算法艺术与信息学竞赛

刘汝佳　黄亮　著

清华大学出版社

算法艺术与信息学竞赛

刘汝佳　黄　亮　著

清华大学出版社

北　京

内 容 简 介

本书较为系统和全面地介绍了算法学最基本的知识。这些知识和技巧既是高等院校“算法与数据结构”课程的主要内容，也是国际青少年信息学奥林匹克（IOI）竞赛和 ACM/ICPC 国际大学生程序设计竞赛中所需要的。书中分析了相当数量的问题。

本书共 3 章。第 1 章介绍算法与数据结构；第 2 章介绍数学知识和方法；第 3 章介绍计算几何。全书内容丰富，分析透彻，启发性强，既适合读者自学，也适合于课堂讲授。

本书适用于各个层次的信息学爱好者、参赛选手、辅导老师和高等院校计算机专业的师生。本书既是信息学入门和提高的好帮手，也是一本内容丰富、新颖的资料集。

图书在版编目（CIP）数据

算法艺术与信息学竞赛 / 刘汝佳，黄亮著. —北京：清华大学出版社，2003

ISBN 7-302-07800-9

Ⅰ. 算… Ⅱ. ① 刘… ② 黄… Ⅲ. 算法-自学参考资料 Ⅳ. 0242.23

中国版本图书馆 CIP 数据核字（2003）第 116045 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦

http://www.tup.com.cn　　**邮　　编**：100084

社 总 机：010-62770175　　**客户服务**：010-62776969

组稿编辑：欧振旭

文稿编辑：余　姬　陈韦凯

封面设计：钱　诚

版式设计：郑轶文

印 刷 者：清华大学印刷厂

装 订 者：三河市李旗庄少明装订厂

发 行 者：新华书店总店北京发行所

开　　本：185×260　**印张**：28　**字数**：618 千字

版　　次：2004 年 1 月第 1 版　2004 年 1 月第 1 次印刷

书　　号：ISBN 7-302-07800-9/TP・5685

印　　数：1～4000

定　　价：45.00 元

序　言

计算机解题的核心是算法设计。算法设计涉及许多先修的基础知识，包括数据结构、高级语言程序设计、离散数学、图论、组合数学、人工智能、计算几何等。当然还包括除数学与信息学之外的其他学科知识，因为没有这些知识，往往连题目都会看不懂，这可能也是要求参加 ACM 大赛的选手应该具备全面科学素养的原因之一。

刘汝佳、黄亮两位作者都曾在高中时参加过信息学奥林匹克竞赛活动，他们在如何用计算机解难题方面投入过很大精力，有着比较丰富的经验。在上大学之后，又投入了大量精力帮助训练中国队的小选手和参加 ACM 世界大学生程序设计大赛。他们深感这些活动对提高学生的能力和全面素质所起的巨大作用。因此，他们利用课余时间，广泛收集各地各类试题，并着力研究、分析与归类，想出比较好的解法，特别是总结出若干的思路和经验写成书和大家共享。从他们开始动笔到成书期间又花了大量的心血，对一些难题的解法也有了独到的见解。这本书应该说是很难得的经验之谈。对一个编程高手来说，借鉴别人的经验是十分重要的。因此，我想将这本书推荐给参加 IOI 的中学生和参加 ACM/ICPC 的大学生阅读。

本书的命名也有独到之处。就我本人的教学经历来看，算法的确是艺术。艺术与科学本来就是孪生姊妹，不科学的东西谈不上艺术。艺术给人以美的感受，而算法属于数学文化范畴，数学的美在算法中得到了充分的体现，特别是当今数学已经进入新的机器时代，利用计算机求解问题，需要人充分开动脑筋，解决一系列难点，解题过程本身就是一个精益求精追求完美的过程。在这样一个过程中，编程者在付出艰辛的努力之后，会有一种获得成功的愉悦。正如一些数学大师所言：数学是理性的艺术，是创造性的艺术。在编程解题的过程中，通过理性的思维和理性的实践，你一定会感受到算法艺术的无穷魅力。一个颇具匠心的好算法会让你拍案叫绝，感受到它的思维艺术之美。我们许多参加过 IOI 和 ACM/ICPC 程序设计大赛的选手，当问起他们当年的拼搏是否艰苦时，他们都说苦中有乐，苦中有甜。这可能就是感受到了这种思维艺术的魅力。

科学思维能力的提高是成就事业最重要的一个因素，希望本书对你思维能力的提高能有很大的帮助。

清华大学计算机系教授，博士生导师
信息学奥林匹克中国队总教练

2003 年 12 月

前　言

信息学（informatics）是一门新兴的学科，主要是指利用计算机及其程序设计来分析问题、解决问题的学问。随着计算机逐步深入生活，网络日趋普及，“信息革命”已经到来。信息学的地位逐步提高，青少年信息学竞赛也方兴未艾，在世界范围内受到了越来越多的重视。国际上，信息学竞赛主要分为两类：中学生的国际信息学奥林匹克和大学生的国际大学生程序设计竞赛。国际信息学奥林匹克（International Olympiad in Informatics, IOI）始于1989年，是联合国教科文组织倡导举行的五项国际学科奥林匹克之一。中国队每届都取得了名列前茅的佳绩，所有选手全部获奖，多次总分名列第一。国际大学生程序设计竞赛（ACM International Collegiate Programming Contest, ACM/ICPC）分为地区赛和国际总决赛，中国从1996年开始参加，现有3个地区赛赛点。清华大学、上海交通大学和中山大学等学校多次进入国际总决赛。上海交通大学队还获得了2002年的世界冠军的好成绩。从大局看，竞赛不是目的，而是推动普及的手段。虽然国内在培养信息学的优秀人才方面做了很大的努力，而且成绩斐然，但仍然存在一些遗憾之处：

第一，书籍资料的缺乏性。由于信息学竞赛的普及不如其他学科竞赛，辅导教师和研究人员相对匮乏，参与写作的人员也较少，致使国内此类书籍缺乏，而好书尤其缺乏。由此造成了不少信息学爱好者求书无门。另外，一些已出版的同类书籍或艰深难懂，或教条灌输，使不少爱好者望而却步，中途放弃，以至于信息学竟成了一种“曲高和寡”的学问。然而，社会信息化和信息学普及的大势已不可逆转，所以，信息学竞赛呼唤通俗易懂又有相当学术含量的好书。

第二，地区的不平衡性。由于信息学发展的时间毕竟不算长，很多地区在中学阶段刚刚开设或者还未开设信息学课程，师资力量相对比较薄弱。而且，信息学竞赛相对于其他学科，毕竟上手比较困难，这使得每年都有相当数目的信息学爱好者因为自学难度过大而中途放弃，从而导致国内信息学教学水平存在着很明显的地区不平衡性。很多地处信息学竞赛起步较晚地区的信息学爱好者们渴望能买到既实用又便于自学的书籍。

第三，国内视野的局限性。国内不少从事信息学竞赛研究的教育工作者把大部分精力放到了国内竞赛题目的研究中，而忽视了其他国家和地区的竞赛题目和研究成果。信息学竞赛题目的风格和内容往往受到本地传统文化等因素的影响，不同地区的题目往往有较大差异，因此熟悉各个国家的出题风格和特点，训练自己各方面的解题能力是很有必要的。所以，对于中高层选手和长期从事信息学竞赛辅导的老师来说，很需要紧跟国际动向、充分介绍国外成果的好书。

总之，从以上3点可以看出，国内信息学的普及度还远远不够。本书的出版，一方面是为了弥补国内信息学便于自学的普及读物方面的不足，拉近国内信息学竞赛起步较晚地

区与发达地区的差距。另一方面是为了向国内读者介绍国外最新研究成果和竞赛试题，填补这方面的空白，拉近国内和国际最新发展的距离。

本书在理论方面参考了国外的最新研究成果的论文报告，在实际运用方面大量选用了在国内研究较少的外国竞赛的优秀题目，对信息学竞赛理论研究和实践都具有一定的参考价值，同时本书也是一本难得的资料集。

本书分为 3 章，第 1 章介绍算法与数据结构。算法与数据结构是信息学中最重要的部分之一，内容多而杂，不容易从整体上把握。本章的前三节介绍复杂度分析基本理论、基础算法和基础数据结构，重在给读者一定的感性认识，然后分三个专题介绍三个重要的问题：数据结构的应用、动态规划和搜索。第 2 章介绍信息学中用到的数学。数学是信息学的基础，因此本书专门用一章的篇幅加以详细论述。本章容量大，理论性比第 1 章强，涉及到基本代数方法、初等数论、组合数学和图论等问题。第 3 章介绍计算几何的基础知识、基本算法以及技巧。3.1 节从最基本的位置和方向问题介绍叉积和点积；3.2 节过渡到多边形、多面体及其容积、重心的求取以及形内形外的判断；3.3 节讨论凸包这个最基本的几何模型及其应用；3.4 节介绍了几个常用的特殊算法，包括分治法、离散化和扫描法，还介绍了增量和随机增量算法。

本书包含的内容非常多，各个层次、各种需求的读者都能在本书中找到适合自己的内容。本书丰富的内容能给读者以很大的选择余地，不同难度的例题和习题中既有引导读者兴趣的入门题目，又有极富挑战性的精彩题目，习题编号前的 * 越多，表示作者越推荐。

本书的第 1 章和第 2 章由刘汝佳编写，第 3 章由黄亮编写，在成书过程中还得到了很多老师和选手的大力支持。

在第 3 章的写作过程中，上海交通大学 ACM 代表队的不少队员和教练给了作者许多帮助。

要特别感谢陆靖；感谢远在美国的陈晓敏，他与作者进行了多次富有启发意义的讨论并提供了不少国内罕见的资料；感谢吕侃、陶云峰、杨寅、严琦琦、林晨曦、龚理、田斌、张羲等同学对本书写作的支持和与作者进行的讨论。

感谢世界著名计算几何学家 Joseph O'Rourke 博士对作者的启发。

在前两章的写作过程中，部分 IOI2002 和 IOI2003 中国国家集训队的成员提出了不少宝贵的意见，也提供了一些资料作为帮助，他们是王知昆、张一飞、李睿、何林、毛子青、骆骥、齐鑫、赵爽、金恺、李益明、符文杰、刘才良、张宁、黄芸。

感谢俞玮和林希德，他们与作者一起进行了大量的试题翻译工作，并讨论和撰写了题目分析。

感谢在讨论中启发作者思维并教会作者不少知识的外国朋友们：瑞典的 Jimmy Mardell、加拿大的 Derek Kisman、波兰的 Tomasz Malesinski、乌克兰的 Alexandar Grushetsky、保加利亚的 Petko Minkov。

感谢中国著名人像摄影家魏德运先生在本书的出版过程中所给予的帮助。

感谢北京师范大学 ACM 代表队的吴莹莹同学为本书的出版所给予的大力支持和帮助。

特别感谢在本书写作过程中对作者大力支持的各位老师！他们是：IOI 中国队总教练吴文虎老师、ACM/ICPC 清华大学代表队总教练王帆老师和邬晓钧老师、ACM/ICPC 上海交通大学代表队总教练俞勇教授、ACM/ICPC 德黑兰赛区总裁判 Ramtin Khosravi 先生、ACM/ICPC 世界总决赛裁判 Shahriar Manzoor 先生和 ACM/ICPC 世界总决赛筹划指导委员会的 Miguel Revilla 先生。

作　者

2003 年 12 月 2 日

如何阅读本书

欢迎大家阅读本书！为了更好地发挥本书的作用，我们建议在阅读本书之前先熟悉本书的内容概况和特点，花一些时间来认真看看“如何阅读本书”部分。

本书的读者对象

本书的读者大致分为四类，我们分别给出不同的建议，供参考。

第一类，中学信息学竞赛选手

中学生的竞赛分为不同的层次。从纯普及的分区联赛，到普及兼选拔的 NOI 全国青少年信息学奥林匹克竞赛，纯选拔性质的 IOI 中国国家队选拔赛 CTSC 和纯竞赛性质的 IOI 国际青少年信息学竞赛，需要掌握的内容和难度都有很大的变化。分区联赛的选手应当学习一门程序设计语言并掌握到相当熟练的程度，然后学习本书中比较重要而易懂的内容；NOI 选手应当学会书中除了部分高难度的例题和带 * 内容之外的其他内容，而国家集训队员应当尽量掌握书中所有内容并完成大部分习题。极少数习题只是提供给有兴趣的读者进行研究，而不用强迫自己掌握。由于中学生竞赛对算法设计的要求很高，所以这一类读者应该特别注意体会书中例题并完成习题。

第二类，ACM/ICPC 大学生程序设计竞赛选手

大学生竞赛的特点和中学生不一样。ACM/ICPC 中编程速度和正确性显得尤其重要，而对算法设计的要求从整体上比中学生竞赛低了很多。针对比赛的这个特点，建议这一类读者重点阅读 1.3 节、2.1 节、2.2 节、2.3 节的全部内容以及 1.4 节、1.5 节、1.6 节、2.5 节中的理论部分和比较简单的例题，而其他内容只需要学习基础内容和 ACM/ICPC，UVA Problem Archive 和 Ural State Problem Archive 中的例题，其中难度过大的题目可略过。国内 ACM 竞赛正处在发展阶段，有一大批选手刚刚接触到竞赛，经验还不够。建议这些选手先浏览全书，然后先阅读理论部分，跳过一些思路巧妙或者难度很大的题目。ACM 竞赛中，速度和正确性是非常重要的，因此强烈建议大家把书中大部分的程序都亲自编写一下，并加入到自己的程序库中。

第三类，竞赛辅导教师和教练

教师和选手不一样，他们往往有比较充足的时间且不用花很多精力来训练自己的编程能力与保持状态，因此可以更从容地阅读本书。建议各位老师能通读全书，不断找出自己最感兴趣的部分深入阅读，用这种方式逐渐了解本书的所有内容。对于教师来说，了解全貌是非常重要的，因为这可以极大地帮助学生沿着正确的路子学习和训练。

第四类，计算机编程爱好者和其他读者

兴趣是最好的老师。这一类读者可以随意选择感兴趣的内容进行学习，如 1.2 节、1.3 节、2.5 节等节的题目都很有趣。另外，由于读者可以忽略大部分的证明和复杂度分析，而重点体会书中蕴涵的思想，这样的学习必将是有趣和吸引人的。

本书的目标和特点

学习信息学主要有两个层次的要求：理解和掌握。本书尽量让读者多理解一些难于理解的内容，若想熟练地掌握，读者必须另外下很大的功夫，配合阅读其他书籍并做大量的练习，进行一些实践。限于篇幅，很多理论和题目的细节本书都无暇顾及。这虽不大影响读者理解内容的精髓，但是当读者在细究一些问题（例如一些术语的准确定义、证明、伪代码和一些实现细节等）时会发现书中并未提到。其实，在多数情况下，这些内容可以在同类书中学到，也可在作者为本书专门设计的主页上找到丰富的学习资料。突出特色，启发思维才是本书的目标。

本书的特点有三个。了解了这些特点有助于读者在学习的过程中有意识地体会作者的“另一个用意”，从而最大限度地掌握内容。

第一个特点是启发性。这也是本书相对比国内已出版的同类书籍最大的特色。与传统此类书籍“填鸭式”灌输算法不同，本书力戒教科书式的死板讲解，力争引导读者进行创造性的思维，点拨关键思路，使读者在探索中自然领悟算法的精髓，并为其优美所吸引，激发浓厚的兴趣。传统书籍讲新算法时，往往作为一个全新的事物引入，而不是转化为已学的知识，也不讲与其他算法的联系，导致学生认为信息学千头万绪，琐碎复杂，而始终不得其精髓，更不用说“融会贯通”。其实算法都是相通的，万变不离其宗。故本书讲解时，大多先由一个有趣的实例引出，引导读者从几个已学的方向自行探索，逐步抛弃错误的方向，改为正确的方向，循循善诱，一步步把读者引向正确的算法，让读者觉得，新算法不过就是从老算法演化而来，并不深奥难懂。因此，建议读者在阅读时特别注意各个知识点是怎样被引入的。一个知识点引用得越曲折，说明这个知识点越是重要，越有可学之处。总之，本书能启发读者进行创造性思维，自然体会算法和知识的正确性、优美性、深刻性、广泛性和统一性，不仅能化为己用，还能有所创新。

第二个特点是信息量大。本书在描述算法时尽可能简洁，突出主要矛盾。书中涉及的源程序在本书网站上都有，而书上的篇幅尽可能多的讨论算法的精髓、与其他算法的异同、比较时空复杂度优劣、适用条件、互相转化等，大部分内容都是同类书籍中很难找到的。读者也许会抱怨某些知识讲得不够深，不够细，但实际上这些有限的篇幅被用来讲解更为有用的知识了。我们用网站来保证读者可以有条件学习到这些知识，而腾出更多的篇幅来介绍本书特有的内容。信息学竞赛是一种讲究创新的比赛，本书在讲解知识之余十分注重鼓励读者思维创新，在介绍大量知识和技巧的同时给读者留下广阔的思维空间。书中某些问题只是轻描淡写地提了一句，但是为读者提供了很大的思考空间，读者可以独立研究，

也可以互相讨论。本书主页上的内容是本书一个很好的补充。只要读者肯钻研，能从本书中获取的信息量是很大的。

第三个特点是内容新。本书大量选用了同类书籍中很少涉及的其他国家的国内比赛和一些区域性比赛中的优秀题目，其中有很多题目是从波兰语，罗马尼亚语等翻译过来的。还有大量例题和习题来自近年来蓬勃发展的网上在线试题库（如西班牙巴拉多里德大学（Universidad de Valladolid）试题库和俄罗斯乌拉尔州立大学（Ural State University）试题库），以及各种在线程序设计比赛的题目。另外，在理论方面参考了国外的最新研究成果的论文报告。这些内容对于国内的绝大多数读者来说，题目耳目一新，本身就反映了国际信息学发展的动态，其解法又代表了世界信息学发展的最新成果。这对于“久经沙场”的老选手和长期从事信息学竞赛的辅导老师来说，也是紧跟国际最新发展的趋势。

使用本书的建议

信息学是一门实践性很强的学科，读者在学习本书时应该多种方法相结合。下面一些建议，这对于大多数希望掌握本书大部分内容的读者来说是适用的。

第一，重视基础和语言的实现能力。本书一般没有完整的源程序，只有比较重要的理论，有PASCAL/C++或者其伪代码，而且格式也不统一。请大家自己进行编码练习，实现书中介绍的算法，如参考数据结构和算法书籍，本书的网站上也可以找到一些题目的源程序。书里对编码比较困难的部分做了专门提示，请读者留意这些内容。

第二，善用其他参考书结合学习。建议把本书和其他书籍结合起来阅读。例如，数据结构部分可以先阅读本书的 1.3 节，有一个感性认识，然后认真学习相关的教材，搞清楚很多细节，最后再研究一下 1.3 节，这时将会体会到一些新内容。永远记住，本书的性质是导引，目的是让读者明白该学些什么，怎样学，很多具体问题还需要专门学习。如果读者没有这些书籍也不要紧，在本书的网站上可以找到这些熟悉的资料。

第三，感性和理性相结合。本书的内容虽然科学性很强，但这并不意味着读者应该用完全理性的方式来学习。在本书的很多地方，作者都用了相当“感性”的语言。这样做虽然损失了一定的严密性，但是可以让读者了解到其中的思路和动机等内涵很丰富的内容。读者不要轻视这些内容，仔细琢磨一定会受益匪浅。

第四，注意细节。也许一些读者对本书的大部分内容是很熟悉的，但作者想提醒的是，即使某些内容大家已经非常熟悉了，但还是建议仔细阅读本书。不管是为了明确概念，澄清误区还是开阔眼界，很多细节都是值得注意的。

第五，和 Internet 积极配合。在 Internet 飞速发展的今天，本书将采用借助网页来达到更好的学习效果。本书主页的 URL 是：http://oibh.ioiforum.org/book-lrjhl/。如果以后本书的主页转移位置，则可以通过用本书书名作为关键字在 Internet 上检索。在本书的主页中，你可以看到以下内容：

- 进一步阅读的资料；

- 本书中部分题目的测试数据和源程序；
- 本书的勘误表；
- 读者提问和解答；
- 作者和读者写的关于本书的文章；
- 一个专门为读者设立的论坛，供读者交流阅读心得。

目　录

第1章　算法与数据结构

算法与数据结构是信息学竞赛最核心的部分，也是选手必备的基础知识。“数据结构+算法=程序设计”。这些知识不仅很重要，而且是学习其他内容的基础。

本章内容简介

本章从理论分析和实际应用两方面阐述了算法与数据结构的基本知识。由于同类书籍对相关理论作了较为详尽的叙述，本书把重点放在了实例分析上，因为它们可以帮助读者更为深刻地理解算法与数据结构的精髓。

1.1 节概括地叙述了算法、数据结构以及计算理论的一些简单概念，目的是让读者从宏观上对第 1 章的基础做一定的了解。

1.2 节从实例出发，概括地介绍了一些基本算法，包括枚举、贪心、递归、递推法等。本节的重点是让读者领会到常用的算法思想，因此所选的部分例题有相当的难度，读者需反复体会才能明白其中的道理。本节还有一些小专题，供读者选学。

1.3 节介绍基本数据结构，包括线性表、队列、栈、树、二叉树以及图的遍历与拓扑排序。和 1.2 节不同的是，本节的系统性比较强，为了突出趣味性和实用性，本节选用了一些有趣的故事引入这些内容，从而增加读者的感性认识。建议读者阅读专门的数据结构教材来获得系统的理论知识，打好基础。

1.4 节介绍一些实用数据结构，包括哈希表、二叉搜索树、Trie 结构、线段树、堆、并查集等。本节的难度较大，而且实用性比较强，1.5 节和 1.6 节将广泛使用这里介绍的知识。

1.5 节介绍动态规划方法和一些经典动态规划问题，并结合一些例子重点讲述常见的建模技巧。

1.6 节介绍搜索算法。本节内容包含盲目搜索、启发式搜索、博弈算法和搜索的优化问题。本节的部分理论知识和例题比较难懂，建议读者多写程序来实现题解中提到的一些细节。

在大多数小节的后面，作者从近年来国内外竞赛中出现的基本算法与数据结构的题中，精选出一些问题以检验读者的学习效果。部分题目具有相当的难度，读者可以反复思考。书后附有部分练习的提示。

1.1　编程的灵魂——数据结构+算法=程序

本节介绍数据结构、算法的基本概念和复杂度分析的基本方法。本节的理论性比较强

且部分内容和一些有一定基础的读者的“经验”有些违背。这些内容比较抽象，部分内容（如并行计算机、PS 完全问题等）主要只是开阔读者的眼界。本节的重点是渐进时间复杂度的计算和化简，而难点是各个相关概念的正确理解和对计算机解题能力的整体认识。

本节不难学，建议读者同时另外阅读一本介绍计算复杂性的入门书籍，和本书结合起来看。阅读本书之前，读者需要熟练掌握一种程序设计语言，推荐使用 C/C++或者 JAVA，但同时需要了解一下 Pascal，如果不熟悉这些语言，阅读本书的效果则将会大打折扣。

首先需要弄清楚：什么是算法？什么是数据结构？为什么要学它们？简单地说，**算法（algorithm）**就是解决问题的方法或者过程。如果把问题看成是函数，那么算法就能把输入转化成输出；**数据结构（data structure）**是数据的计算机表示和相应的一组操作。这二者是最基本的，但对于初学者来说，最熟悉的名词并不是它们，而是**程序（program）**算法和数据结构的具体实现。这三者的关系在本节的标题中已经指出了：“数据结构+算法=程序”。

本章在介绍算法时强调：

1. 算法思想：深刻地理解这些算法设计思想将有助于开阔读者的思维。

2. 算法分析：从时空性能、适用范围等对介绍的算法进行分析，增强读者对这些内容的理性认识，学会定性、定量地分析算法和进行比较。

需要说明的是，我们强调的是算法设计和分析，而不是对问题本身的分析。对问题本身性质的分析只是粗略提一下，作为算法设计的指导，而不作为重点学习和研究的内容。不过，一些重要的结果和方法，如 NP 完全理论和判定树模型，由于它们容易理解且用处很大，我们仍会花一些篇幅进行讲述。

时空开销增长　要比较两个算法的各方面性能有很多方法。其中之一是各写一个程序，比较它们的运行情况。但是由于程序并不是算法的直接对应结果，无法避免一些和算法无关，只由代码产生的问题。而且需要写程序，这将花费比较大的精力，尤其是在算法复杂，实现容易出错的时候。另一个办法是在算法设计出来以后直接针对算法进行分析，估计它的时间效率和空间开销。由于运行时间等因素和计算机运行速度有关，所以我们只关心在问题规模扩大时时空开销的增长情况。再次强调，我们很少进行程序分析，一般只进行算法分析。由算法描述写出高效程序属于程序设计方法的范畴，这不是我们讨论的重点。

基本操作　为了进行这样的估计，首先给这个估计问题建立合适的模型。用变量 n 表示输入问题的规模，而定义“基本操作”为一个运行时间不依赖于操作数的操作。

考虑两个正整数相加的操作 add。如果是两个不大于 100 的正整数相加，那么运行时间总是常数；如果两个数的大小没有限制，那么操作依赖于两个数的位数。因此可以把情况一中的 add 看作基本操作，情况二中的 add 就不能看成基本操作（这时我们一般把每一位相加看成一个基本操作，它的运行时间是常数）。后面将看到，在不同的时候，把不同的操作看成基本操作。基本操作的选择反映了对问题的主要矛盾的认识。

可以用程序执行的基本操作数来衡量它的时间效率。例如下面程序一[①]：

[①] 这是一段 C 程序。作者假定本书的读者熟悉 C 语言程序设计

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

如果把一次加法操作看成一个乘法操作，那么程序共执行了 n 个基本操作。注意，这里忽略了循环时改变变量 i 所花费的时间，而只关心和算法最有关系的基本操作。再看程序二：

```
sum = 0;
for(i = 1; i <= n; i++)
for(j = 1; j <= n; j++)
   sum += a[i][j];
```

它执行了 n^2 个基本操作。

两个程序哪个快呢？不知道。因为不知道加法操作和乘法操作哪个快。假设加法速度是乘法的 10 倍，那么哪个程序快呢？您一定会说：不一定。当 $n \leqslant 10$ 是程序二快，而 $n>10$ 时才是程序一快。这样说有道理。但是算法分析的结论会是：程序一的渐进时间复杂度低，因此更优。理由是：当规模扩大 10 倍时程序一的运行时间只扩大 10 倍，而程序二会扩大 100 倍。在这里，忽略了常数因子，因为我们只对增长情况感兴趣。下面不加说明地给出一些定义，请读者阅读相关书籍来了解这方面的详细内容。

复杂度分析的常用符号　若存在两个正常数 c 和 n_0，对任意 $n>n_0$ 都有 $|T(n)| \leqslant c|f(n)|$，则称 $T(n)$ 在集合 $O(f(n))$ 中。记作 $T(n)=O(f(n))$。O 读作“大欧”。它的直观含义是：$T(n)$ 的增长速度不会比 $f(n)$ 差。**大 O 表示法（big-Oh notation）**表示的是运行时间的上限。同理可以定义下限 Ω。如果上下限相等，还可以用 Θ 表示。本书主要只采用大 O 表示法。由于进行了简化，把大 O 表示法衡量的运行时间称为“**渐进时间复杂度（asymptotic time complexity）**”。事实上，还有**小 o 表示法**，表示这个上限是“松”的[①]。空间也可以用它来表示，但是由于在实际问题中，空间需求的一点点差异就可能引起由“存得下”到“存不下”的质的变化，所以空间复杂度常常需要用非渐进形式的精确表示。在这样的情况下，读者应当对一些程序设计细节进行了解，如各种数据类型的空间占用，动态结构的指针附加空间等，这些不是本书的讨论重点，故不展开讨论。

简化法则　利用这个定义可以把运行时间简化。简化的方法是忽略常数和低次项，例如 $3n^4+8n^2+n+2=O(n^4)$。这样做简化以后虽然忽略了一些因素，但是更容易看出算法的时间效率增长情况。如果读者怀疑这种简化法则的合理性，请参考任何一本算法与数据结构的书籍。

复杂度的等级　算法的渐进复杂度有几个等级的。一是多项式算法，例如 $O(n^t)$，t 是常数，它的运行时间随着规模的扩大增长不大，因此叫做**有效算法**；二是指数级算法，如

[①] 有兴趣的读者可以在任何一本微积分课本上学到小 o 表示法