



数据结构与算法分析

韩英杰

ieyjhan@zzu.edu.cn

计算机与人工智能学院 软件工程系



为什么学？

■ 数据结构

- 是介于数学、计算机软件、硬件三者之间的核心课程；
- 一般程序设计（尤指非数值计算的程序设计）的基础；
- 设计、实现数据库系统、操作系统、编译程序及其它系统程序和大型应用程序的重要基础。

■ 算法（设计与）分析

- 是关于算法的方法论，是计算机软件设计的基础；
- 对培养高质量计算机人才所必需的基本理论与知识、抽象思维能力、逻辑思维能力以及灵活运用算法解决问题的能力有极大帮助；
- 对未来从事计算机系统结构、系统软件及应用软件研究与开发的专业人员必不可少。



2024届校招要求-字节跳动

职位描述

日常实习：面向全体在校生，为符合岗位要求的同学提供为期3个月及以上的项目实践机会。

团队介绍：朝夕光年是面向全球用户与开发者的游戏研发与发行业务品牌。以“激发创造，丰富生活”为使命，朝夕光年致力于服务全球玩家，帮助玩家在令人惊叹的虚拟世界中一起玩耍与创造。在这里你将有机会和来自不同国家和地区的同事并肩战斗，和业界的顶尖高手交流学习，亲历一个游戏从创意到上线的全过程。

- 1、参与字节跳动游戏技术中台方向的服务端研发；
- 2、根据业务功能需求和设计方案进行开发，完成代码的编写和调试等工作；
- 3、参与产品需求分析，设计系统技术方案，核心代码开发和系统调优等。

字节跳动游戏服务端

职位要求

- 1、2024届本科及以上学历在读，计算机、通信等相关专业，每周至少实习4天，半年以上；
- 2、扎实的计算机知识，深刻理解计算机原理，有良好的数据结构和算法基础；深入了解Go、Python、Java、C++等至少一门语言；
- 3、热爱编程，责任心强，有较强的学习能力，有强烈的求知欲，具有良好的团队沟通与协作能力；
- 4、较好的产品意识，有中台方向相关经验者优先，喜欢玩游戏及有游戏行业经验者优先。



2024届校招要求-字节跳动

- 1、负责架构设计和开发，为亿级用户提供优质顺畅的信息服务和极致体验；
- 2、参与设计系统技术方案，核心代码开发和系统调优；
- 3、参与制定代码规范、测试规范，建立起开发质量控制方法；
- 4、协助团队攻克各种高并发、数据隔离、系统解耦等方面的技术难关；
- 5、参与各专项技术调研，新技术引入等前瞻项目；
- 6、参与机器学习与异构计算系统研发。

字节跳动后端服务

职位要求

- 1、2024届及以后毕业，本科及以上学历在读，计算机等相关专业优先；
- 2、热爱计算机科学和互联网技术，精通至少一门编程语言，包括但不限于：Java、C、C++、PHP、Python、Go；
- 3、掌握扎实的计算机基础知识，深入理解数据结构、算法和操作系统知识；
- 4、有优秀的逻辑分析能力，能够对业务逻辑进行合理的抽象和拆分；
- 5、有强烈的求知欲，优秀的学习和沟通能力。

职位描述

团队介绍：Gauth是一款爆火全球的AI工具产品，占据了出海+AI两个热门赛道，自上线来一直广受用户欢迎，多次登顶美国在内的全球多国家App Store总榜第一。当下ChatGPT为首的AGI革命正席卷全球，同时给教育、知识行业带来了巨大变革和机遇，我们相信大语言模型可以改变人类过往语言、文化、知识学习的方式，也在积极探索应用层的创新，希望为用户创造不一样的知识学习体验。

- 1、主要负责互联网产品质量保障工作及提效工具开发等；
- 2、深度参与产品研发项目，协同产品和研发团队高质量交付产品；
- 3、日常项目线下测试与线上质量分析；
- 4、参与质量体系规划和建设；
- 5、参与开发效率工具和保证技术项目质量。

职位要求

- 1、2025届获得本科及以上学历，计算机相关专业优先；
- 2、热爱计算机科学和互联网技术，对软件质量保障工作有浓厚兴趣；
- 3、扎实的数据结构和算法基础；熟悉至少一门编程语言，包括但不限于：Java、C、C++、Python、Go、PHP；
- 4、优秀的产品意识，对市场上典型 App 有自己的想法和改进方案；
- 5、快速适应和学习能力，具备较强的解决问题的能力；
- 6、了解常用客户端、服务端开发或测试工具，如自动化框架、压测工具、大数据处理工具者优先。

客户端开发工程师-飞书技术中台

北京 | 正式 | 研发 - 客户端 | 2025届校园招聘 | 职位 ID: A221488

职位描述

团队介绍：飞书技术中台是围绕字节跳动飞书产品矩阵所建设的研发团队，一方面助力业务进行AI、搜索等相关前沿技术探索，同时提供安全、数据、架构等基础服务支撑；另一方面也致力于打造行业领先的开放平台、交付（包含私有化）工具平台。以技术为驱动，提升企业内外部的资源利用率、协作效率和创造力。

- 1、参与iOS端、Android端以及跨平台开发；
- 2、参与 APP 性能、体验优化等相关工作；
- 3、参与客户端基础组件及架构设计，推进研发效率。

职位要求

- 1、2025届获得本科及以上学历，计算机、软件工程等相关专业优先；
- 2、热爱计算机科学和互联网技术，对移动产品有浓厚兴趣；
- 3、具有扎实的数据结构和算法基础，熟悉一种以上编程语言，如：Golang/Java/C++等；
- 4、至少有一个在校/实习/竞赛等期间优秀iOS端、Android端研发的项目；
- 5、具备良好的学习能力和团队合作精神，能够快速适应新环境。

北京 | 正式 | 研发 - 客户端 | 2025届校园招聘 | 职位 ID: A245035

职位描述











团队介绍：专注于探索AI和智能硬件的结合，为用户提供更自然和便捷的交互体验的研发团队，隶属于产品研发与工程架构部。作为负责AI技术应用场景探索的部门，是字节在智能硬件领域提供综合方案研究的核心部门。我们欢迎期待心怀技术理想、不断挑战技术难题的“你”的加入，和顶尖团队一起参与技术攻坚，开启更多可能。

- 1、负责XR系统Framework，为XR设备的特殊需求对Framework进行定制开发；
- 2、负责XR系统的一些通用模块的开发和维护；
- 3、对接算法团队，调度各种XR算法，为上层业务方提供统一和标准的算法数据；
- 4、解决系统的性能/稳定性问题。

职位要求

- 1、2025届获得本科及以上学历，计算机、软件工程等相关专业优先；
- 2、具有良好的思维能力，能高效和准确地分析问题，定位问题，并得出解决问题的有效路径；
- 3、具有良好的自学能力，对未知充满好奇，并乐于付出实践；
- 4、熟练掌握计算机基础，包括不限于操作系统原理，计算机组成原理，数据结构和一些通用算法；
- 5、熟练掌握一门计算机编程语言，C/C++优先，具有良好的编码能力，注重代码的简洁性，扩展性，维护性，追求极致和高效。

2024 年 6 月TIOBE 编程语言排行榜

Jun 2024	Jun 2023	Change	Programming Language		Ratings	Change
1	1			Python	15.39%	+2.93%
2	3	▲		C++	10.03%	-1.33%
3	2	▼		C	9.23%	-3.14%
4	4			Java	8.40%	-2.88%
5	5			C#	6.65%	-0.06%
6	7	▲		JavaScript	3.32%	+0.51%
7	14	▲▲		Go	1.93%	+0.93%
8	9	▲		SQL	1.75%	+0.28%
9	6	▼		Visual Basic	1.66%	-1.67%
10	15	▲▲		Fortran	1.53%	+0.53%

珠海 | 正式 | 研发 - 算法 | 2025届校园招聘 | 职位 ID: A108370

职位描述

团队介绍：Data-电商团队，负责电商创新项目的算法和大数据工作。依托于字节跳动产品，帮助用户发现并获得好物，享受美好生活。在这个团队，我们不仅要通过推荐和搜索算法帮助用户买到感兴趣的好东西，也要通过风控算法和智能平台治理算法去甄别违规行为，保护用户的购物体验；我们还要建设智能客服技术、大规模商品知识图谱来提升各个交易环节的效率；我们也要通过物流和运筹算法，来提高供应链的效率；另外我们还会用人工智能来帮助商家提升经营能力。我们的使命：没有难卖的优价好物，让美好生活触手可及。

- 1、参与电商增长&营销算法优化，包括用户价值建模、个性化消息触达、推荐、智能营销等核心能力建设；
- 2、建立用户全生命周期数据和价值体系，解决电商用户增长中的各种核心痛点和业务问题；
- 3、参与实现推送等个性化消息触达、推荐承接等用户增长引擎的产品和技术方案，提升电商DAU和渗透率；
- 4、负责电商新用户推荐的召回/排序等相关算法优化，提升引流来源承接相关性和新用户推荐的准确性和多样性；
- 5、参与智能营销算法优化，利用Uplift模型和运筹规划等方法提升营销效率，促进电商GMV增长。

职位要求

- 1、2025届获得本科及以上学历，计算机、软件工程等相关专业优先；
- 2、有扎实的算法和数据结构基础，优秀的编码能力；
- 3、学习能力强，对事物保有好奇心，良好的沟通能力和团队协作能力；
- 4、在AI领域顶级会议/期刊上发表过论文，或参加ACM/机器学习类竞赛并获得过好名次的优先；
- 5、有推荐、广告、用户增长、智能营销等相关领域研究或者项目实践，有LTV预估、Uplift、运筹规划、序列建模、多场景建模优化等经验的优先。

大模型算法研究员 (Code AI方向) - 国际化短视频

上海 | 正式 | 研发 - 算法 | 2025届校园招聘 | 职位 ID: A92472

2025届校招要求-字节跳动

职位描述

团队介绍：国际化短视频产品研发团队，旨在实现字节跳动国际化短视频业务的研发工作，搭建及维护业界领先的产品。加入我们，你能接触到包括用户增长、社交、直播、电商C端、内容创造、内容消费等核心业务场景，支持产品在全球赛道上高速发展；也能接触到包括服务架构、基础技术等方向上的技术挑战，保障业务持续高质量、高效率、且安全地为用户服务；同时还能为不同业务场景提供全面的技术解决方案，优化各项产品指标及用户体验。

在这里，有大牛带队与大家一同不断探索前沿，突破想象空间。在这里，你的每一行代码都将服务亿万用户。在这里，团队专业且纯粹，合作氛围平等且轻松。目前在北京，上海，杭州、广州、深圳分别开放多个岗位机会。

- 1、推动大语言模型代码方向的核心技术建设，持续优化大模型代码理解、推理与生成能力；
- 2、致力于提升真实生产环境代码库的代码理解推理与生成能力，提升国际化短视频服务代码性能和隐私合规能力；
- 3、探索适合实际业务生产环境的Code Agent能力，提升国际化短视频研发效率。

职位要求

- 1、2025届获得硕士及以上学位，计算机相关专业优先；
- 2、在机器学习（ML）、计算机视觉（CV）、自然语言处理（NLP）等领域发布过论文者优先（NeurIPS、ICML、ICLR、CVPR、ACL、OSDI、NSDI、SC和SigMOD等），对LLM有实际开发经验者优先；
- 3、具备出色的编程能力、数据结构和算法技能，熟练掌握C/C++或Python编程语言，在ACM/ICPC、NOI/IOI、Top Coder、Kaggle等比赛中获奖者优先；
- 4、拥有机器学习领域的研究经历，特别是在大规模语言模型（LLMs）和生成式人工智能方面；
- 5、出色的问题分析和解决能力，并对解决具有挑战性的问题充满热情；
- 6、对技术充满激情，良好的沟通能力和团队合作精神。

多模态大模型算法研究员（图文方向）-今日头条

北京 | 正式 | 研发 - 算法 | 2025届校园招聘 | 职位 ID: A221569

2025届校招要求-字节跳动

职位描述

团队介绍：产品覆盖今日头条、头条极速版等各类产品，为上亿用户持续提供优质的资讯、视频等服务。我们通过建立良好的内容生态，鼓励优质的原创内容，为创作者提供优质的服务和创作体验，促进创作和交流，同时我们致力于通过尽可能丰富的内容体裁和尽可能多的分发方式，连接人与信息，丰富大家的精神生活，让人们看到更大的世界。

- 1、负责多模态理解与生成大模型的研究与落地，跟进多模态前沿领域进展；
- 2、优化多模态基础模型训练和模型精调方法，解决多模态匹配、多模态生成、多模态理解等问题；
- 3、推动多模态技术在今日头条各项业务落地，或驱动新应用产生。

职位要求

- 1、2025届获得本科及以上学历，人工智能、机器学习、计算机视觉、自然语言处理、计算机、自动化等相关专业优先；
- 2、有多模态研究或者实践经验优先；
- 3、具备扎实的传统机器学习算法理论基础，熟悉常见AI生成模型，包括扩散模型、自回归模型等，了解业界最前沿进展，在顶级会议（ACL、NIPS、ICML、CVPR等）上发表论文者优先；
- 4、优秀的代码能力、数据结构和基础算法功底，熟悉Python，熟练掌握深度学习框架TensorFlow或者PyTorch；
- 5、出色的问题分析和解决能力，能深入解决大模型训练和应用存在的问题，有自主探索解决方案的能力；
- 6、良好的沟通协作能力，能和团队一起探索新技术，推进技术进步；
- 7、充满好奇心与想象力，有优秀的大局观与目标价值导向，认同技术推动产品革新。



如何利用计算机解决实际问题?

- 需经过以下三步骤:
 - ① 从问题中抽象出一个恰当的数学模型;
 - ② 设计一个解此数学模型的算法;
 - ③ 编程调试, 得到最终答案。

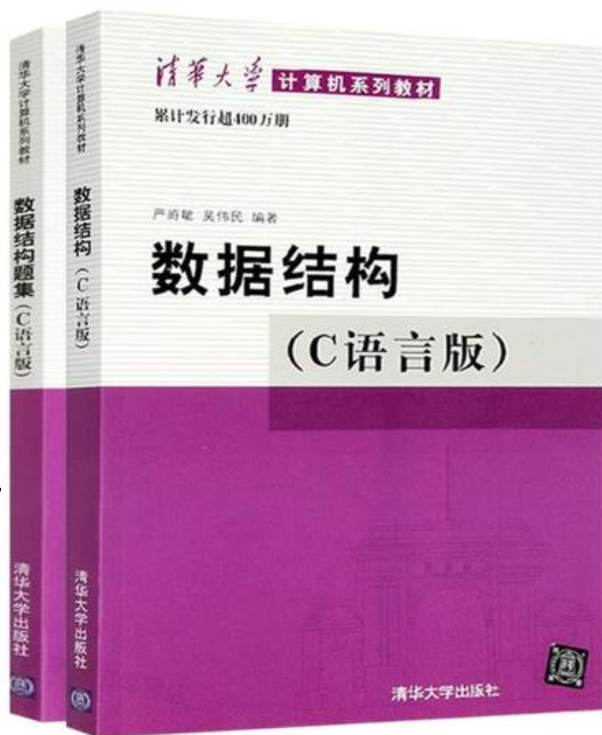


Niklaus Wirth:

Data Structures + Algorithms = Programs

- 数据结构: 解决问题的数学模型
- 算法: 解决问题的策略
- 程序: 为计算机解决问题编制的一组指令集

例1 图书馆书目检索自动化问题



数学模型

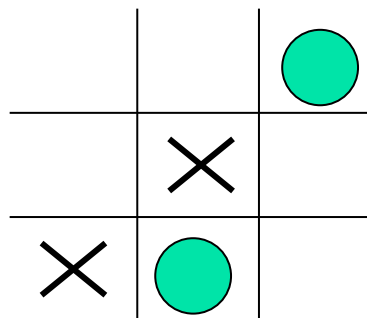
一对一的线性结构

登录号	书名	作者	出版单位	ISBN	出版时间
N.3.73.762	数据结构	严蔚敏 吴伟民	清华大学出版社	9787302147510	2018.8

例2 人机对弈问题

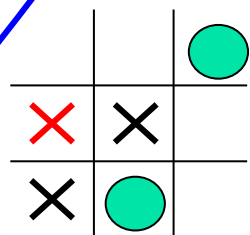
先手: ×

后手: ●

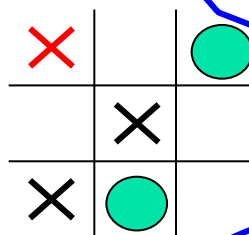


数学模型

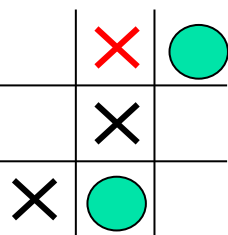
一对多的树形结构



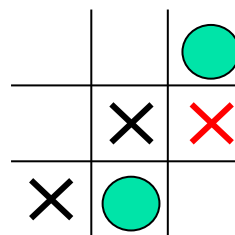
(a)



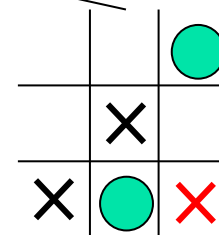
(b)



(c)



(d)



(e)

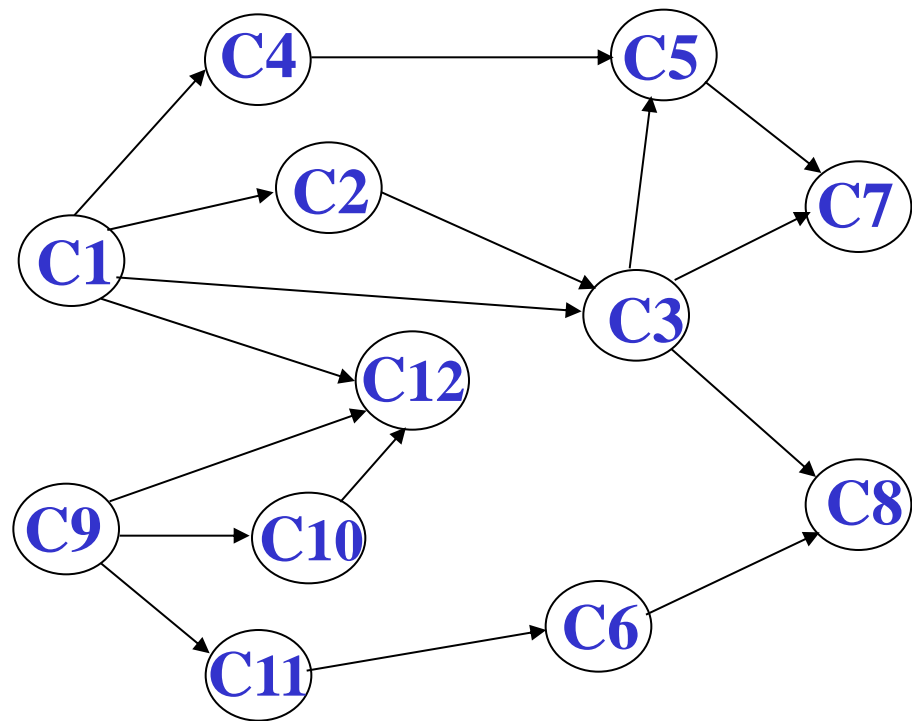
井字棋对弈树

例3 计算机类开设课程问题

数学模型

多对多的图形结构

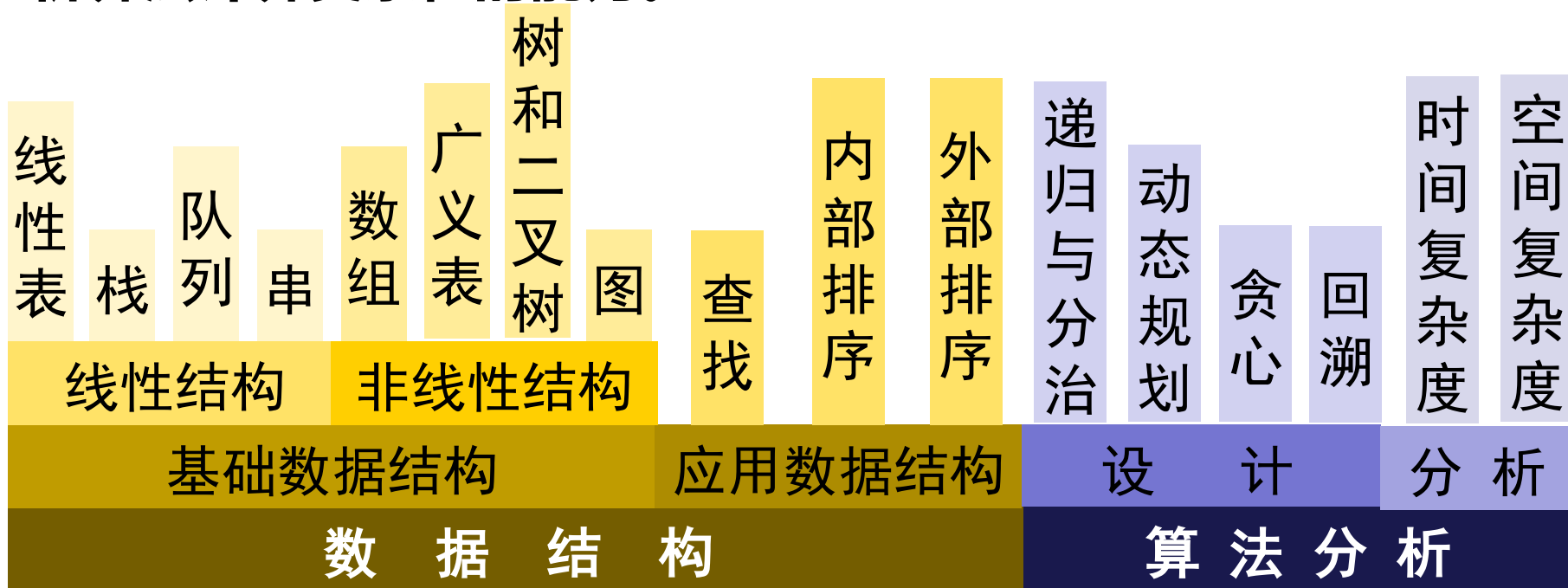
课程编号	课程名称	先行课程编号
C1	程序设计基础	无
C2	离散数学	C1
C3	数据结构	C1, C2
C4	汇编语言	C1
C5	算法设计与分析	C3, C4
C6	计算机组成原理	C11
C7	编译原理	C5, C3
C8	操作系统	C3, C6
C9	高等数学	无
C10	线性代数	C9
C11	大学物理	C9
C12	数值分析	C9, C10, C1



计算机类课程开设先后关系图

学什么?

数据结构主要研究**非数值计算**的程序设计中计算机的操作**对象**及其之间的**关系**和**操作**。算法分析主要研究如何设计算法以及如何**定量分析**一个算法需要的计算时间和存储空间，培养正确分析算法计算复杂性的能力。





怎样学？

- **动脑：**牢记基本概念和经典算法。
- **动手：**掌握一门编程语言，多做题，注意画图。
 - 力扣leetcode: <https://leetcode.cn/>
 - 刷题 A : <https://pintia.cn/>
- **善用资源：**《数据结构与算法图解》、《剑指offer》、慕课、B站等。
- **宜：**联系实际，富于联想，总结共性和特性，归类，对比……。
- **忌：**求偏、求难、重算法轻概念。
- **三阶段：**模仿->总结->创新



学习目标-计算机学科专业基础考试408

- **掌握数据结构的基本概念、基本原理和基本方法；**
- **掌握数据的逻辑结构、存储结构及基本操作的实现，能够对算法进行基本的时间、空间复杂度分析；**
- **能够运用数据结构的基本原理和方法进行问题的分析与求解，具备使用C或C++语言设计与实现算法的能力。**

(合理地组织数据，恰当地表示数据，高效地处理数据)



课程考核

- **平时成绩：**
 - **平时作业：PTA**
 - **考勤**

综合成绩 = 平时成绩 * 30% + 考试成绩 * 70%



1.2 基本概念和术语(1)

- **数据(Data)**: 信息的载体, 是描述客观事物的数、字符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
 - 一个数据元素可由若干个数据项组成;
 - 数据项是数据不可分割的最小单位。
- **数据对象(Data Object)**: 数据的子集。具有相同性质的数据元素的集合。
 - 例: 整数集合 $N = \{0, \pm 1, \pm 2, \dots\}$



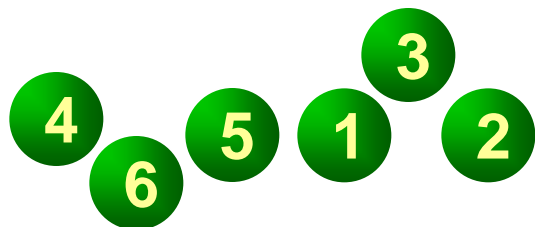
1.2 基本概念和术语(2)

- **数据结构（逻辑结构）**：相互间存在一种或多种**特定关系**的数据元素集合。
 - **结构**：数据元素相互之间的关系
 - **特定关系**：

特定关系	数据结构	示例
没有关系	集合	正整数
一对一关系	线性表	图书管理
一对多关系	树	棋局对弈树
多对多关系	图（网）	课程开设先后关系图

四类基本逻辑结构关系图

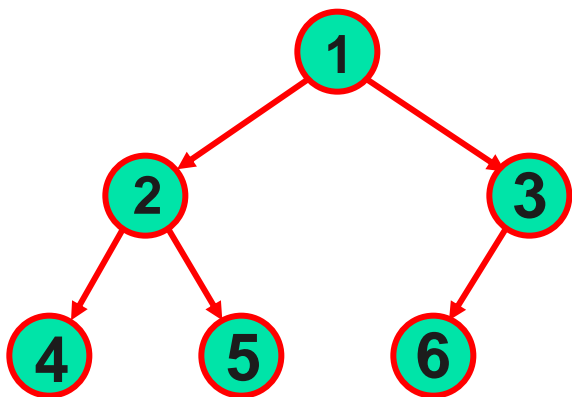
集合结构



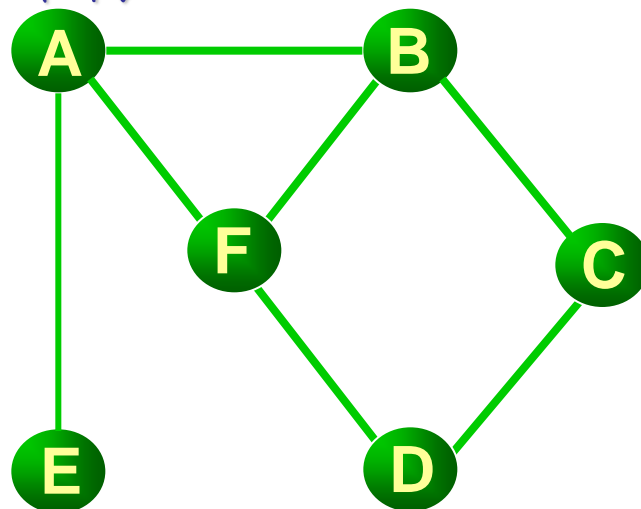
线性结构

姓名	学号	性别	年龄	班级	健康状况
王小林	790631	男	18	计科05	健康
陈红	790632	女	20	计科06	一般
刘建平	790633	男	21	计科07	健康
张立立	790634	男	17	计科08	神经衰弱

树形结构



图形结构



1.2 基本概念和术语(3)

- **数据结构(形式化定义): $DS=(D,S)$**

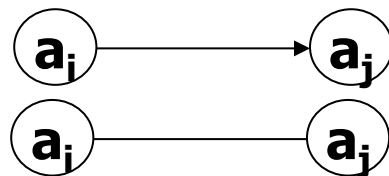
- **D: 数据元素的有限集合, 如:**

$$D = \{a_1, a_2, \dots, a_i, a_{i+1}, \dots, a_n\};$$

- **S: 定义在D上的关系的有限集合。**

- 若 $a_i R a_j$, 则 $\langle a_i, a_j \rangle \in S$

- 若 $a_i R a_j$, 且 $a_j R a_i$, 则 $(a_i, a_j) \in S$



- **例1 复数的数据结构 $COMPLEX = (C, R)$**

- **C: 含有两个实数的集合 $\{C_1, C_2\}$;**
- **R: 定义在C上的关系 $\{\langle C_1, C_2 \rangle\}$ 。**

- **例2 $T=(K,R)$, 其中, $K=\{1,2,3,4,5,6\}$,
 $R=\{\langle 1,2 \rangle, \langle 1,3 \rangle, \langle 2,4 \rangle, \langle 2,5 \rangle, \langle 3,6 \rangle\}$, 画出T。**

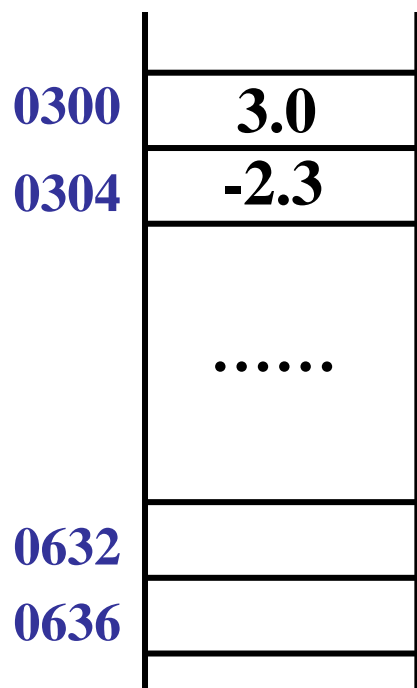


1.2 基本概念和术语(4)

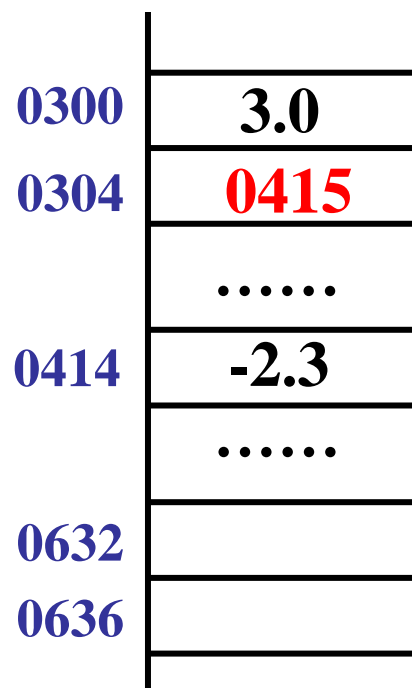
- **数据的物理结构(存储结构):数据逻辑结构在计算机中的表示(映象):**
 - 数据元素的表示
 - 数据元素之间关系的表示
 - **顺序映象:** 借助元素在存储器的相对位置表示数据元素之间的逻辑关系, 对应于顺序存储。
 - **非顺序映象:** 利用指示元素存储地址的指针表示数据元素间的逻辑关系。
 - 链式存储结构(Linked Lists)
 - 索引树(Indexed Trees)
 - 哈希表(Hash Tables)

例 复数存储结构示意图

$$Z=3.0-2.3i$$



顺序存储结构示意图

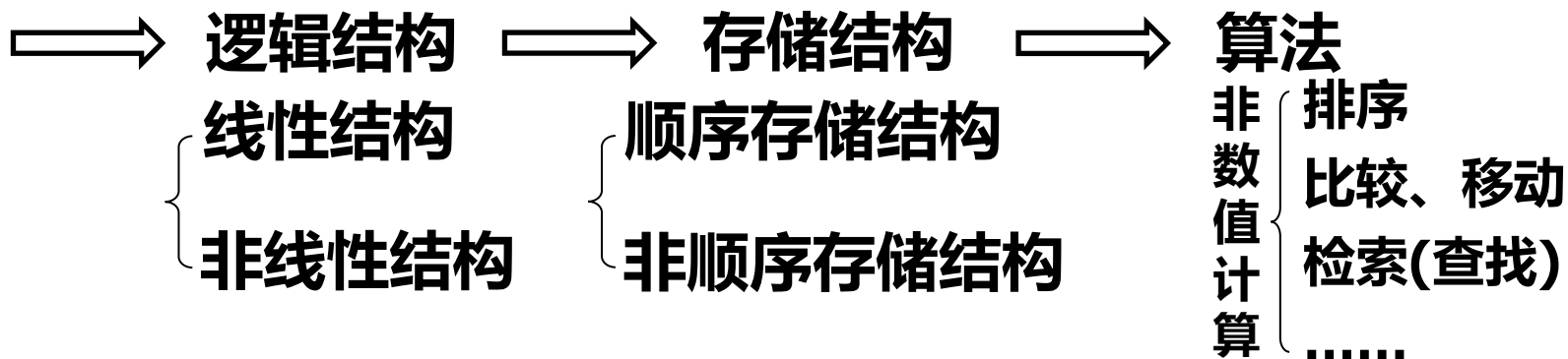


链式存储结构示意图

学习目标

- 分析、把握问题中数据的特性，学习合理地组织数据，恰当地表示数据，高效地处理数据；
- 为问题涉及的数据选择恰当的逻辑结构、逻辑结构的存储映象（物理结构）及相关操作及实现（算法）；
- 提高算法设计的水平。

问题涉及
的数据



任意一个算法的设计取决于选定的逻辑结构
算法的实现依赖于采用的物理结构



1.2 基本概念和术语(5)

- **数据类型**：一组性质相同的值的集合以及定义于该值集上的一组操作的总称。
 - **例**：C语言中整型变量
 - **值**：定义在某区间上的整数
 - **操作**：加、减、乘、除、取模
- **数据类型的分类**（按值的不同特性）
 - **原子类型**：值不可再分，如C的五种基本数据类型：`char`, `int`, `float`, `double`, `void`
 - **结构类型**：值由若干个成分按某种结构组成。



1.2 基本概念和术语(6)

- **抽象数据类型**: 一个数据模型及定义在该模型上的一组操作。 $DS=(D, S, P)$:
 - **D**: 数据对象
 - **S**: D上的关系集
 - **P**: 对D的基本操作集
 - 构造性操作: 插入、删除、修改
 - 非构造性操作: 查找、排序
- **抽象数据类型的分类** (按值的不同特性)
 - **原子类型**: 值不可再分
 - **结构类型**:
 - **固定聚合类型**: 值由确定数目的成分组成。
 - **可变聚合类型**: 组成值的成分数目不确定。

抽象数据类型的描述(本书适用)

ADT 抽象数据类型名 {
 数据对象：〈数据对象的定义〉
 数据关系：〈数据关系的定义〉
 基本操作：〈基本操作的定义〉

} ADT 抽象数据类型名

基本操作的定义格式如下：

 基本操作名(参数表)

 初始条件：〈初始条件描述〉

 操作结果：〈操作结果描述〉

参数表有两种参数：

- 赋值参数：只为操作提供输入值；
- 引用参数：以&打头，可提供输入值，还将返回操作结果。

初始条件：描述操作执行前数据结构和参数应满足的条件。若不满足，操作失败，返回相应出错信息。

操作结果：描述操作正常完成之后，数据结构的变化和应返回的结果。若初始条件为空，则省略之。



例 抽象数据类型矩阵

ADT Matrix{

数据对象: $D=\{a_{i,j} \mid i=1,2,\dots,m;j=1,2,\dots,n;a_{i,j}\in \text{ElemSet}\}$

数据关系: $R=\{\text{Row},\text{Col}\}$

$\text{Row}=\{ \langle a_{i,j},a_{i,j+1} \rangle \mid 1\leq i\leq m,1\leq j\leq n-1 \}$

$\text{Col}=\{ \langle a_{i,j},a_{i+1,j} \rangle \mid 1\leq i\leq m-1,1\leq j\leq n \}$

基本操作:

CreateMatrix(&M)

DestroyMatrix(&M)

PrintMatrix(&M)

AddMatrix(M,N,&Q)

SubMatrix(M,N,&Q)

MultMatrix(M,N,&Q)

TransposeMatrix(M,&T)

} ADT Matrix



1.3 抽象数据类型的表示与实现

类 C 语言的语法规则(P9-11)

- 1、预定义常量和类型
- 2、数据结构的表示和数据元素类型的定义
- 3、基本操作的算法用函数描述
- 4、算法描述中可以使用的赋值语句形式
- 5、算法描述中可以使用的选择结构语句形式
- 6、算法描述中可以使用的循环结构语句形式
- 7、描述算法中可以使用的结束语句形式
- 8、算法描述中可以使用的输入输出语句形式
- 9、算法描述中可以使用的注释格式
- 10、算法描述中可以使用的基本函数
- 11、算法描述中可以使用的逻辑运算的约定



类 C 语言的语法规则示例

例1

```
typedef int Status;  
void example_1( )  
{  
    Status a[10];  
    a[0..9]=0;  
}
```

例2

```
#define OK 1  
typedef int Status;  
Status example_2(int x,int y)  
{    if(x>y)  
        x<->y;  
        return OK;  
}
```




类 C 语言的语法规则示例（续）

例3

```
typedef struct student{  
    char id[5];  
    char name[11];  
    int age;  
    int math;  
    int eng;  
    int ds;  
    int os;  
}student;
```

```
void example_3( )  
{  
    student s;  
    s=(", ",0,0,0,0,0);  
}
```



附加内容：算法的描述

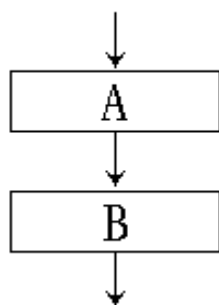
- 步骤法
 - 程序流程图
 - N-S图
 - 伪码
-
- 例：顺序查找data[1..n]中是否有key存在，如果存在，返回其位序；如不存在，返回0。



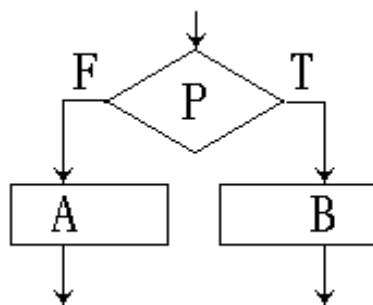
步骤法

- 步1：输入数据序列data和欲查找值key；**
- 步2：从序列中的最后一个元素开始查找。**
- 步3：若该元素值不等于key，查找前一项。**
- 步4：若该元素值等于key，查找成功，返回key在序列中的位置，去第6步。**
- 步5：如果数据全部查找过，但未能找到key，查找失败，返回0。**
- 步6：结束。**

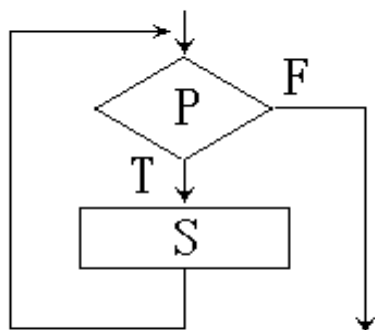
程序流程图



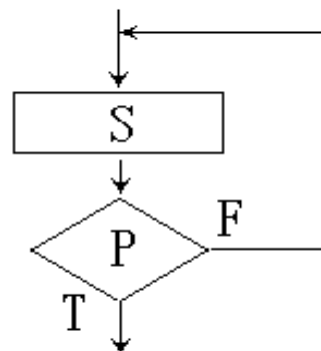
① 顺序型



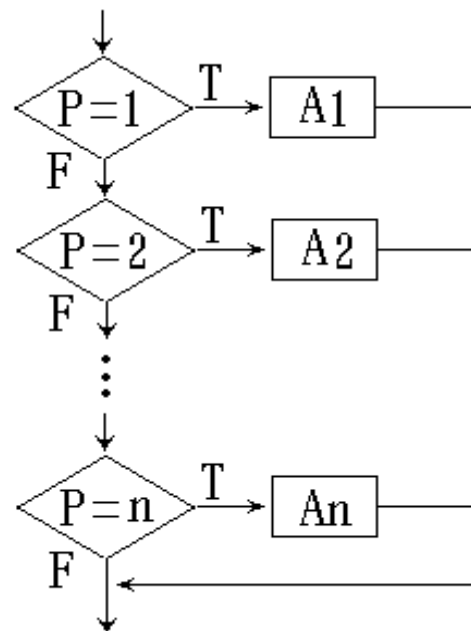
② 选择型



③ 先判定型循环
(DO-WHILE)

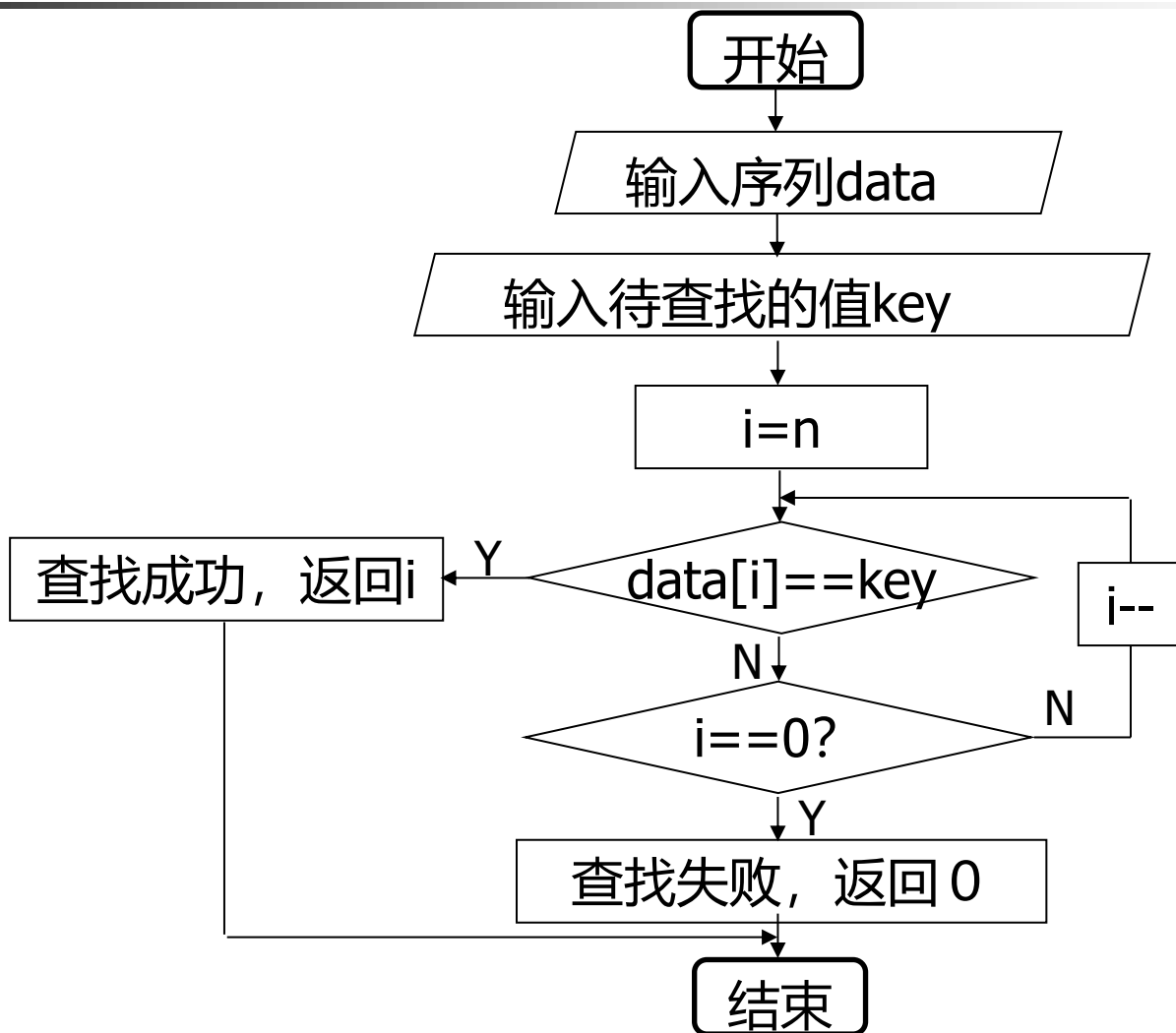


④ 后判定型循环
(DO-UNTIL)



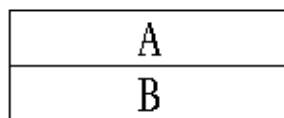
⑤ 多情况选择型
(CASE 型)

程序流程图举例

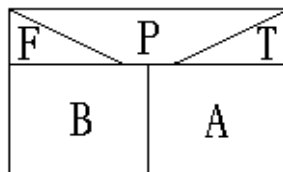


N-S图

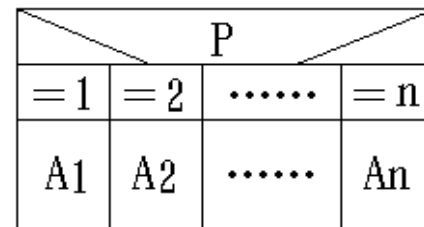
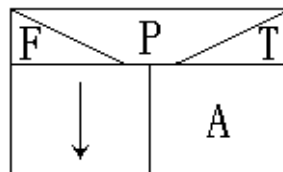
- N-S图也叫盒图，一种符合结构化程序设计原则的图形描述工具。
- 五种基本控制结构由五种图形构件表示：



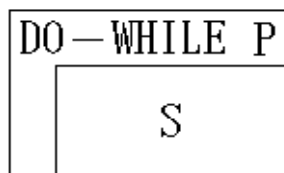
① 顺序型



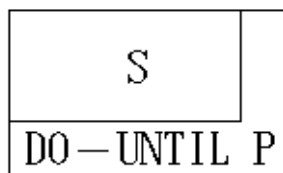
② 选择型



⑤ 多分支选择型
(CASE型)



③ WHILE 重复型



④ UNTIL 重复型



N-S图举例

输入序列data		
输入要查找的值key		
从最后一个元素开始查找		
T / data[i] == key / F		
查找成功 返回key 所在位置	T	全查过
	F	
	查找失败 返回 0	Do while (data != key) 下一次循环



伪码

- 以夹杂程序语法和自然语言的形式来描述解决问题的方法，有类C、类PASCAL语言等，本书用的是类C语言。
- 用类C语言描述

```
Status Search_Seq(SqList data,KeyType key)  
{  
    data.elem[0]=key;  
    for(i=data.length;!EQ(data.elem[i],key);i--);  
    return i;  
}
```


顺序查找数据序列中某特定值的程序代码



Search_Seq.h代码

//说明部分

#include <stdio.h> **//包含预编译头文件**

#define TRUE 1 **//宏定义**

#define FALSE 0

typedef int KeyType; **//类型别名的定义**

typedef struct {

 KeyType *elem;

int length;

}SqList;

bool EQ(KeyType i, KeyType j) **//判断两个整数是否相等**

{ **if**(i==j) **return** TRUE;

else return FALSE;

}

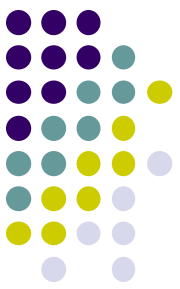
stdio.h

math.h

string.h

malloc.h

顺序查找数据序列中某特定值的程序代码

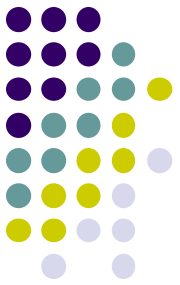


//函数实现

```
void Construction(SqList &L)    //构造无序序列
{
    int i;
    printf("input the length of data:\n");
    scanf("%d",&L.length);
    L.elem=(KeyType*)malloc((L.length+1)*sizeof(KeyType));
    printf("input the key:\n");    //输入待查找的关键字
    for(i=1;i<=L.length;i++)
        scanf("%d",&L.elem[i]);
}

int Search(SqList L,KeyType key)    //查找函数
{
    int i;
    L.elem[0]=key;
    for(i=L.length;!EQ(L.elem[i],key);--i);
    return i;
}
```

顺序查找数据序列中某特定值的程序代码



Search_Seq.cpp代码

```
#include "Search_seq.h"
```

```
void main()
```

```
{    KeyType key;
```

```
    SqList L;
```

```
    printf("input key:\n");
```

```
    scanf("%d",&key);
```

//读入要查找的值

```
    Construction(L);
```

//创建数据序列

```
    int position=Search(L,key);
```

//查找

```
    printf("the key is located in %d", position);
```

```
}
```

1.4 算法和算法分析

- **算法**：对特定问题求解步骤的描述，指令的**有限**序列，一条指令表示一个或多个操作。
- **算法的特性**：
 - **有穷性**：算法应在执行**有穷步**后结束，每步应在**有穷时间**内完成。
 - **确定性**：每条指令都有确切含义，且算法只有唯一执行路径，对相同输入只能得出相同输出。
 - **可行性**：算法中描述的操作可通过已经实现的**基本运算**执行有限次实现。
 - **输入**：有0个或多个输入。
 - **输出**：有一个或多个输出。



算法与程序的区别？



1.4.2 算法设计的要求

- **正确性**：设计或选择的算法能满足具体问题的需求。
 - 对几组输入数据能够得出满足要求的结果；
 - 对精心选择带有刁难性的几组数据能得出满足要求的结果；
 - 对于一切合法的输入数据都能产生满足要求的结果。
- **可读性**：易于阅读和交流。
- **健壮性**：
 - 当输入数据非法时，算法能适当地作出反应或进行处理，保证不会产生莫名其妙的输出结果；
 - 处理出错的方法是返回一个表示错误或错误性质的值，以便在更高的抽象层次上进行处理，不应中断程序的执行。
- **高效率和低存储量需求**：
 - **效率**：算法执行的时间；
 - **存储量**：算法执行过程中所需要的最大存储空间。



1.4.3 算法效率的度量

- **事后统计：**运用计算机计时统计算法运行时间，缺陷：
 - 要运行算法对应的程序；
 - 时间统计结果依赖于计算机软、硬件环境；
- **事前分析估计：**和算法执行时间相关的因素有：
 - 算法采用的策略
 - 问题的规模
 - 书写程序的语言
 - 编译器产生的机器代码的质量
 - 计算机执行指令的速度



1.4.3 算法效率的度量

- 算法=控制结构+原操作(固有数据类型的操作)

算法执行时间= $\sum_{i=1}^n$ (原操作i的执行次数×原操作i的执行时间)

- 从算法中选一种对所研究问题来说是**基本操作**的原操作，以该基本操作重复执行次数作为算法的时间量度。
- **算法的渐近时间复杂度**：设算法中基本操作重复执行次数是问题规模n的某个函数f(n)，算法的时间量度记作 $T(n)=O(f(n))$ ，表示随问题规模n的增大，算法执行时间的增长率和f(n)的增长率相同，称为算法的渐近时间复杂度（时间复杂度）。
- 算法时间复杂度的大O表示法一般表示算法在最坏情况下的时间代价。



大O表示法的加法规则和乘法规则

- **加法规则：**针对**并列**程序段

$$\begin{aligned}T(n, m) &= T_1(n) + T_2(m) \\ &= O(\max(f(n), g(m)))\end{aligned}$$

- **乘法规则：**针对**嵌套**程序段

$$\begin{aligned}T(n, m) &= T_1(n) \times T_2(m) \\ &= O(f(n) \times g(m))\end{aligned}$$

- **特例：**若 $T_1(n)=O(c)$, $T_2(m)=O(g(m))$,

$$T(n, m)=T_1(n) \times T_2(m)=O(c \times g(m))=O(g(m))$$



例1

语句重复执行的次数

程序段	语句频度	时间复杂度
<code>{++x;s=0}@</code>	1	$O(1)$
<code>for(i=1;i<=n;++i)</code> <code>{++x;s+=x;}@</code>	n	$O(n)$
<code>for(i=1;i<=n;++i)</code> <code>for(j=1;j<=n;++j)</code> <code>{++x;s+=x;}@</code>	n^2	$O(n^2)$
<code>i=1;</code> <code>while(i<=n)</code> <code>i=i*2; @</code>	$\log_2 n$	$O(\log_2 n)$



例2

```
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];
  }
```

问题规模:

$n \times n$

基本操作:

$c[i][j] += a[i][k] \times b[k][j];$

基本操作重复执行的次数:

$f(n) = n^3$

算法时间复杂度:

$T(n) = O(f(n)) = O(n^3)$



例3

```
void add (float x[ ][ ], int m, int n)
{   for(int i= 0; i<m; i++)
    {   sum[i] = 0.0;
        for(int j= 0; j< n; j++)
            sum[i] +=x[i][j]; }
    for(i=0; i<m; i++)
        printf("%f",sum[i]) ;
}
```

问题规模:

$m \times n$

原操作:

$sum[i] += x[i][j]$

基本操作重复执行的次数:

$f(n) = m \times n$

算法时间复杂度:

$T(n) = O(f(n)) = O(m \times n)$



例4

```
for(i=2;i<=n;++i)
    for(j=2;j<=i-1;++j)
    {
        ++x;
        a[i][j]=x;
    }
```

问题规模:

跟n有关

原操作:

{++x; a[i][j]=x;}

基本操作重复执行的次数: 难以精确计算, 量阶是 n^2

算法时间复杂度:

$T(n) = O(n^2)$



例5

```
void bubble_sort(int a[],int n)
{   for(i=n-1,change=TRUE;i>=1,change;--i)
    {   change=FALSE; //判断是否有相邻元素交换
        for(j=0;j<i;++j)
            if(a[j]>a[j+1])
            {   a[j] ↔ a[j+1];
                change=TRUE; }     } }
```

问题规模:

n

基本操作:

$\text{if}(a[j]>a[j+1]) \{ \dots \}$

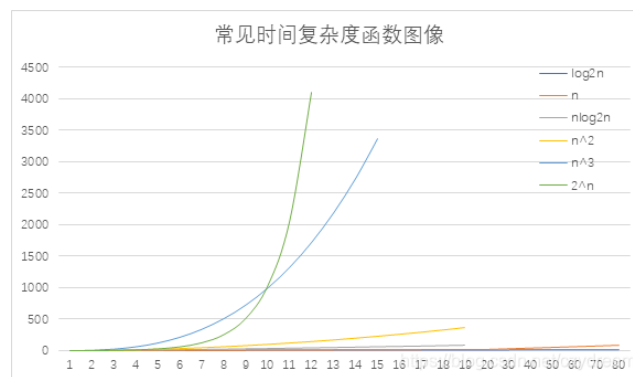
基本操作重复执行的次数: 不定

算法时间复杂度:

$T(n) = O(n^2)$

算法设计、选取和时间复杂度分析的原则

- 尽可能设计、选择多项式阶 $O(n^k)$ 的算法，不用指数阶。



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

- 算法的时间复杂度只考虑问题规模 n 的增长率，在难以精确计算基本操作执行次数时，只需求得它关于 n 的**增长率或阶**即可。
- 算法中基本操作重复执行次数随问题的输入数据集不同而不同，要考虑**最坏**情况下的时间复杂度。



例6 百钱买百鸡问题(1)

鸡翁一，值钱五，鸡母一，值钱三，鸡雏三，值钱一，百钱买百鸡，问鸡翁、鸡母、鸡雏各几何？

解：设鸡翁、鸡母、鸡雏分别为 x , y , z 只。则

$$x + y + z = 100$$

$$5x + 3y + z/3 = 100$$

方法1：用三重循环

```
for(i=0;i<=100;i++)  
    for(j=0; j<=100; j++)  
        for(k=0; k<=100;k++)  
        {  
            if(k%3 ==0 && i+j+k==100 && 5*i+3*j+k/3==100)  
                printf(“%d,%d,%d\n”, i,j,k);  
        }
```



例6 百钱买百鸡问题(2)

方法2: 用两重循环

```
for(i=0;i<100;i++)
    for(j=0;j<100;j++)
    {
        k=100 -i -j ;//小鸡的数目可以由母鸡数和公鸡数得到
        if(k%3==0 && 5*i+3*j+k/3==100)
            printf("%d,%d,%d\n",i,j,k);
    }
```

方法3: 用两重循环

```
for(i=0;i<=20;i++) //公鸡数不可能超过20只
    for(j=0;j<=33;j++)//母鸡数不会超过33只
    {
        k=100 -i -j ;
        if(k%3==0 && 5*i+3*j+k/3==100)
            printf("%d,%d,%d\n",i,j,k);
    }
```


例6 百钱买百鸡问题(3)

方法4: 用一重循环

由 $x+y+z = 100$ 和 $5*x+3*y+z/3=100$ 合并为一个方程:

$14*x+8*y = 200$, 进一步简化为: $7*x+4*y=100$

x 不超过14, 且必为4的倍数, 有:

```
for(i=0;i<=14;i+=4)
{
    j = (100 - 7*i)/4;
    k=100 - i - j;
    printf("%d,%d,%d\n", i,j,k);
}
```

时间复杂度

- 方法一的循环次数为: $100*100*100$ 次; $O(n^3)$
- 方法二的循环次数为: $100*100$ 次; $O(n^2)$
- 方法三的循环次数为: $21*34$ 次; $O(n^2)$
- 方法四的循环次数为: 4次。 $O(n)$



1.4.4 算法的空间复杂度度量

- **渐进的空间复杂度：** 算法所需存储空间的量度，记作： $S(n)=O(g(n))$ ，其中， n 为问题的规模，表示随着问题规模 n 的增大，算法运行所需存储量的增长率与 $g(n)$ 的增长率相同。
- **算法需要的存储量包括：**
 - 输入数据所占空间；
 - 程序本身所占空间；
 - 额外辅助空间，当其相对于输入数据所占空间（问题规模）是常数，称算法为**原地工作**。

对一个算法在运行过程中临时占用存储空间大小的量度



1.4.4 算法的空间复杂度度量

```
void add(int n,int a[n][n],b[n][n],c[n][n])
{  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];
    }
}
```

空间复杂度 $O(1)$

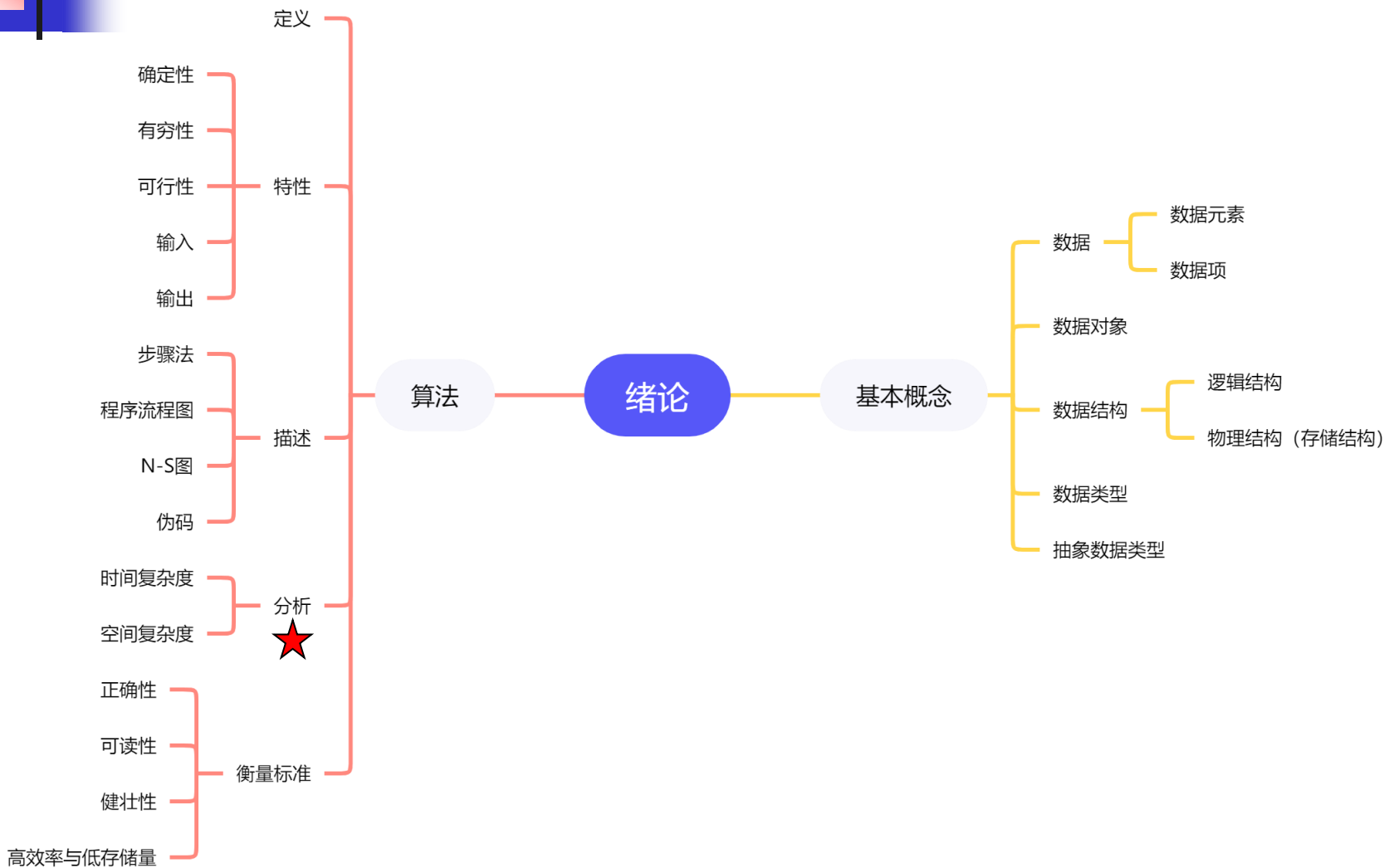


1.4.4 算法的空间复杂度度量

```
void add(int n,int a[n][n],b[n][n])
{  int c[n][n];
   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];
     }
}
```

空间复杂度 $O(n^2)$

第一章 思维导图



练习

下面操作违反了算法的（ ）。

```
void sam( )
{   n=2 ;
    while(n%2==0) n=n+2;
    printf ( n ) ;
}
```

- ☒ A 有穷性
- ☐ B 确定性
- ☐ C 可行性
- ☐ D 健壮性

提交



练习-考研真题

(2022) 下列程序段的时间复杂度是

```
int sum=0;  
for (int i=1; i<n; i*=2)  
    for (int j=0; j<i; j++)  
        sum++;
```

A $O(\log_2 n)$ B $O(n)$ C $O(n^2)$ D $O(n \log_2 n)$

什么是问题规模? 是用算法求解问题时需处理数据量的多少,一般用整数 n 表示。 n 越大, 算法执行的时间越长。

什么是量阶/阶? 描述算法解决问题所需时间与问题规模之间的关系的方式



练习

下面算法的空间复杂度为 _____。

```
1  int foo(int n)
2  {
3      int i, j, s = 0;
4      for (i = 1; i <= n; ++i)
5      {
6          for (j = 1; j <= n; ++j)
7          {
8              s += i * j;
9          }
10     }
11     return s;
12 }
```

空间复杂度 $O(1)$ 64

应掌握的类型题(1)

一、数据结构的基本概念

- 1 在数据结构中，从逻辑上可以把数据结构分成____和____。
- 2 线性结构中元素之间存在____关系，树形结构中元素之间存在____关系，图形结构中元素之间存在____关系。
- 3 数据元素是数据的最小单位(对/错)。
- 4 数据类型是一个值集合和定义在该值集上一组____总称。
- 5 数据元素在计算机中有两种不同的存储结构即_____。

二、算法的时间、空间复杂度分析

1、将下列函数按 $n \rightarrow \infty$ 时的无穷大阶数，从小到大排序

$n, n - n^3 + 7n^5, n \log_2 n, 2^{n/2}, n^3, \log_2 n, n^{1/2} + \log_2 n, (3/2)^n, n!, n^2 + \log_2 n$

答： $\log_2 n, n^{1/2} + \log_2 n, n, n \log_2 n, n^2 + \log_2 n, n^3, n - n^3 + 7n^5, 2^{n/2}, (3/2)^n, n!$



应掌握的类型题(2)

```
2 void prime(int n)
  {   int i=2;
      while((n%2)!=0&& i*1.0<sqrt(n))      i++;
      if (i*1.0>sqrt(n)) printf("%d is a prime");
      else    printf("%d is not a prime");
  }
```

解：时间复杂度是 $O(\sqrt{n})$ 。

3 用事前估计与事后统计相结合的方法估算算法运行时间。设两个矩阵相乘算法的时间复杂度为 $T(n)=O(n^3)$, 两个 $10*10$ 的矩阵相乘, 执行时间为12ms, 请估计两个 $31*31$ 矩阵相乘的时间。

解：time=12*(31/10)³



郑州大学历年考试题选(1)

1. 已知有实现同一功能的两个算法A和B，其时间复杂度分别为 $O(n)$ 和 $O(n^2)$ ，当 $n=100$ 时，执行A算法需要10秒钟，请问在同一运行环境下，对 $n=100$ 能否估计B算法的运行时间？若能试估计之；否则说明原因。
2. 任何一个算法设计取决于选定的数据(逻辑)结构，而算法的实现依赖于采用的存储结构。
3. 算法有穷性的含义是对任何合法输入值，一个算法必须在执行有穷步后结束，且每一步都可在有穷时间内完成，也就是说，在算法中不能使用无限循环语句。



郑州大学历年考试题选(2)

4. 计算下列程序段中 $x-=10$ 的执行次数。

```
x=91; y=100;  
while ( y>0 )  
    if ( x>100 ) { x-=10; y - -; }  
    else x++;
```

5. 有一计算矩阵相乘的程序，它的时间复杂度为 $O(n^3)$ ，当上机运行两个 10×10 矩阵相乘时，执行时间为 5ms，试计算两个 30×30 的矩阵相乘所需的时间。

6. 程序段 $\text{for}(i = 1; i \leq n; i++)$
 $\text{for}(j = i; j \leq n; j++)$
 $k++$;

$k++$ 执行的频度是_____。



郑州大学历年考试题选(3)

7. 设问题规模为 n ，算法1和算法2的基本语句执行的频度分别为 $f(n)$ 和 $g(n)$ ，用户从 $n=1$ 测试到 $n=10^3$ ，发现 $f(n) < g(n)$ 总成立，请问：算法1的时间复杂度小于算法2的时间复杂度吗？说明理由。
8. 一个算法的基本语句执行的频度如下，其中 n 是问题的规模，试计算该算法的时间复杂度。

$$T(n) = \begin{cases} 0 & n = 1 \\ T(n-1) + n - 1 & n > 1 \end{cases}$$

郑州大学历年考试题选(4)

9. 设n为3的倍数，试分析带 “ * ” 语句的执行频度。

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

```
    * if (3*i <= n) -----n
```

```
        * for ( j=3*i; j<=n; j++ ) -----  $\sum_{i=1}^{n/3} (n - 3i + 2) = \frac{n(n+1)}{6}$ 
```

```
            { x++; y=3*x+2; }
```

10. 试分析带 “ * ” 语句的语句执行频度。

```
* for ( i = 1; i < n; i++; ) -----n
```

```
    * for ( j= n; j >= i+1; j-- ) -----  $\sum_{i=1}^{n-1} (n - i + 1) = \frac{(n-1)(n+2)}{2}$ 
```

```
        if ( a[ j-1 ] > a[ j ] )
```

```
            { t = a[j-1]; a[j-1] = a[j]; a[j] = t; }
```



练习(1)

1. 任何一个算法的设计都取决于选定的_____。
A. **数据逻辑结构** B. 数据的物理结构 C. 数据存储结构 D. 数据的数据类型
2. 算法的健壮性的含义是_____。
A. 程序对于几组输入数据能够得出满足规格说明要求的结果。
B. 程序对刁难性的输入数据能得出满足规格说明要求的结果。
C. 程序不含语法错误。
D. **当输入数据非法时，算法能适当地作出反应或进行处理，不会产生莫名其妙的输出结果。**



练习(2)

3.数据元素是_____。

- A. 数据不可分割的最小单位
- B. 性质相同的数据集合
- C. 数据的逻辑关系
- D. **数据的基本单位，常作为一个整体处理**

4. 算法效率是根据该算法编制的程序在计算机上运行所消耗的时间而度量的，程序所消耗的时间_____。

- A.取决于算法选用的策略和问题规模，与书写程序的语言无关.
- B.取决于问题规模和书写程序的语言，与算法选用的策略无关.
- C.只与编译程序产生的机器代码质量和机器执行指令速度有关.
- D.**与算法选用的策略、问题规模、书写程序的语言、编译程序产生的机器代码的质量和机器执行指令的速度都相关。**



练习(3)

5. 下面函数不能满足算法的要求，它违反了算法的_____。

```
void sam ( )  
{  n=2 ;  
    while(n%2== 0)  n = n + 2 ;  
    printf ( n ) ;  
}
```

A. 有穷性 B. 确定性 C. 可行性 D. 健壮性

6. 下面函数不能满足算法的要求，它违反了算法的_____。

```
void sam ( )  
{  int y = 0 ;  
    int x = 5 / y ;  
    printf(x, y) ;  
}
```

A. 有穷性 B. 确定性 **C. 可行性** D. 健壮性