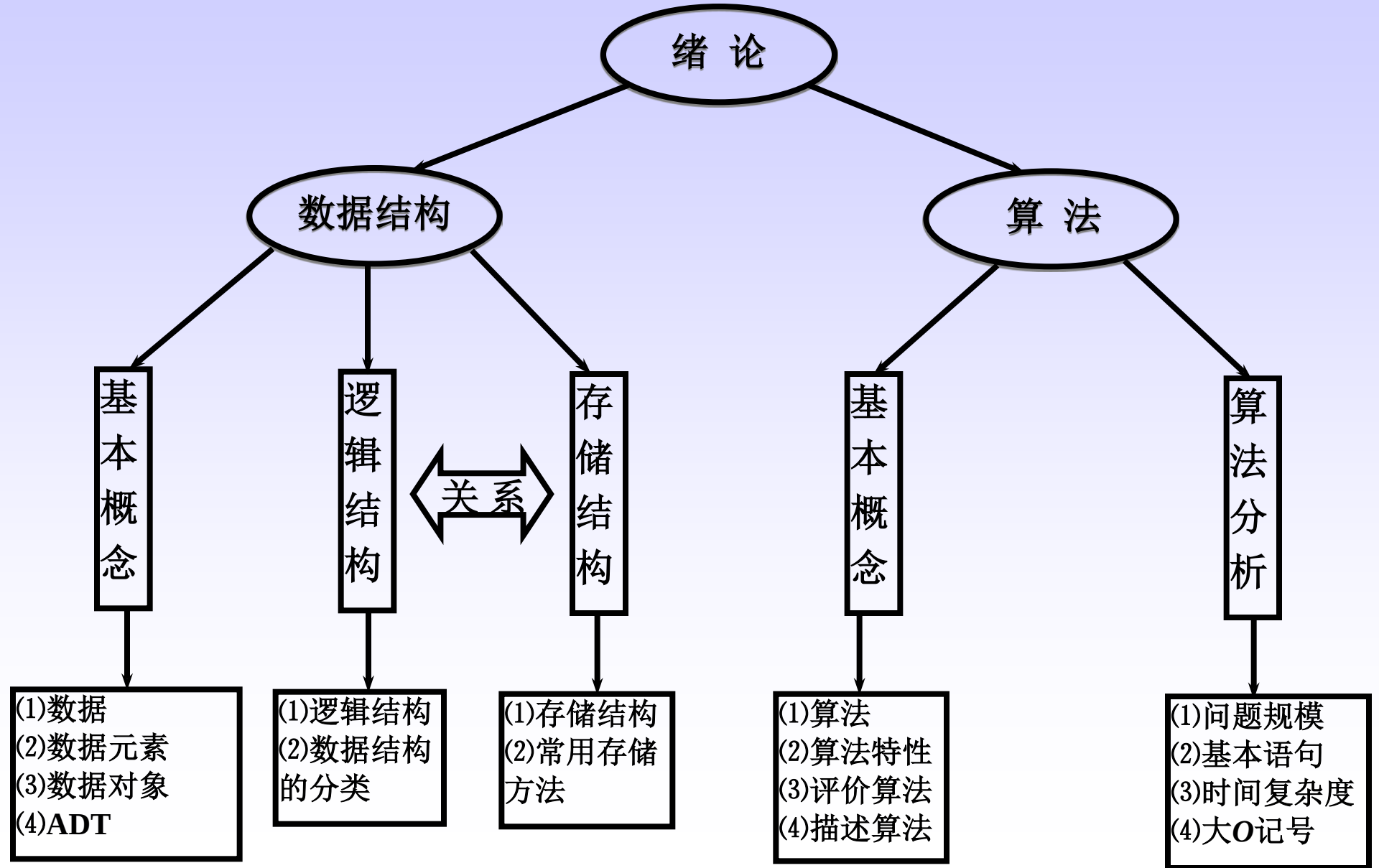
1.4 算法及算法分析

最好情况、最坏情况、平均情况

结论：如果问题规模相同，时间代价与输入数据有关，则需要分析最好情况、最坏情况、平均情况。

- ✓最好情况：出现概率较大时分析
- ✓最差情况：实时系统
- ✓平均情况：已知输入数据是如何分布的，通常假设等概率分布

本章小结——知识结构图


绪 论
数据结构
算 法
基本概念
逻辑结构
关系
存储结构
基本概念
算法分析
(1)数据
(2)数据元素
(3)数据对象
(4)ADT
(1)逻辑结构
(2)数据结构的分类
(1)存储结构
(2)常用存储方法
(1)算法
(2)算法特性
(3)评价算法
(4)描述算法
(1)问题规模
(2)基本语句
(3)时间复杂度
(4)大O记号

上机：

复习：c++多文件结构

自学：模板类

上机实验：

P169 顺序表操作验证