北京航空航天大學
BEIHANG UNIVERSITY


数据结构与程序设计
(Data Structure and Programming)

前导

北航计算机学院 晏海华

课程重要性

- 是信息类专业学生最重要的专业技能之一
- 是其它许多计算机重要专业课程（操作系统、编译原理）的基础
- 是信息类核心专业课程之一

作业及参考书

- 教师：晏海华
- 联系方式：新主楼G916，Tel：82328212，82317625　Email:yhh@buaa.edu.cn
- 参考书：
  - **《C程序设计导引（第2版）》，尹宝林编著，机械工业出版社，2020**
  - **《C程序设计语言》，B.W.Kernighan, D.M.Ritchie，机械工业出版社 2012，徐宝文等译**
  - **《数据结构教程（第3版）》 唐发根 北京航空航天大学出版社 2017**
  - **《数据结构与算法分析（第二版）》—C语言描述》，Mark Allen Weiss著 冯舜玺译，机械工业出版社 2013**
- 课程网站(课程信息、作业、答疑及考试)：

  http://judge.buaa.edu.cn (http://10.251.0.206)
- 考核方式：
  - 作业及考试均采用上机方式
  - 作业占20%，期中考试占30%，期末占50%

课程网站(课程信息、作业、答疑及考试)

- 网址（建议用Chrome+**Windows**访问）：

  http://judge.buaa.edu.cn

  (http://10.251.0.206)

- 用户名：学号　　密码：学号
- 选择课程：数据结构与程序设计（信息大类）
- 请登录后尽快修改密码！

课程MOOC资源：

https://coursehome.zhihuishu.com/courseHome/1000103811#teachTeam

目标和方法

动手能力非常重要，MIT的校训是“脑与手”（Mens et Manus）。

目标

培养学生利用所学知识解决问题的能力，包括解决没有遇到过的问题的能力。（即程序设计能力）

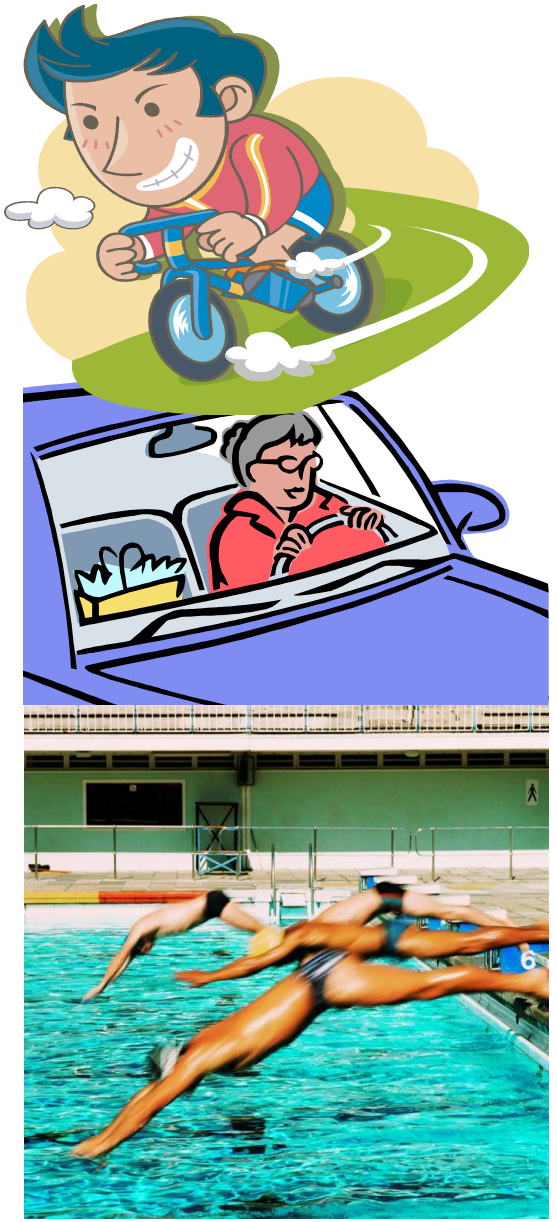
2016.6.2国际工程联盟接纳中国为《华盛顿协议》成员。国际认可的工程教育强调的是学生应具有**应用知识解决问题能力，包括解决没有遇到过的问题的能力**。北航已加入了《华盛顿协议》。
2017.2 我国开始推动新工科建设，**强调工程实践能力和创新能力**培养。

方法

中国著名桥梁专家、工程教育家茅以升先生提出了：
“习而学”、“做中学”（Learning by Doing）的工程教育思想

获得程序设计能力的唯一途径：

上机实践(编程)!
（Try!!!）



忠告

也许有些同学会认为本课程作业太重（需要花很多课外时间），是一个负担，但我给大家的忠告是：

“负担通常会带来痛苦，但是和将来的后悔相比，哪个更痛苦一些?”

(这个问题也写在 NBA 凯尔特人队的训练馆里)

“What Hurts More,
The Pain of Hard Work
or The Pain of Regret?”

啥佛图书馆墙壁训言：

“学习时的苦痛是暂时的，未学到的痛苦是终生的。”

(Time the study pain is temporary, has not learned the pain is life-long.)

程序设计教材与参考书

北京航空航天大學 BEIHANG UNIVERSITY

教材：

《C程序设计导引（第2版）》

尹宝林编著，机械工业出版社，2020

参考书

《C程序设计语言》，B.W.Kernighan, D.M.Ritchie, 机械工业出版社 2012，徐宝文等译

《C程序设计思想与方法》，尹宝林编著，机械工业出版社，2009

数据结构教材与参考书

北京航空航天大學 BEIHANG UNIVERSITY

教材：

《数据结构教程》
第3版
唐发根 编著
北京航空航天大学出版社 2017

参考书1*

《数据结构与算法分析-C语言描述》，M.A.Weiss著 冯舜玺译，机械工业出版社 2013

参考书2*

《C++数据结构与算法分析(第4版)》，M.Drozdek著 徐丹 吴伟敏译，清华大学出版社 2014

参考书3

《大话数据结构》，程杰著，清华大学出版社 2011 ··· 一本通俗读本

参考书4

《数据结构(C语言版)》，严蔚敏 吴伟民编著，清华大学出版社

课程作业要求

- 普通作业
  - 共七套（基础1、基础2、线性表、栈和队、树、图、查找与排序），每套有一组选择填空概念题，多个应用编程题
  - 普通作业应用题主要考查学生对单一知识点的应用能力
- 综合性能（Project）作业
  - 综合编程题，占整个作业20分中的5分，其测试数据规模较大，主要考查学生对数据结构综合应用的能力。有3道要求完全一样的题目，但测试数据的规模不一样：
    - 小测试数据题评判要求：正确占100%，不考查性能。同时为了方便学生调试，给出了跟测试数据基本一样的样例数据。**占分：2.5分**。
    - 大测试数据题评判要求：正确占20%，性能占80%。即在给定时间（100秒）内正确完成，得20%分，性能部分分数以运行最快的前10%程序为基准（其为满分），依次计算得分。运行结果不正确或在给定时间内没有运行结束，则不得分。**占分：2.0分**。
    - 期末评判题：平时只提交，不评判，课程结束后评判。测试数据比大测试数据题要大一些，只考查正确性（可扩展性），不考查性能。**占分0.5分**。

- 应在规定时间内提交；
  注：每套普通作业大约开放3~4个星期
- 一定要按照题目要求提交，比如：输入、输出数据格式，提交文件名称等等；
- 严禁抄袭，编程作业查出抄袭要处理！
  - 不能拷贝提交他人源代码（提交同学的代码）
  - 不能下载提交网上源代码（提交网上的代码）
  - 不要将自己的源码拷贝给他人（可能被抄袭）

程序设计与程序设计语言

- 程序设计(Programming): 为计算机解决问题所需的分析、设计、编写及调试程序过程。（The process of planning, writing, testing, and correcting the steps required for a computer to solve a problem or perform an operation.）
- 程序设计语言(Programming Language): 用来表达程序的计算机能够执行的人工语言，是程序设计过程中用到的一种工具。如，Fortran, C, C++, Java, C#, Python等。

可以这理解两者之间的关系：
程序设计语言是我们解决问题过程（程序设计）中用到的工具之一（可能还有其它工具，如分析与设计工具。

为什么要学C语言程序设计

- C语言是一种简洁精练的语言，所涉及的概念比较少，语法和结构也简单，主要特点：
  - 表达能力强，支持结构化程序设计；
  - 语言简洁；
  - **代码效率高**： C编写的程序仅比用汇编语言编写的程序仅相差20%（C语言是运行**效率最高**的高级语言）；
  - 可移植性好；
  - 特别适合编写操作系统、编译程序、数据库系统、嵌入式软件及图形/图象处理等对性能要求高的软件；
- C语言仍是目前广泛使用的编程语言，其应用领域遍及系统软件（如Windows,UNIX, Linux...）、嵌入式软件（飞机机载、导弹弹载、轨道交通、通信、...）、图形图像处理软件、游戏软件...
- C语言是目前广泛流行的面向对象语言C++、C#及Java的基础

为什么要学C语言程序设计*

TIOBE 编程语言社区排行榜是编程语言流行趋势的一个指标，每月更新，该排名是基于互联网上有经验的程序员、课程和第三方厂商的数量。

2023 IEEE Spectrum 编程语言排行榜

*IEEE Spectrum：一本IEEE（美国电气电子工程师学会）出版的旗舰杂志，旨在探讨未来技术发展趋势及其对社会和商业的影响，主要内容为报道国际航空航天、计算机和电信、生物医学工程、能源和消费电子等领域的最新技术进展和成果。

TIOBE 最新(202402)编程语言排行榜

Feb 2024	Feb 2023	Change	Programming Language	Ratings
1	1		Python	15.16%
2	2		C	10.97%
3	3		C++	10.53%
4	4		Java	8.88%
5	5		C#	7.53%
6	7	^	JavaScript	3.17%
7	8	^	SQL	1.82%
8	11	^	Go	1.73%
9	6	v	Visual Basic	1.52%
10	10		PHP	1.51%

TIOBE 编程语言（1988-2023）历史排行榜

Programming Language	2023	2018	2013	2008	2003	1998	1993	1988
Python	1	4	8	7	12	25	19	-
C	2	2	1	2	2	1	1	1
Java	3	1	2	1	1	18	-	-
C++	4	3	4	4	3	2	2	5
C#	5	5	5	8	9	-	-	-
Visual Basic	6	15	-	-	-	-	-	-
JavaScript	7	7	11	9	8	21	-	-
SQL	8	251	-	-	7	-	-	-
Assembly language	9	13	-	-	-	-	-	-
PHP	10	8	6	5	6	-	-	-
Objective-C	19	17	3	44	53	-	-	-
Ada	25	28	18	19	15	9	6	3
Lisp	29	31	12	16	14	8	5	2
Pascal	189	147	15	18	99	12	3	14
(Visual) Basic	-	-	7	3	5	3	8	6

一些著名的软件编写语言*

分类	软件	编写语言
操作系统	**Microsoft Windows**	汇编 -> C -> C++
	Linux/UNIX	C
	Apple MacOS	主要为 C，部分为 C++
	Google Android	C
图形界面	Google Desktop Search	C++
	Microsoft Windows Desktop Search	C++
办公软件	**Microsoft Office**	汇编 -> C -> C++
	Adobe Systems **Acrobat Reader/Distiller**	C++
关系数据库	**Oracle**	主要为 C++
	MySQL	C++
	Microsoft SQL Server	汇编 -> C->C++
浏览器	**IE**	C++
	Google Chrome	C++
邮件客户端	**Microsoft Outlook**	C++
开发环境 IDE	**Microsoft Visual Studio**	C++
	Eclipse	Java （其图形界面 SWT 基于 C/C++）
	Code::Blocks	C++
虚拟机	Java Virtual Machine (JVM)	Java 虚拟机：C++
图形处理	Adobe **Photoshop**	C++
搜索引擎	**Google**	主要为 C++
游戏	**星际争霸、魔兽争霸、CS、帝国时代、跑跑卡丁车、传奇、魔兽世界....**	C、C++
编译器	**Microsoft Visual C++ 编译器、gcc (GNU C compiler)**、PHP、Perl	**C、C++**

C语言历史

肯·汤普逊（左）和丹尼斯·里奇（右）

- C语言的产生与UNIX操作系统是密不可分的：
  - UNIX由Bell Lab的K.Thompson和D.M.Ritchie最先在1969年开发的O.S.（它的前身是MIT和AE开发的Multies）
  - 1970年，V1，V2版在PDP-7机上用*汇编语言*实现
  - 1971年V3 PDP11/23； 1972年V4 PDP11/45
  - 1972年，D.M.Ritchie开发出新语言C。（C ← B ← BCPL ← CPL单数据型语言）
  - 1973年，Ritchie和Thompson用C改写了UNIX核心（90%）即V5

Linux是一种开放源码的类Unix操作系统。Linux可安装在各种计算机硬件设备中，从手机、平板电脑，到台式计算机、大型机和超级计算机。世界上运算最快的10台超级计算机运行的都是Linux操作系统。Linux得名于计算机业余爱好者Linus Torvalds。当然它本身也是由C语言开发的。

Android是一种基于Linux的自由及开放源代码的操作系统，主要使用于移动设备，如智能手机和平板电脑，由Google公司和开放手机联盟领导及开发。其底层核心用C/C++开发。

C语言历史*

- C语言的原型ALGOL 60语言。（也称为A语言）
- 1963年，剑桥大学将ALGOL 60语言发展成为CPL(Combined Programming Language)语言。
- 1967年，剑桥大学的Matin Richards 对CPL语言进行了简化，于是产生了BCPL语言。
- 1970年，美国贝尔实验室的Ken Thompson将BCPL进行了修改，并为它起了一个有趣的名字“B语言”。意思是将CPL语言煮干，提炼出它的精华。并且他用B语言写了第一个UNIX操作系统。而在1973年，B语言也给人“煮”了一下，美国贝尔实验室的D.M.RITCHIE在B语言的基础上最终设计出了一种新的语言，他取了BCPL的第二个字母作为这种语言的名字，这就是C语言。

  由此可见，最早的C编译器当然是B语言写的。

C语言历史*

在日益纷繁复杂的程序设计语言王国中，C语言因其简洁、有效、通用的特性而始终占据一席之地。被誉为“C语言之父”，同时也是操作系统Unix之父的C语言发明人之一丹尼斯·里奇2015年10月9日以70岁之龄辞世。

1983年，美国计算机协会将当年的图灵奖破例颁给了作为软件工程师的肯·汤普逊与里奇，获奖原因是他们“研究发展了通用的操作系统理论，尤其是实现了Unix操作系统”。

在众多的国际互动论坛上，计算机爱好者们以特有的方式纪念这位编程语言的重要奠基人。许多网友的发帖中没有片言只字，仅仅留下一个分号“;”

“现在的程序编写朝着越来越冗长庞大的方向发展，而C语言虽然属于相对‘低级’的编程语言，但它的简洁之美是无可替代的。”

“感谢丹尼斯·里奇，令我们拥有这一简洁而美丽的语言。”

“C语言之父”也是“黑客之父”-关于里奇的故事

在Unix研发成功后不久，安装了这一程序的PDP-11被放在贝尔实验室供大家使用。有一天，大家发现两位创始人总是可以得到最高的权限轻松进入他们的帐户，在贝尔实验室这种高人云集的地方，这简直是太不能容忍的事情了。于是，若干愤懑的同事仔细分析Unix代码，找到后门，修改后再重新编译整个Unix程序。当所有人都以为这个世界应该从此清静了的时候，却发现他们的帐户权限还是很容易泄露。直到很多年后，肯和里奇才道出其中的原委原来代码里确实存在后门，不过并不在Unix代码中，而是藏在编译Unix的编译器里。

数据结构 + 算法设计与分析

主要研究数据之间的关系和数据的组织方式

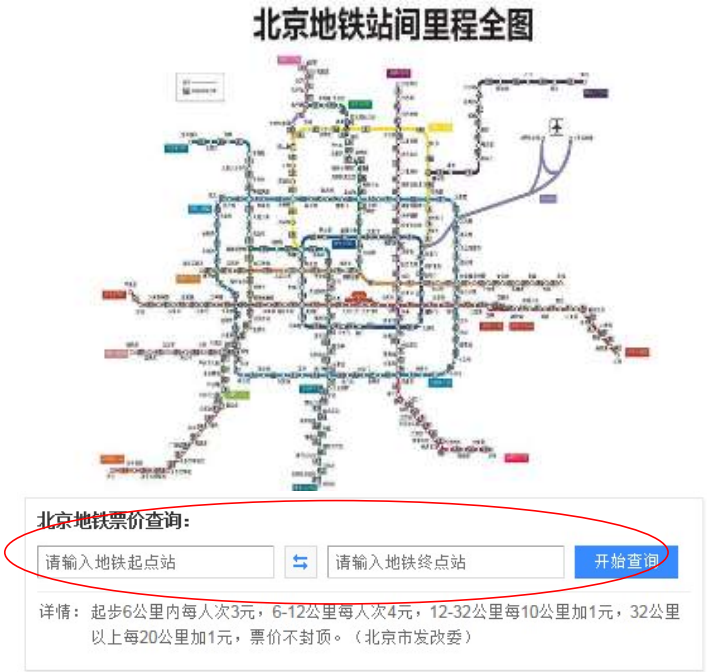
主要研究常用算法的设计和算法优劣的分析（性能）

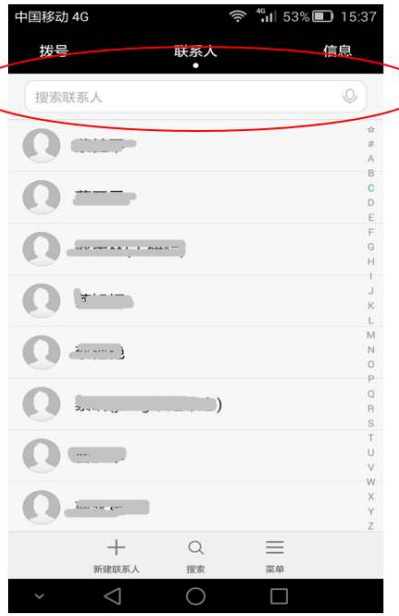
“算法（algorithm）是解决特定问题求解步骤的描述，在计算机中表现为指令的有限序列，并且每条指令表示一个或多个操作。”

数据结构、算法设计与分析

数据结构：研究数据的关系与组织

算法设计与分析：研究算法的设计和优劣的分析（性能）





索引	地址
M	
W	
Y	
……	……

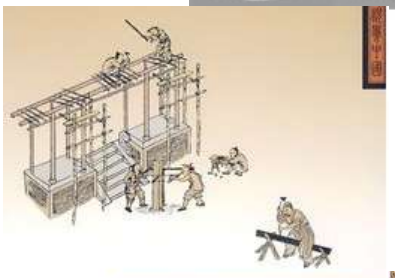
姓名	电话号码
马俊如	82338211
马志杰	82338212
王伟	82051123
王朦胧	62311284
……	……
杨为仁	82051122
杨扬	82051121
杨光	82051120

为什么要学习数据结构

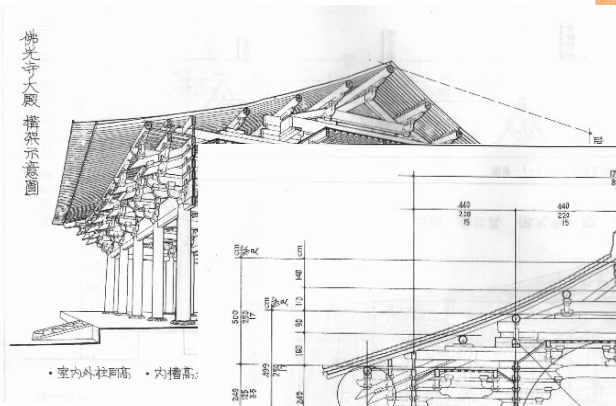
- 信息学科最重要的专业基础课程之一
- 后续专业课程学习的必要知识与技能准备
  - **编译技术**要使用栈、散列表及语法树
  - **操作系统**中使用队列（优先队列）、存储管理表及目录树
  - **数据库系统**运用线性表、多链表、及索引树
  - …
- 成为一名专业程序员应具备的基本技能（是微软、Google、百度、腾讯…面试中必不可少的内容）

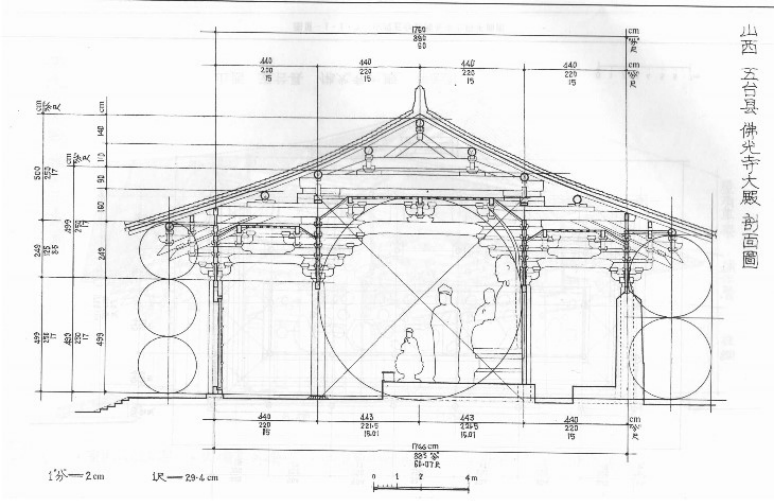
程序设计与数据结构

程序设计：如何使用程序设计语言（工具）来解决问题。



数据结构：研究数据的构造方法





软件与数据结构

软件 = ？　　程序 = ？

软件　=　程序　+　文档

程序　=　数据结构 + 算法

（图灵奖得主N.Wirth(沃斯)）

好数据结构 + 好算法 + 好风格 = 好程序

数据结构课程的主要内容和目的

主要内容

- 逻辑结构：通过抽象的方法研究被处理数据之间存在何种逻辑关系，即逻辑结构，有两大类：线性结构（线性表、数组、栈、字符串等）和非线性结构（树、图等）。
- 存储结构：研究每种逻辑关系在计算机内部如何表示，即存储结构或者物理结构，如线性存储结构、链式存储结构、索引结构和散列结构。
- 算法：研究在数据各种结构的基础上如何对数据实施有效的操作或处理（创建、清除、插入、删除、搜索、更新、访问、遍历等）。

主要目的

掌握数据处理的基本原理和方法，更好地进行算法设计与算法分析，提高程序设计的水平和能力。

数据结构

绪论

内容

1 什么是数据结构

2 算法及其描述

3 算法分析的基本概念

1 什么是数据结构

1.1 名词术语

数据

描述客观事物的数字、字符以及一切能够输入到计算机中，并且能够被计算机程序处理的 **符号的集合**

数据元素

数据这个集合中的一个一个的元素。

数据对象

具有相同特性的数据元素的 **集合**

自然数（1, 2, 3, 4, … ）

数 据

数据元素

1

(25, 78, 36, 100, 28, 45)

数列

2

(‘A’, ‘B’, ‘C’, …, ‘Z’)

字母表

3

学 号	姓 名	性别	年龄	其 他
99001	张 三	女	17	……
99002	李 四	男	16	……
99003	王 五	女	18	……
99004	周 六	女	17	……
⋮	⋮	⋮	⋮	⋮
99035	刘 末	男	19	……

数据文件

表

1 什么是数据结构

1.1 名词术语

数据　描述客观事物的数字、字符以及一切能够输入到计算机中，并且能够被计算机程序处理的 **符号的集合**。

数据元素　数据这个集合中的一个一个的元素。

数据对象　具有相同特性的数据元素的 **集合**。

结 构　数据元素之间具有的关系。

1.2 数据结构的定义

1. 数据元素之间的联系称之为 结构，**数据结构** 就是具有结构的数据元素的集合。

2. **数据结构** 是一个二元组

Data-Structure=(D, R)

其中，D是数据元素的有限集合，R是D上的关系的集合。

某一数据对象

数据结构包括：

- 数据的逻辑结构
- 数据的存储（物理）结构
- 数据的操作（算法）

逻辑结构

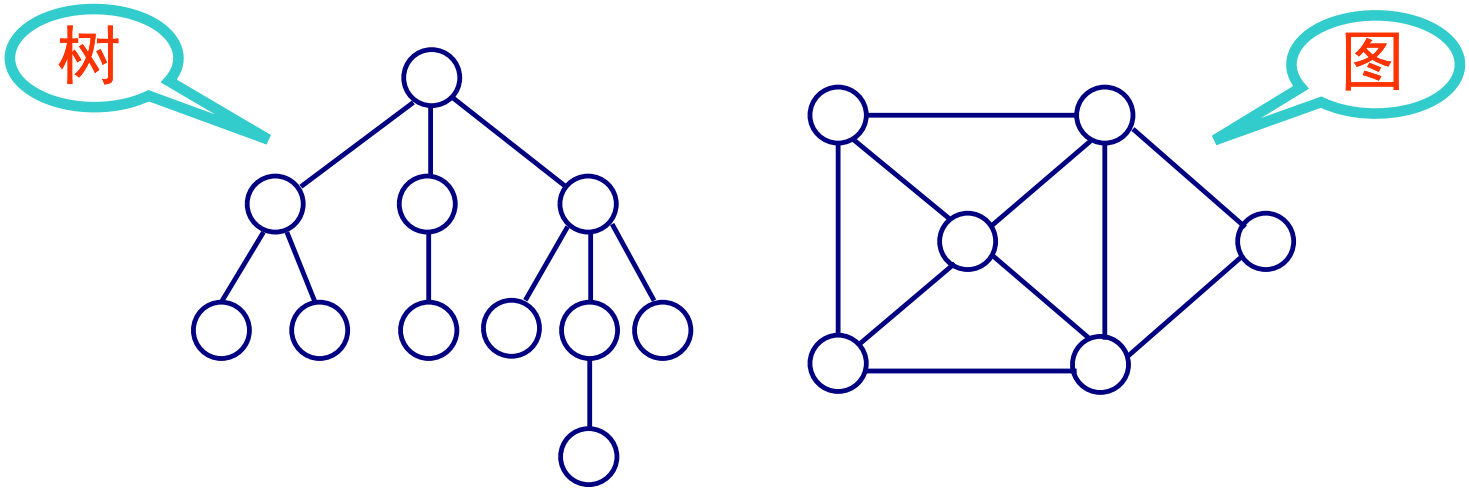
数据元素之间具有的逻辑关系(结构)

线性结构

如线性表、栈、队列、串、文件等

非线性结构

如树、二叉树、图、集合等

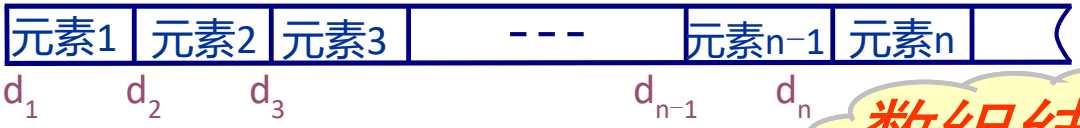


存储结构

具有某种逻辑结构的数据在计算机中的存储方式(存储映象)

1. 顺序（sequential）存储结构

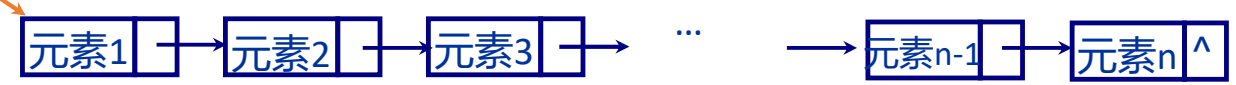
用一组地址连续的存储单元依次存放数据元素，数据元素之间的逻辑关系通过元素的地址直接反映。也就是说数据间的逻辑关系与物理关系是一致的。



数组结构！

2. 链式（linked）存储结构

用一组地址任意的存储单元依次存放数据元素，数据元素之间的逻辑关系通过指针间接地反映。在这种结构中，每个数据节点由两部分组成，一部分是数据本身（数据字段）；另一部分是指针，用于存放后续结构的地址（指针字段）



链表结构！

3. 索引（indexing）存储结构

构造原理

利用数据元素的索引关系来确定数据元素的存储位置，由数据元素本身与索引表两部分组成。

特点

诸如查找、插入和删除等操作的时间效率较高，但存储空间开销较大。

4. 散列（hashing）存储结构

以后详细讨论

构造原理

通过事先准备好的散列函数关系与处理冲突的方法来确定数据元素的存储位置。

特点

诸如查找、插入和删除等操作的时间效率较高，主要缺点是确定好的散列函数比较困难。

例

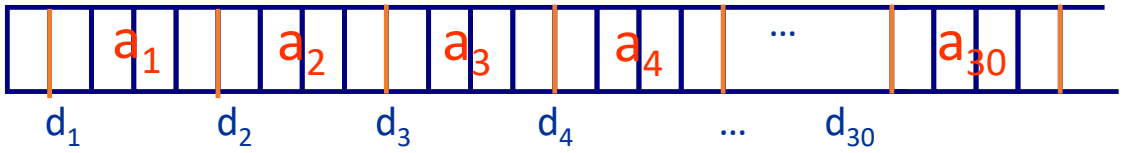
	姓 名	性别	民族	年龄	其 他
a_1	刘晓光	男	汉	16	…
a_2	马广生	男	回	17	…
a_3	王 民	男	壮	19	…
:	⋮	⋮	⋮	⋮	⋮
a_{30}	张淑华	女	汉	24	…

逻辑结构: 线性结构（线性表）

存储结构:

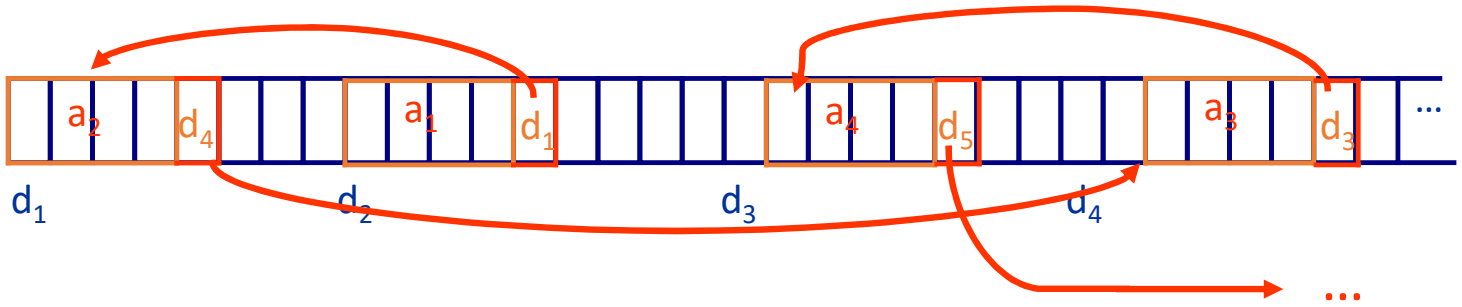
数据元素之间的逻辑关系通过地址直接反映

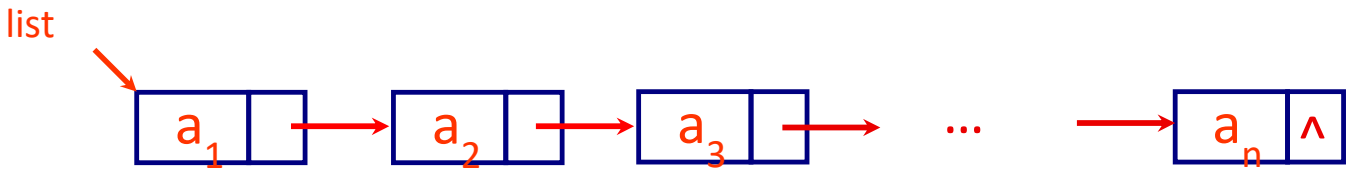
1. 顺序存储结构



用一片地址任意的存储空间！

2. 链式存储结构





链表！

操作

对具有某种逻辑结构的数据所实施的一系列统操作（算法）

基本操作:

- 构造：构造具有某种逻辑结构的数据集，如构造一个线性表、链表、树和图等
- 检索：在已有数据集中查找某一指定元素
- 插入：在已有数据集中插入一指定元素
- 删除：在已有数据集中插入一指定元素
- 排序：对一数据集中元素按照某一顺序进行排列
- 遍历：访问数据集中所有元素
- ...

1.3 数据结构课程研究的主要内容

逻辑结构

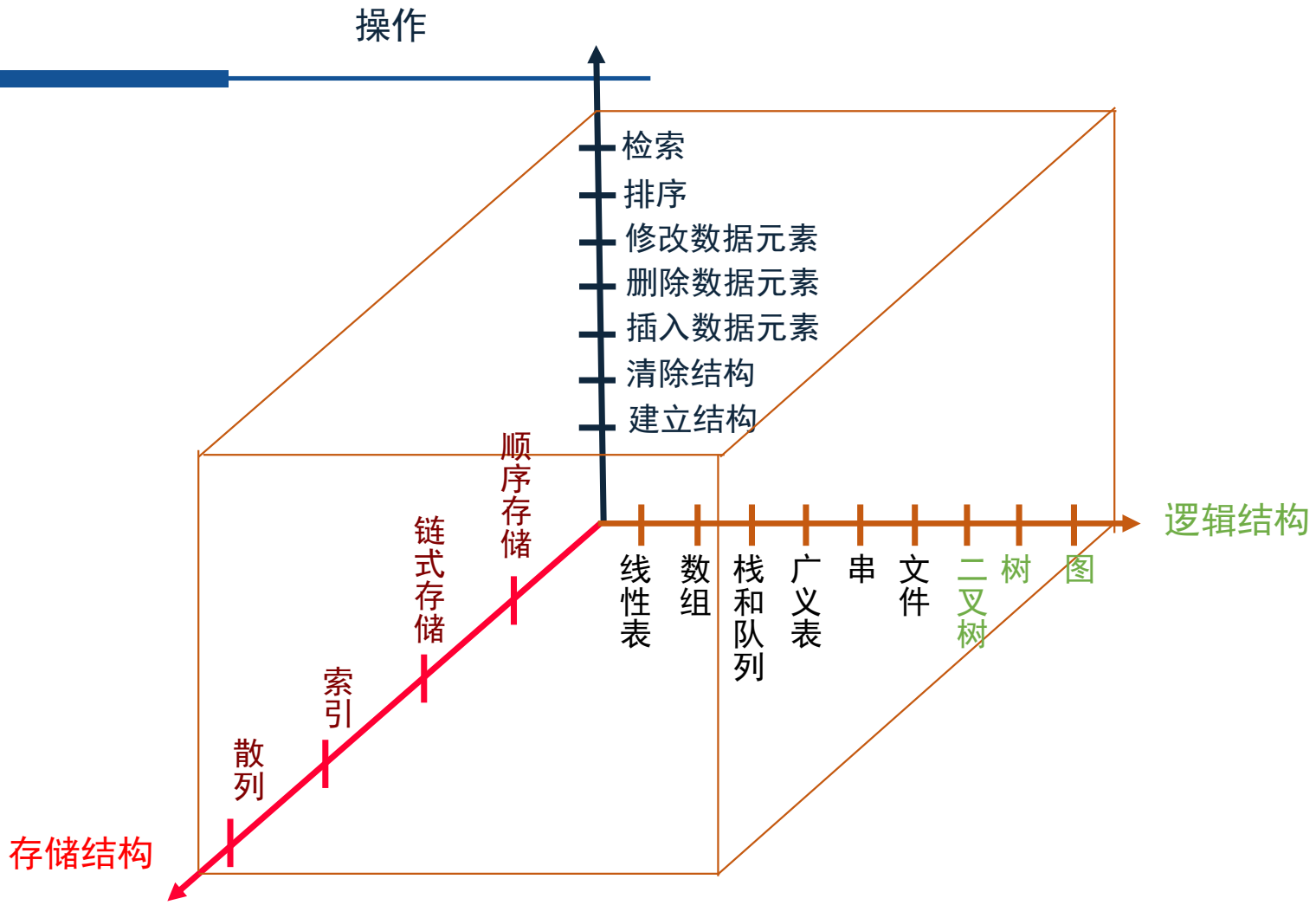
1. 研究数据元素之间的客观联系。
2. 研究具有某种逻辑关系的数据在计算机存储器内的存储方式。

存储结构

3. 研究如何在数据的各种关系（逻辑的和物理的）的基础上对数据实施一系列有效的基本操作。

算法

逻辑结构 + 存储结构 + 算法


操作
检索
排序
修改数据元素
删除数据元素
插入数据元素
清除结构
建立结构
逻辑结构
线性表
数组
栈和队列
广义表
串
文件
二叉树
树
图
存储结构
顺序存储
链式存储
索引
散列

2 算法及其描述

2.1 算法及其性质

1.算法的定义

(1) 算法是用来解决某个特定问题的指令的集合。

(2) 算法是由人们组织起来准备加以实施的一系列有限的基本步骤。

(3) 算法是一组解决问题的清晰指令，它能够对符合一定规范的输入，在有限的时间内获得所需要的输出。

2.算法的性质

一个完整的算法应该具有下面五个基本特性：

输入 由算法的外部提供n≥0个有限量作为算法的输入。

输出 由算法的内部提供n>0个有限量作为算法的输出。

有穷性 算法必须在有限的步骤内能够结束。

确定性 组成算法的每一条指令必须有清晰明确的含义。

有效性 算法的每一条指令必须具有可执行性。

2.2 算法的描述

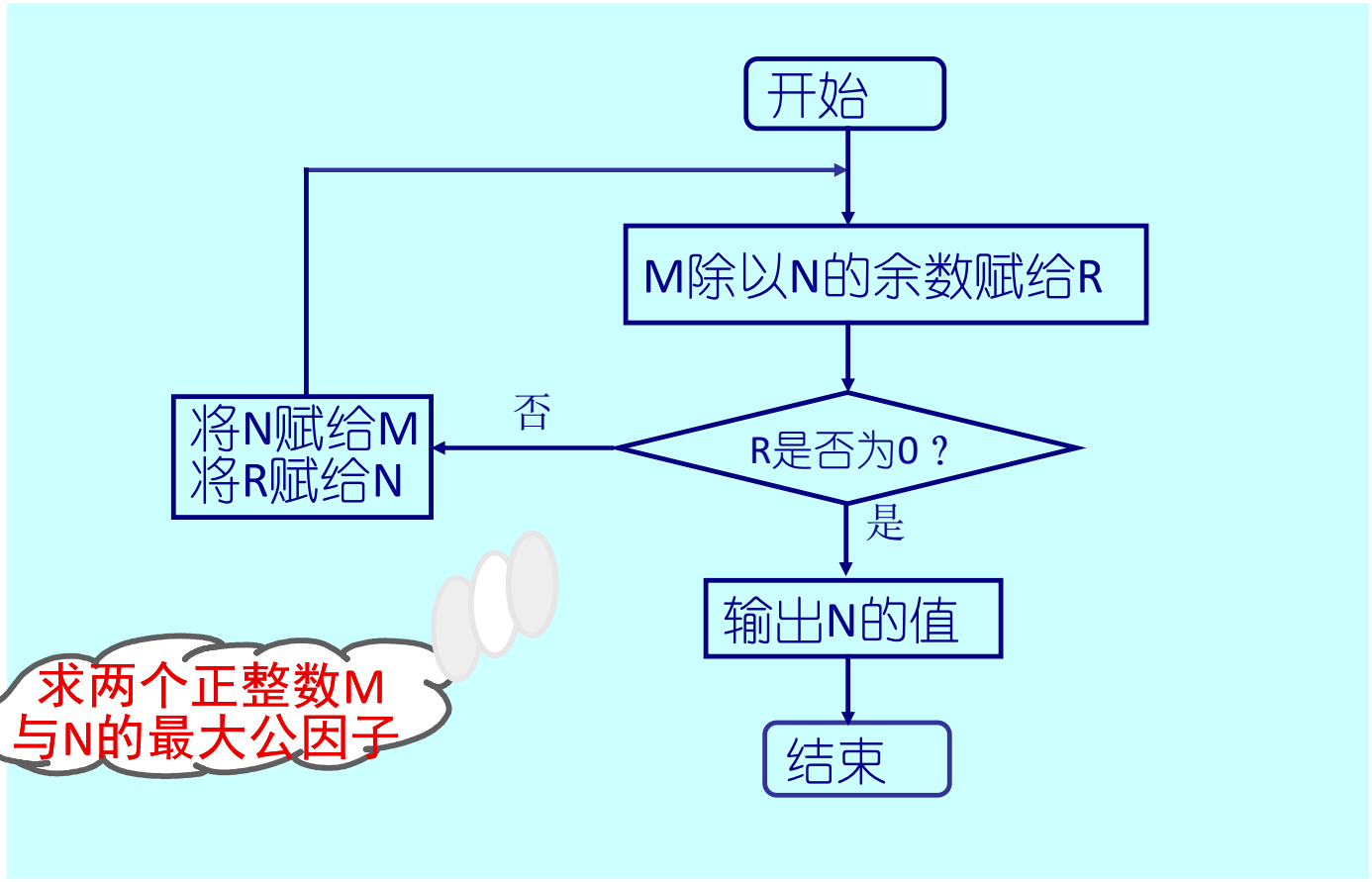
例

1. 采用自然语言来描述

问题 求两个正整数M与N的最大公因子。

(1) M除以N，将余数赋给中间变量R；
(2) 判断余数R是否等于零？
  a) 若R等于零，求得的最大公因子为当前N的值，算法到此结束。
  b) 若R不等于零，则将N赋给M，将R赋给N，重复步骤(1)和(2)。

2. 采用程序流程图的形式来描述



3. 用一种既脱离某种具体的程序设计语言，又具有各种程序设计语言的共同特点的形式化语言来描述（类语言）。

类Pascal语言　　类C语言

SPARKS语言（一种类Pascal语言）

```
        void
        Unweighted( Table T )  /* Assume T is initialized */
        {
            int CurrDist;
            Vertex V, W;

/* 1*/      for( CurrDist = 0; CurrDist < NumVertex; CurrDist++ )
/* 2*/          for each vertex V
/* 3*/              if ( !T[ V ].Known && T[ V ].Dist == CurrDist )
                    {
/* 4*/                  T[ V ].Known = True;
/* 5*/                  for each W adjacent to V
/* 6*/                      if( T[ W ].Dist == Infinity )
                            {
/* 7*/                          T[ W ].Dist = CurrDist + 1;
/* 8*/                          T[ W ].Path = V;
                            }
                    }
        }
```

Dijkstra算法伪代码（类C）

4. 利用某种具体程序语言来描述

C、 C++、 Java 、Python …

```
int comfactor( int m, int n )
{
        int  r;
        while( 1 ){
             r = m  %  n;
             if(r == 0)
                  return  n;
             m = n;
             n = r;
        }
}
```

用C语言描述的求两个正整数最大公因子的算法（C函数）

3 算法分析

目的：改进算法的质量。

前提：算法必须正确。

算法分析 是指对算法质量（效率）优劣的评价。

时空效率高的算法才是一个好的算法，
它常常是不懈努力和反复修正的结果

除正确性外，通常从三个方面分析一个算法：

- 依据算法编写的程序在计算机中运行时间多少的度量，称之为 **时间复杂度**。

  反映算法运行的快慢

- 依据算法编写的程序(运行时)在计算机中占存储空间多少的度量，称之为 **空间复杂度**。

  反映算法需要的额外空间的多少

- 其他方面。如算法的可读性、可移植性以及易测试性的好坏。

时间复杂度

一个程序在计算机中运行时间的多少与诸多因素有关，其中主要有：

1. 问题的规模。
2. 编译程序功能的强弱以及所产生的机器代码质量的优劣。
3. 机器执行一条指令的时间长短。
4. 程序中那些基本语句的执行次数。

几乎所有算法的时间效率都与问题的规模有关

相关

重点

对算法运行时间贡献最大的语句

频度统计法

以语句执行的次数的多少作为算法的时间度量的分析方法称为***频度统计法***。

一条**语句的频度**是指该语句被执行的次数，而整个**算法的频度**是指算法中所有语句的频度之和。

例

求两个n阶矩阵的乘积

```
#define  M  1000
void MATRIX(int A[ ][M],int B[ ][M],int C[ ][M],int n)
{
    int i, j, k;
    for(i=0;i<n;i++)
        for(j=0;j<n;j++){
            C[i][j]=0;                          /* 执行n²   */
            for(k=0;k<n;k++)
                C[i][j]=C[i][j]+A[i][k]*B[k][j];   /* 执行n³   */
        }
}
```

语句频度

算法的频度：

$$t(n) = n^2 + n^3$$

时间复杂度

假设算法A的语句频度为$2n^2$，算法B的频度为3n+1，算法C的语句频度为$2n^2+3n+1$，则有下表：

次数	算法 A（$2n^2$）	算法 B（3n+1）	算法 C（$2n^2+3n+1$）
n = 1	2	4	6
n = 2	8	7	15
n = 5	50	16	66
n = 10	200	31	231
n = 100	20 000	301	20 301
n = 1,000	2 000 000	3 001	2 003 001
n = 10,000	200 000 000	30 001	200 030 001
n = 100,000	20 000 000 000	300 001	20 000 300 001
n = 1,000,000	2 000 000 000 000	3 000 001	200 000 3000 001

从中可以看出：

判断一个算法的效率时，算式中的常数和其它次要项通常可以忽略，而更应该关注主要项（最高阶项）的阶数。

关于符号O（Order）的定义

时间复杂度的大O表示法

当且仅当存在正整数c和n0，使得t(n)≤cg(n)对所有的n≥n0成立，则称函数t(n)与g(n)同阶，或者说，t(n)与g(n)同一个数量级，记作

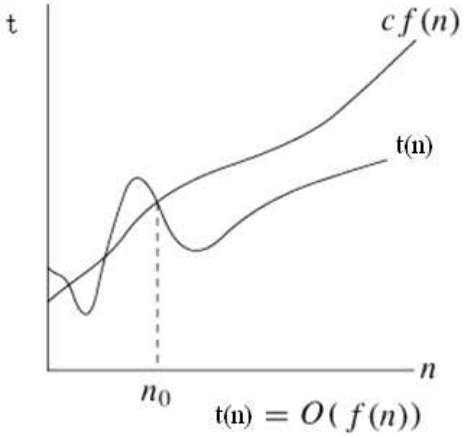
$$t(n)=O(g(n))$$

称上式为算法的**渐近时间复杂度**，或称该算法的时间复杂度为O(g(n))。其中，n为问题的规模（大小）的量度。其表示了算法的**相对增长率**。

当问题的输入规模n趋于无穷大时，算法的运行时间表现出固定的增长次数。

$t(n)=n^3+n^2$　　$g(n)=n^3$

称算法的时间复杂度为$O(n^3)$。



算法性能分析相关知识：

- 大O表示法：　O(g(n))表示算法上界
  t(n) ≤cg(n)
- Ω表示法：　Ω(g(n))表示算法下界
  t(n)≥cg(n)
- Θ表示法：　Θ(g(n))表示算法的上下界相同

当且仅当O(g(n)) = Ω(g(n))

计算时间复杂度

大O表示法关注的是问题规模n增长时，算法执行次数的相对增长率。因此，可以只关注算法中基本语句执行频度，不必对算法的每一个步骤都进行详细的分析。

例

```
i=1;                (1)
while (i<=n)        (2)
  i=i*2;            (3)
```

可只关注语句(3)的频度，设为t(n)，则有：

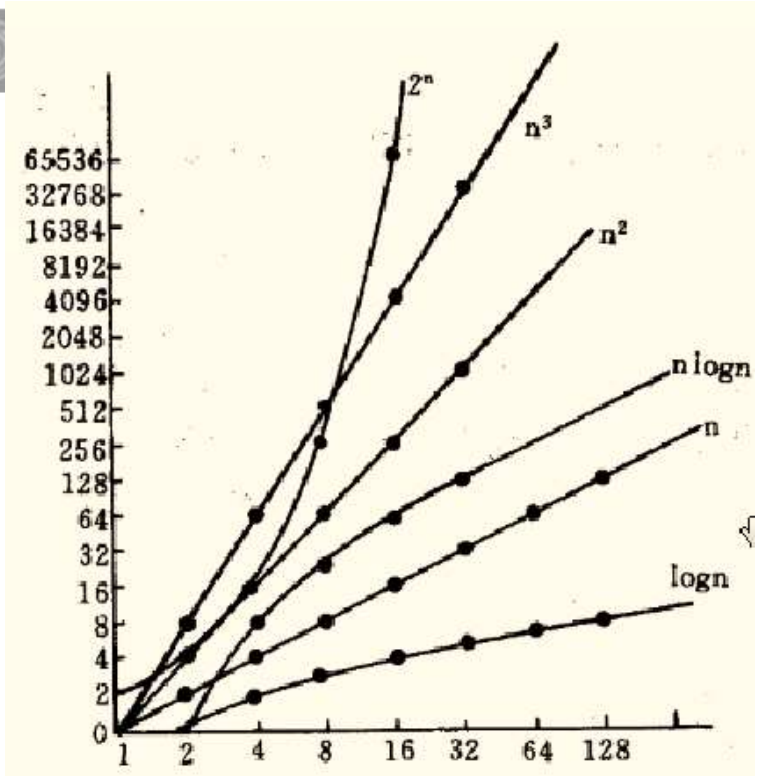
$2^{t(n)}<=n$

即：$t(n)<=\log_2 n$，取最大值$t(n)=\log_2 n$

即：时间复杂度为$O(\log_2 n)$

对于算法分析具有重要意义的常见函数值

n	$\log_2 n$	n	$n\log_2 n$	n^2	n^3	2^n	n!
10	3.3	10^1	3.3×10^1	10^2	10^3	10^3	3.6×10^6
100	6.6	10^2	6.6×10^2	10^4	10^6	1.3×10^{30}	9.3×10^{157}
1000	10	10^3	1.0×10^4	10^6	10^9		
10000	13	10^4	1.3×10^5	10^8	10^{12}		
…	……						



常见的时间复杂度：

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

O(1) —— 表示算法的复杂度为常量，不随问题规模n的大小而改变。

时间复杂度

最坏情况与平均情况

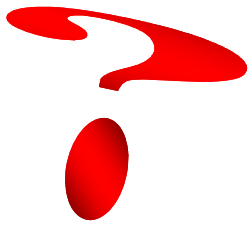
- 最坏情况运行时间是一种保证，那就是运行时间将不会再坏了。在实际应用中，这是一种最重要的需求。如机载软件关键软件中必须要做最坏情况时间分析。
- 平均运行时间是所有情况中最有意义的，因为它是期望的运行时间。
- 对算法的分析，一种是平均时间复杂度；另一种是最坏时间复杂度。一般在没有特殊说明的情况下，都是指最坏时间复杂度。

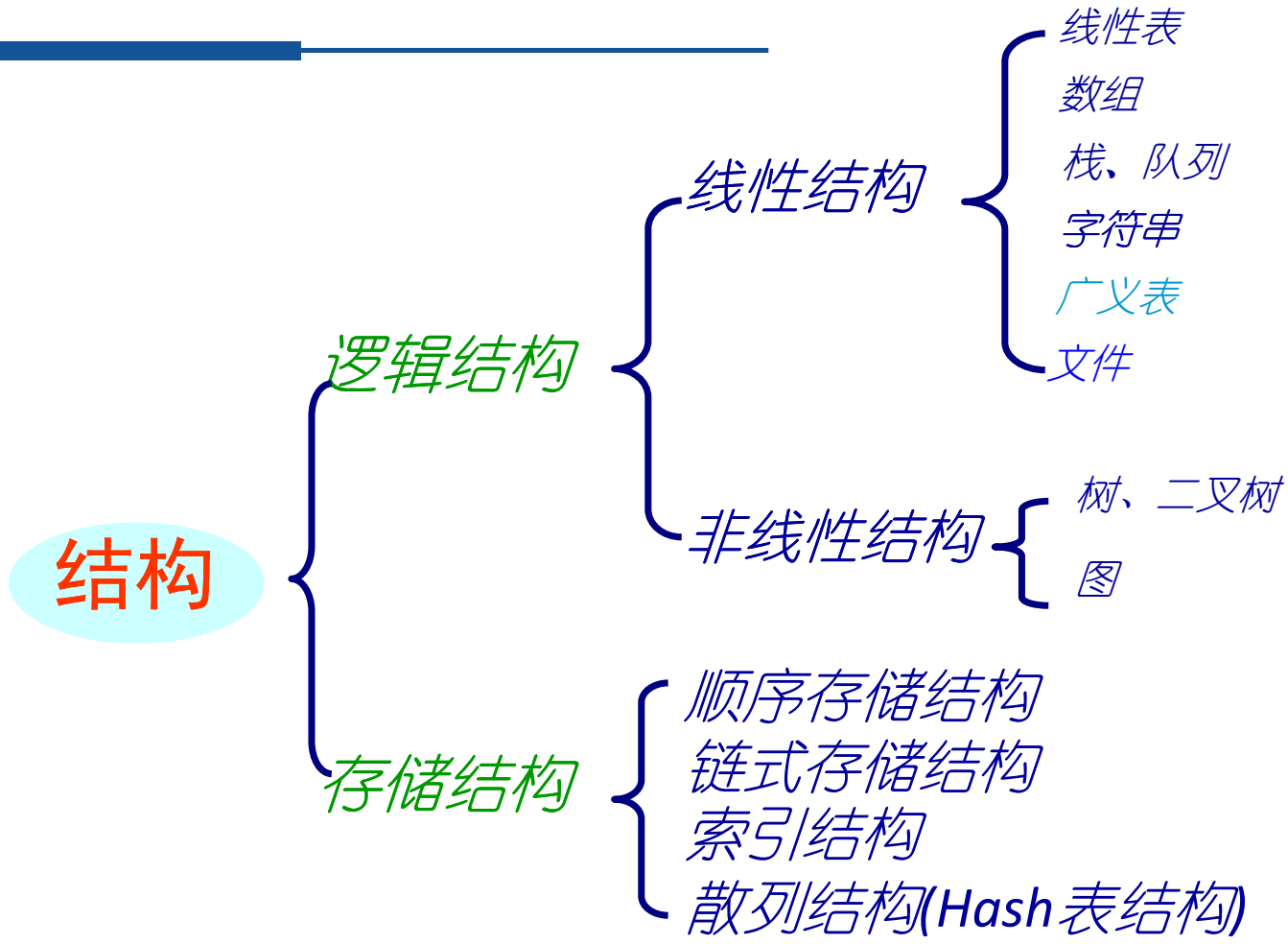
延伸学习* - 算法空间复杂度

- 算法空间复杂度通过计算算法运行时所需的存储空间来实现。

在写程序时，经常会用空间来换时间。**算法空间复杂度**是衡量算法效率的另一个重要指标。

本章内容小结




结构
逻辑结构
线性结构
线性表
数组
栈、队列
字符串
广义表
文件
非线性结构
树、二叉树
图
存储结构
顺序存储结构
链式存储结构
索引结构
散列结构(Hash表结构)

算法

算法的定义

- 用来解决某个特定课题的指令的集合。
- 是由人们组织起来准备加以实施的一系列有限的基本步骤。

算法的基本特征

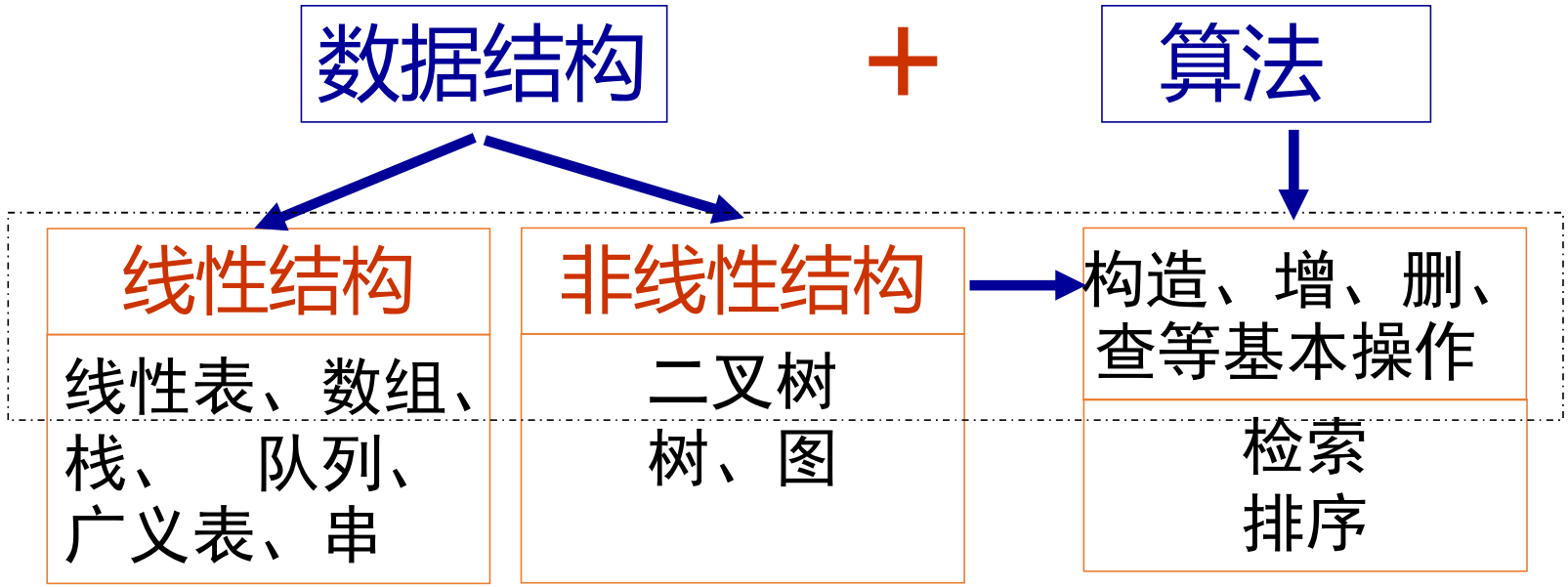
输入、输出、有穷性、确定性、有效性

算法的描述

- 可以采用自然语言或程序设计框图的形式。
- 可以采用某一种自定义的符号语言。
- 可以采用某一种具体的程序设计语言。

算法分析

- 什么是算法分析？
- 算法分析的目的是什么？
- 算法分析的前提是什么？
- 通常从哪几个方面对算法进行分析？


数据结构
+
算法
线性结构
非线性结构
构造、增、删、查等基本操作
线性表、数组、栈、 队列、广义表、串
二叉树
树、图
检索
排序

本讲结束