

国际大学生 程序设计竞赛例题解 (四)

广东省信息学奥林匹克竞赛试题
(2003-2006年)

郭嵩山 张惠东 林祺颖 莫瑜 著

含光盘1张



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

国际大学生程序设计竞赛例题解

(四)

广东省信息学奥林匹克竞赛试题
(2003—2006 年)

郭嵩山 张惠东 林祺颖 莫瑜 著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书收录了《国际大学生程序设计竞赛例题解》系列书前三册中未出现过的 2003—2006 年广东省信息学奥林匹克竞赛 (GDKOI, GDOI) 全部试题、完整的测试数据和答案。为了方便读者学习, 本书对每个题目作了详尽的题目分析, 并详细地讲解其算法实现的原理, 同时提供了完善的参考程序及其程序分析。书中还提供了竞赛时评判用的基本测试数据, 以便读者测试自行完成上述题目的结果。随书附带的光盘中存放了竞赛时评判用的全部测试数据, 以便有更高、更严格要求的同学能利用规模更大的测试数据进行训练和学习。

本书所提供的题目均为原创题, 题目构思新颖, 所涉及的算法知识面广, 其涉及的算法知识基本上覆盖大学计算机类本科专业所学到的基本算法。本书可以作为高等院校有关专业的研究生和本科学生参加国际大学生程序设计竞赛的辅导教材, 也可作为高等院校有关专业相关课程的教学参考书和例题集, 并且适于作为中学青少年信息学奥林匹克竞赛省级及省级以上优秀选手备战信息学奥林匹克竞赛的培训教材及训练题集。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有, 侵权必究。

图书在版编目 (CIP) 数据

国际大学生程序设计竞赛例题解.4, 广东省信息学奥林匹克竞赛试题: 2003~2006/郭嵩山等著. —北京: 电子工业出版社, 2008.2

ISBN 978-7-121-05781-6

I. 国… II. 郭… III. 程序设计-竞赛-高等学校-解题 IV. TP311.1-44

中国版本图书馆 CIP 数据核字 (2008) 第 003920 号

责任编辑: 龚立堇

印 刷: 北京市李史山胶印厂
装 订:

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 15.75 字数: 403 千字

印 次: 2008 年 2 月第 1 次印刷

印 数: 5000 册 定价: 29.00 元 (含光盘 1 张)

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前 言

ACM 国际大学生程序设计竞赛 (ACM International Collegiate Programming Contest, 简称 ACM/ICPC) 是由国际计算机界历史悠久、颇具权威性的组织 ACM 学会 (Association for Computer Machinery) 主办, 世界上公认的规模最大、水平最高的国际大学生程序设计竞赛, 其目的是使大学生运用计算机来充分展示自己分析问题和解决问题的能力。该项竞赛从 1970 年举办至今已历 31 届, 因历届竞赛都荟萃了世界各大洲的精英, 云集了计算机界的“希望之星”, 而受到国际各知名大学的重视, 并受到全世界各著名计算机公司的高度关注, 成为世界各国大学生最具影响力的国际级计算机类的赛事。ACM 所颁发的获奖证书也为世界各著名计算机公司、各知名大学所认可。该项竞赛分区域预赛和世界决赛两个阶段进行, 各预赛区第 1 名获得参加世界决赛的资格, 世界决赛安排在每年的 3—4 月举行, 而区域预赛于上一年的 9—12 月在各大洲举行。ACM/ICPC 的区域预赛是规模很大、范围很广的赛事, 以 2006 年为例, 全世界有超过 1700 所大学的 6099 支参赛队, 在六大洲的 82 个国家 (或地区) 的 35 个赛站中争夺全球总决赛的 88 个名额, 其激烈程度可想而知。

与其他编程竞赛相比, ACM/ICPC 题目难度更大, 更强调算法的高效性, 不仅要解决一个指定的命题, 而且必须要以最佳的方式解决指定的命题。它涉及知识面广, 与大学计算机系本科及研究生的相关课程直接关联, 如程序设计、离散数学、数据结构、人工智能、算法分析与设计等, 对数学要求更高, 由于采用英文命题, 对英语要求较高。ACM/ICPC 采用 3 人合作, 共用一台电脑, 所以它更强调团队协作精神。由于许多题目并无现成的算法, 需要具备创新的精神, ACM/ICPC 不仅强调学科的基础, 更强调全面素质和能力的培养。由于 ACM/ICPC 是采用 5 小时全封闭式竞赛, 参赛队员与外界完全隔离, 独立完成, 是其实际能力的真实表露, 其成绩可信度甚高。ACM/ICPC 又是一种“开卷考试”, 可以携带任何书籍、资料, 甚至源程序代码清单 (但不能携带电子媒体), 不需要死背算法, 而强调的是算法的灵活运用。与其他计算机竞赛 (如软件设计、网站设计等) 相比, ACM/ICPC 有严谨而客观的评判规则 (严格的数据测试), 排除了因评委的主观因素而造成评审不公平的现象。所以, 对 ACM/ICPC 成绩的争议较少。

中山大学自 1997 年首次参加 ACM/ICPC 亚洲区预赛以来的 11 年中, 每年都派出多支参赛队, 共参加过 37 次亚洲区预赛, 成绩有 33 次在前 6 名之列 (有 4 次在前 10 名), 其中有 21 次进入三甲, 夺得 4 次冠军 (1999 年台北, 2002、2003 年高雄, 2007 年岷港)、6 次亚军 (2000 年香港、筑波, 2003 年北京、广州, 2006 年河内, 2007 年首尔)、11 次季军 (1998—2000 年上海, 2001 年达卡, 2002 年北京, 2003 年高雄, 2004 年马尼拉, 2005 年台北、北京, 2006 年首尔, 2007 年成都)。中山大学的参赛队曾 8 次进入全球总决赛 (1999—2001 年、2003—2007 年): 2000 年在美国佛罗里达州奥兰多市举行的第 24 届全球总决赛中夺得了第 11 名的好成绩; 2001 年在加拿大温哥华市举行的第 25 届全球总决赛中首获铜牌 (世界第 14 名); 2003 年在美国洛杉矶市好莱坞举行的第 27 届全球总决赛中获得世

界第 8 名并首获银牌的好成绩, 跻身世界八强之列; 2004 年在捷克布拉格市举行的第 28 届全球总决赛中获得世界第 11 名并再获铜牌, 且在中国内地高校中排名第一; 2005 年在上海举行的第 29 届全球总决赛中获得世界第 17 名; 2006 年在美国得克萨斯州圣安东尼奥市举行的第 30 届全球总决赛中获得世界第 19 名; 2007 年在日本东京市举行的第 31 届全球总决赛中获得世界第 26 名, 并将于 2008 年 4 月赴加拿大艾伯塔省班夫市参加第 32 届全球总决赛。

为了帮助备战国际大学生程序设计竞赛的高等院校学生们提高程序设计水平, 培养他们更强的分析问题与解决问题的能力, 我们编写了这套《国际大学生程序设计竞赛例题解》。本书是这套书的第四册, 编程所用的语言版本是 Microsoft Visual C++ 6.0 或 Free Pascal V1.0.10。全书共分 7 章, 收录了在这套《国际大学生程序设计竞赛例题解》前三册未出现过的 2003—2006 年广东省信息学奥林匹克竞赛 (GDKOI, GDOI) 的全部试题、完整的测试数据和答案。为了方便读者学习, 本书对每个题目作了详尽的题目分析, 并详细地讲解其算法实现的原理, 同时提供了完善的参考程序及其程序分析。书中提供了基本测试数据, 以方便读者测试自行完成上述题目的结果。随书附带的光盘中存放了所有例题中完整的测试数据, 以便有更高、更严格要求的同学能利用规模更大的测试数据进行训练和学习。

参与 2003—2006 年广东省信息学奥林匹克竞赛 (GDKOI, GDOI) 命题的有莫瑜、关沛勇、苏孟豪、林祺颖、梁锋、陈实、王嘉恒、张子臻、杭啸、张惠东、王磊、吴毅、刘祖立、梁志荣等, 他们大部分都是硕士研究生, 都是参加过全球总决赛或亚洲多个赛站区域预赛并取得很好成绩的中山大学队的主力队员。

本书由中山大学郭嵩山教授和他指导的 3 位研究生撰写。郭嵩山老师是国际大学生程序设计竞赛中山大学队的主教练, 张惠东、林祺颖、莫瑜是参加过亚洲多个赛站区域预赛并取得很好成绩的中山大学队的主力队员。我们期望能将自己的知识、经验、心得和体会, 奉献给广大的程序设计爱好者, 以便与大家共同探讨和交流。

本书所提供的题目均为原创题, 题目构思新颖, 背景有趣, 所涉及的算法知识面广, 其涉及的算法知识基本上覆盖大学计算机类本科专业所学到的基本算法。本书可以作为高等院校大学生和研究生准备参加各级国际大学生程序设计竞赛活动的辅导教材和训练题集, 也可以作为高等院校研究生和本科高年级学生学习相关课程的参考书, 还可以作为中学省级及省级以上信息学奥林匹克优秀选手, 准备高层次程序设计竞赛的参考用书。

由于我们水平所限, 书中难免有不足之处, 欢迎读者批评指正, 谢谢!

作 者

2007 年 11 月

目 录

本书试题涉及知识点的说明	1
第 1 章 GDKOI 2003 及 GDOI 2003 试题选解	4
1.1 最大公共子串 (难度: ★★☆☆☆)	4
1.1.1 试题	4
1.1.2 题目分析及算法实现	4
1.1.3 参考程序及程序分析	5
1.1.4 测试数据及输出结果	7
1.2 圆桌会议 (难度: ★★★★★☆)	7
1.2.1 试题	7
1.2.2 题目分析及算法实现	8
1.2.3 参考程序及程序分析	8
1.2.4 测试数据及输出结果	12
1.3 可怜的蜘蛛 (难度: ★★★★★☆)	13
1.3.1 试题	13
1.3.2 题目分析及算法实现	15
1.3.3 参考程序及程序分析	15
1.3.4 测试数据及输出结果	19
1.4 星际争霸 (难度: ★★★★★☆)	20
1.4.1 试题	20
1.4.2 题目分析及算法实现	21
1.4.3 参考程序及程序分析	22
1.4.4 测试数据及输出结果	24
1.5 求值 (难度: ★☆☆☆☆)	25
1.5.1 试题	25
1.5.2 题目分析及算法实现	25
1.5.3 参考程序及程序分析	26
1.5.4 测试数据及输出结果	27
第 2 章 GDKOI 2004 试题选解	28
2.1 石子游戏 (难度: ★★☆☆☆)	28
2.1.1 试题	28
2.1.2 题目分析及算法实现	29
2.1.3 参考程序及程序分析	29
2.1.4 测试数据及输出结果	29

2.2	汉诺塔 (难度: ★★☆☆☆)	30
2.2.1	试题	30
2.2.2	题目分析及算法实现	30
2.2.3	参考程序及程序分析	31
2.2.4	测试数据及输出结果	32
2.3	城市统计 (难度: ★★☆☆☆)	32
2.3.1	试题	32
2.3.2	题目分析及算法实现	33
2.3.3	参考程序及程序分析	34
2.3.4	测试数据及输出结果	36
2.4	使命的召唤 (难度: ★★★★★)	37
2.4.1	试题	37
2.4.2	题目分析及算法实现	38
2.4.3	参考程序及程序分析	38
2.4.4	测试数据及输出结果	41
2.5	AWP (难度: ★★☆☆☆)	41
2.5.1	试题	41
2.5.2	题目分析及算法实现	42
2.5.3	参考程序及程序分析	42
2.5.4	测试数据及输出结果	44
第3章	GDOI 2004 试题选解	45
3.1	软件工程 (难度: ★☆☆☆☆)	45
3.1.1	试题	45
3.1.2	题目分析及算法实现	46
3.1.3	参考程序及程序分析	46
3.1.4	测试数据及输出结果	47
3.2	词法分析程序 (难度: ★★☆☆☆)	47
3.2.1	试题	47
3.2.2	题目分析及算法实现	49
3.2.3	参考程序及程序分析	49
3.2.4	测试数据及输出结果	52
3.3	多项式运算 (难度: ★★☆☆☆)	53
3.3.1	试题	53
3.3.2	题目分析及算法实现	54
3.3.3	参考程序及程序分析	54
3.3.4	测试数据及输出结果	57
3.4	城市规划 (难度: ★★★★★)	58
3.4.1	试题	58
3.4.2	题目分析及算法实现	58

3.4.3	参考程序及程序分析	59
3.4.4	测试数据及输出结果	60
3.5	烟花的寿命 (难度: ★★★★★☆)	61
3.5.1	试题	61
3.5.2	题目分析及算法实现	62
3.5.3	参考程序及程序分析	62
3.5.4	测试数据及输出结果	64
3.6	中奖的机率 (难度: ★★★★★☆)	65
3.6.1	试题	65
3.6.2	题目分析及算法实现	65
3.6.3	参考程序及程序分析	66
3.6.4	测试数据及输出结果	67
3.7	课程的安排 (难度: ★★★★★★)	67
3.7.1	试题	67
3.7.2	题目分析及算法实现	68
3.7.3	参考程序及程序分析	69
3.7.4	测试数据及输出结果	73
第 4 章	GDKOI 2005 试题解	75
4.1	二叉树 (难度: ★☆☆☆☆)	75
4.1.1	试题	75
4.1.2	题目分析及算法实现	76
4.1.3	参考程序及程序分析	77
4.1.4	测试数据及输出结果	77
4.2	商人的宣传 (难度: ★★★★★☆)	78
4.2.1	试题	78
4.2.2	题目分析及算法实现	79
4.2.3	参考程序及程序分析	79
4.2.4	测试数据及输出结果	81
4.3	建立航道 (难度: ★★★★★☆)	82
4.3.1	试题	82
4.3.2	题目分析及算法实现	82
4.3.3	参考程序及程序分析	84
4.3.4	测试数据及输出结果	85
4.4	打开遗迹之门 (难度: ★★★★★☆)	85
4.4.1	试题	85
4.4.2	题目分析及算法实现	87
4.4.3	参考程序及程序分析	87
4.4.4	测试数据及输出结果	92
4.5	信号分析 (难度: ★★★★★☆)	93

4.5.1	试题	93
4.5.2	题目分析及算法实现	94
4.5.3	参考程序及程序分析	94
4.5.4	测试数据及输出结果	95
4.6	脱离地牢 (难度: ★★★☆☆)	96
4.6.1	试题	96
4.6.2	题目分析及算法实现	97
4.6.3	参考程序及程序分析	97
4.6.4	测试数据及输出结果	99
4.7	野外行军 (难度: ★★★☆☆)	100
4.7.1	试题	100
4.7.2	题目分析及算法实现	101
4.7.3	参考程序及程序分析	101
4.7.4	测试数据及输出结果	102
4.8	国王的勇士 (难度: ★★★☆☆)	103
4.8.1	试题	103
4.8.2	题目分析及算法实现	105
4.8.3	参考程序及程序分析	105
4.8.4	测试数据及输出结果	110
第 5 章	GDOI 2005 试题解	111
5.1	积木分发 (难度: ★☆☆☆☆)	111
5.1.1	试题	111
5.1.2	题目分析及算法实现	112
5.1.3	参考程序及程序分析	112
5.1.4	测试数据及输出结果	113
5.2	飞越原野 (难度: ★★★☆☆)	113
5.2.1	试题	113
5.2.2	题目分析及算法实现	114
5.2.3	参考程序及程序分析	115
5.2.4	测试数据及输出结果	117
5.3	山海经 (难度: ★★★★★)	117
5.3.1	试题	117
5.3.2	题目分析及算法实现	118
5.3.3	参考程序及程序分析	119
5.3.4	测试数据及输出结果	122
5.4	河床 (难度: ★★☆☆☆)	122
5.4.1	试题	122
5.4.2	题目分析及算法实现	123
5.4.3	参考程序及程序分析	123

5.4.4	测试数据及输出结果	124
5.5	电路稳定性 (难度: ★★★☆☆)	125
5.5.1	试题	125
5.5.2	题目分析及算法实现	125
5.5.3	参考程序及程序分析	126
5.5.4	测试数据及输出结果	127
5.6	寻宝之旅 (难度: ★★★☆☆)	127
5.6.1	试题	127
5.6.2	题目分析及算法实现	128
5.6.3	参考程序及程序分析	128
5.6.4	测试数据及输出结果	131
5.7	东东的游戏 (难度: ★★★★★)	131
5.7.1	试题	131
5.7.2	题目分析及算法实现	133
5.7.3	参考程序及程序分析	134
5.7.4	测试数据及输出结果	136
5.8	国王的遗产 (难度: ★★★★★☆)	136
5.8.1	试题	136
5.8.2	题目分析及算法实现	137
5.8.3	参考程序及程序分析	138
5.8.4	测试数据及输出结果	141
5.9	猴子的争斗 (难度: ★★★☆☆)	141
5.9.1	试题	141
5.9.2	题目分析及算法实现	142
5.9.3	参考程序及程序分析	143
5.9.4	测试数据及输出结果	144
5.10	排序 (难度: ★★★★★☆)	144
5.10.1	试题	144
5.10.2	题目分析及算法实现	145
5.10.3	参考程序及程序分析	145
5.10.4	测试数据及输出结果	147
5.11	双向环形网络 (难度: ★★★★★)	148
5.11.1	试题	148
5.11.2	题目分析及算法实现	149
5.11.3	参考程序及程序分析	150
5.11.4	测试数据及输出结果	153
5.12	选址 (难度: ★★★★★)	153
5.12.1	试题	153
5.12.2	题目分析及算法实现	154

5.12.3	参考程序及程序分析	155
5.12.4	测试数据及输出结果	157
第 6 章	GDKOI 2006 试题选解	158
6.1	纪念邮票 (难度: ★☆☆☆☆)	158
6.1.1	试题	158
6.1.2	题目分析及算法实现	158
6.1.3	参考程序及程序分析	159
6.1.4	测试数据及输出结果	159
6.2	新红黑树 (难度: ★★★☆☆)	160
6.2.1	试题	160
6.2.2	题目分析及算法实现	160
6.2.3	参考程序及程序分析	161
6.2.4	测试数据及输出结果	162
6.3	破译密文 (难度: ★★★☆☆)	163
6.3.1	试题	163
6.3.2	题目分析及算法实现	164
6.3.3	参考程序及程序分析	165
6.3.4	测试数据及输出结果	168
6.4	单词选择 (难度: ★★☆☆☆)	168
6.4.1	试题	168
6.4.2	题目分析及算法实现	169
6.4.3	参考程序及程序分析	169
6.4.4	测试数据及输出结果	172
6.5	防御力场 (难度: ★★★☆☆)	173
6.5.1	试题	173
6.5.2	题目分析及算法实现	174
6.5.3	参考程序及程序分析	174
6.5.4	测试数据及输出结果	176
6.6	速配游戏 (难度: ★☆☆☆☆)	176
6.6.1	试题	176
6.6.2	题目分析及算法实现	177
6.6.3	参考程序及程序分析	178
6.6.4	测试数据及输出结果	179
第 7 章	GDOI 2006 试题解	180
7.1	伊甸园 (难度: ★★☆☆☆)	180
7.1.1	试题	180
7.1.2	题目分析及算法实现	181
7.1.3	参考程序及程序分析	182
7.1.4	测试数据及输出结果	183

7.2	比武大会 (难度: ★★☆☆☆)	183
7.2.1	试题	183
7.2.2	题目分析及算法实现	183
7.2.3	参考程序及程序分析	184
7.2.4	测试数据及输出结果	185
7.3	道路维修 (难度: ★★★★★☆)	185
7.3.1	试题	185
7.3.2	题目分析及算法实现	186
7.3.3	参考程序及程序分析	187
7.3.4	测试数据及输出结果	190
7.4	Lie dice (难度: ★★★★★☆)	190
7.4.1	试题	190
7.4.2	题目分析及算法实现	192
7.4.3	参考程序及程序分析	192
7.4.4	测试数据及输出结果	195
7.5	亚特兰蒂斯之光 (难度: ★☆☆☆☆)	196
7.5.1	试题	196
7.5.2	题目分析及算法实现	197
7.5.3	参考程序及程序分析	197
7.5.4	测试数据及输出结果	199
7.6	外星信号 (难度: ★★★★★☆)	200
7.6.1	试题	200
7.6.2	题目分析及算法实现	200
7.6.3	参考程序及程序分析	201
7.6.4	测试数据及输出结果	203
7.7	排水系统 (难度: ★★★★★★)	204
7.7.1	试题	204
7.7.2	题目分析及算法实现	205
7.7.3	参考程序及程序分析	206
7.7.4	测试数据及输出结果	210
7.8	The kth element (难度: ★★★★★☆)	211
7.8.1	试题	211
7.8.2	题目分析及算法实现	211
7.8.3	参考程序及程序分析	212
7.8.4	测试数据及输出结果	215
7.9	古文炼成 (难度: ★★☆☆☆)	215
7.9.1	试题	215
7.9.2	题目分析及算法实现	216
7.9.3	参考程序及程序分析	217

7.9.4	测试数据及输出结果	219
7.10	拯救亚特兰蒂斯 (难度: ★★★☆☆)	219
7.10.1	试题	219
7.10.2	题目分析及算法实现	221
7.10.3	参考程序及程序分析	221
7.10.4	测试数据及输出结果	223
7.11	最大和 (难度: ★★★★★☆)	224
7.11.1	试题	224
7.11.2	题目分析及算法实现	225
7.11.3	参考程序及程序分析	225
7.11.4	测试数据及输出结果	228
7.12	天降馅饼 (难度: ★★★★★★)	228
7.12.1	试题	228
7.12.2	题目分析及算法实现	229
7.12.3	参考程序及程序分析	229
7.12.4	测试数据及输出结果	232
附录 A	GDKOI 及 GDOI 简介	233
附录 B	2003—2006 年 GDOI 及 GDKOI 试题分布	235
参考文献		238
作者简介		239

本书试题涉及知识点的说明

本书中的试题选自 2003—2006 年广东省信息学奥林匹克竞赛（GDKOI，GDOI），题型丰富，所覆盖的知识面相当广泛，包括了各种常用的算法设计方法及数据结构设计，其中有各种搜索及优化剪枝、图论、动态规划、数论、模拟、计算几何、博弈等方面的知识。

1. 涉及的算法设计方法

（1）递归和递推。递归和递推是学习算法设计的第一步。递归算法是把大问题分解成相对较小的问题的过程，而递推就是从小问题逐步推导出大问题的过程。无论递归还是递推，都应该有初始的状态。递归算法可参考本书中“2.2 汉诺塔”、“5.5 电路稳定性”和“7.1 伊甸园”；递推在各种数学推导及动态规划中常用到。

（2）搜索、枚举及优化剪枝。搜索在所有算法中既是最简单也是最复杂的算法。说它最简单，是因为算法本身并不复杂，实现容易；说它最复杂，是因为要对搜索的范围进行一定的控制，不然就会出现超时等问题。搜索技术主要包括广度优先搜索和深度优先搜索。当其余算法都无法对问题进行求解时，搜索或许是惟一可用的方法。搜索是对问题的解空间进行遍历的过程。有时问题解空间相当庞大，完全遍历解空间是不现实的，此时就必须充分发掘问题所包含的约束条件，在搜索过程中应用这些条件进行剪枝，从而减少搜索量。参考本书中“1.2 圆桌会议”、“4.4 打开遗迹之门”、“5.2 飞越原野”、“5.10 排序”、“7.3 道路维修”和“7.9 古文炼成”等。

（3）动态规划（简称 DP）。动态规划的特点是能够把很复杂的问题分解成一个个阶段来处理的递推方法。动态规划必须符合两个特点：无后效性（一个状态的抉择不会影响到更大问题的状态的抉择）及最优化原理（一个大问题的最优性必须建立在其子问题的最优性之上）。动态规划是竞赛中经常出现的类型，而且变化很大（有线性 DP，环形 DP，树形 DP 等），难易跨度大，技巧性强，甚至还有 DP 的优化等问题。参考本书中“1.1 最大公共子串”、“2.3 城市统计”、“4.6 脱离地牢”、“5.4 河床”、“5.6 寻宝之旅”和“7.2 比武大会”等。

（4）贪心。贪心算法是所谓的“只顾眼前利益”的算法。其具体策略是并不从整体最优上加以考虑，而是选取某种意义下的局部最优解。当然使用贪心时，要使得到的结果也是整体最优的。参考本书中“3.1 软件工程”、“5.1 积木分发”、“5.11 双向环形网络”等。

（5）分治、构造等。分治就是把问题分成若干子问题，然后“分而治之”；构造是指按照一定的规则产生解决问题的方法。两种算法都是在合理的分析题目后，通过一定的规律性推导，从而解决问题。快速排序可以认为是利用了分治法。分治算法可参考本书中“1.5 求值”，其本质为排序；构造算法可参考本书中“3.7 课程的安排”。

2. 数据结构

算法和数据结构是计算机程序中最基本的两个部分，因此当进行程序设计时，或多或少的使用各种数据结构。最基本的数据结构为线性表（数组，链表，队列，堆栈等），二叉树，树，图等。由此引伸出来的双向链表，最小堆，AVL 树，红黑树，胜者树，并查集，Hash 表，等等。它们能够在相对复杂的算法中大大提高处理速度。例如，在本书中“3.3 多项式运算”使用到链表，“4.1 二叉树”就要对二叉树有很深的了解，而“6.4 单词选择”、“7.8 The kth element”和“7.11 最大和”等就用到更高级的数据结构或者要自己设计一个数据结构了。

3. 图论

图论中的知识点相当广泛，除了最基本的广度优先遍历、深度优先遍历，稍复杂的最短路径问题、拓扑排序、Floyd 算法，更复杂的匹配问题、网络流问题。比赛中的图论问题多数是由各种基本问题通过一定的变化来解决问题。本书中涉及到以下基本图论问题：

- (1) 网络流，如“1.4 星际争霸”。
- (2) 最小路径覆盖引申到二分图最大基数匹配，如“2.4 使命的召唤”。
- (3) Floyd 算法，如“3.4 城市规划”。
- (4) 树的最长路问题：如“3.5 烟花的寿命”。
- (5) L 步后不同路线数（邻接矩阵相乘），如“4.2 商人的宣传”。
- (6) 深度优先遍历，如“5.8 国王的遗产”。
- (7) 最小割转化为网络流，如“6.3 破译密文”。
- (8) 最短路的修改，如“7.7 排水系统”。
- (9) 二分图的匹配，如“7.10 拯救亚特兰蒂斯”。

4. 数学题

数学题包括组合数学、数论、高等数学（概率统计）等。数学问题一般需要通过一定的推导计算，或者从问题中发现其数学规律，从而解决问题。参考本书中“3.6 中奖的机率”、“4.3 建立航道”、“4.5 信号分析”、“5.7 东东的游戏”、“5.9 猴子的争斗”、“6.1 纪念邮票”和“7.6 外星信号”等。

5. 计算几何

计算几何中的知识点也相当多。试题常涉及基本的平面几何知识（极少用到立体几何），如两点间最近距离，直线方程，三角形各心，点、线、面之间的关系，坐标旋转等，还包括求凸包，最近点对，卡壳旋转，二分角度等。参考本书中“2.5 AWP”、“4.7 野外行军”、“5.12 选址”、“6.5 防御力场”、“7.5 亚特兰蒂斯之光”、“7.12 天降馅饼”等。

6. 博弈

博弈题主要是游戏中的致胜策略问题，以及 min-max 剪枝问题。参考本书中“2.1 石子游戏”、“6.2 新红黑树”、“7.4 Lie dice”等。

7. 模拟

模拟题一般只需按照题目要求细心编程即可，如本书中“1.3 可怜的蜘蛛”、“3.2 词

法分析程序”、“4.8 国王的勇士”、“6.6 速配游戏”等；少部分需要一定的优化，如本书中“7.4 Lie dice”。

8. 其他

本书题目还包括其他的知识点，例如 RMQ 问题，参考本书中“5.3 山海经”。

第 1 章 GDKOI 2003 及 GDOI 2003 试题选解

1.1 最大公共子串（难度：★★☆☆☆）

1.1.1 试题

此题为 GDKOI 2003 一试第一题。

问题描述

从一个给定的串中删去（不一定连续地删去）0 个或 0 个以上的字符，剩下的字符按原来顺序组成的串是该串的子串。例如：“”，“a”，“xb”，“aaa”，“bbb”，“xabb”，“xaaabbb”都是串“xaaabbb”的子串。（例子中的串不包含引号。）

编程求 N 个非空串的最长公共子串的长度。限制： $2 \leq N \leq 100$ ； N 个串中的字符只会是数字 0, 1, ..., 9 或小写英文字母 a, b, ..., z；每个串非空且最多含 100 个字符； N 个串的长度的乘积不会超过 30000。

输入格式（输入文件名 maxsub.in）

文件第 1 行是一个整数 T ，表示测试数据的个数（ $1 \leq T \leq 10$ ）。接下来有 T 组测试数据。各组测试数据的第 1 行是一个整数 N_i ，表示第 i 组数据中串的个数。各组测试数据的第 2 到 N_i+1 行中，每行一个串，串中不会有空格，但行首和行末可能有空格，这些空格当然不算作串的一部分。

输出格式（输出文件名 maxsub.out）

输出 T 行，每行一个数，第 i 行的数表示第 i 组测试数据中 N_i 个非空串的最长公共子串的长度。

输入输出样例

输入（maxsub.in）	输出（maxsub.out）
1	0
3	
ab	
bc	
cd	

1.1.2 题目分析及算法实现

分析题目，这道题与传统的最长公共子串题目略有不同，传统的公共子串是要求孩子