

国际大学生 程序设计竞赛例题解 (六)

广东省大学生程序设计竞赛试题
(2008 - 2009年)

郭嵩山 翁雨键 梁志荣 吴毅 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

(六)

广东省大学生程序设计竞赛试题

(2008—2009 年)

郭嵩山 翁雨健 梁志荣 吴毅 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 2008—2009 年广东省大学生程序设计竞赛（GDCPC 2008—2009，即 ACM/ICPC 广东省赛）及第 33 届 ACM/ICPC 亚洲成都赛区预选赛的全部试题、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，供读者参考。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。随书附带的光盘存放了所有例题中完整的测试数据，以便于有更高、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。

本书所提供的题目都是原创题，题目构思新颖，内容有趣。所涉及的算法知识面广，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，同时还可以作为中学省级及以上信息学奥林匹克优秀选手备战高层次程序设计竞赛的参考用书。

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. 6, 广东省大学生程序设计竞赛试题: 2008~2009 年 / 郭嵩山等著.
—北京: 电子工业出版社, 2010.5

ISBN 978-7-121-10727-6

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

中国版本图书馆 CIP 数据核字 (2010) 第 070411 号

策划编辑: 龚立堃

责任编辑: 王玉国 文字编辑: 龚立堃

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: 787×1 092 1/16 印张: 21 字数: 534 千字

印 次: 2010 年 5 月第 1 次印刷

印 数: 3 000 册 定价: 39.00 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) 主办的, 是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, 其目的旨在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 32 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 IT 企业的高度关注, 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知名大学、各著名 IT 企业所认可。该项竞赛分为区域预赛和世界决赛两个阶段进行, 各预赛区第一名自动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, 而区域预赛安排在上一年度的 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 2008 年为例, 全世界有 88 个国家和地区、1838 所大学、超过 7100 支参赛队在六大洲的 37 个赛站中争夺世界决赛的 100 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, 而且必须以最佳的方式解决指定的命题。它涉及的知识面广, 与大学计算机系本科及研究生的课程直接关联, 如程序设计、离散数学、数据结构、人工智能、算法分析与设计等课程; 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、公用一台电脑, 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, 是参赛队员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”, 可以带任何书籍、资料甚至源程序代码清单 (但不能带电子媒体), 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, ACM/ICPC 对成绩的争议较少。

中山大学自 1997 年首次参加 ACM/ICPC 亚洲区预赛以来的 13 年中, 每年都派出多支队伍共参加过 52 次亚洲区预赛, 成绩有 45 次排在前 6 名, 6 次排在前 10 名, 1 次排在前 12 名; 其中有 25 次进入三甲, 夺得 5 次冠军 (1999 年台北, 2002、2003 年高雄, 2007 年岷港, 2009 年合肥)、9 次亚军 (2000 年香港、筑波, 2003 年北京、广州, 2006 年河内, 2007 年首尔, 2008 年雅加达、首尔, 2009 年宁波)、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 11 次进入全球总决赛 (1999—2001 年、2003—2008 年): 2000 年在美国佛罗里达州奥兰多市举行的第 24 届全球总决赛中取得了第 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 届全球总决赛中取得世界第 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 年在上海市举行的第 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；2008 年在加拿大班夫市举行的第 32 届全球总决赛中获得世界第 23 名；2009 年在瑞典斯德哥尔摩市举行的第 33 届全球总决赛中获得世界第 20 名；并取得将在 2010 年举行的第 34 届全球总决赛的参赛资格。

为了帮助高等院校的大学生们备战国际大学生程序设计竞赛，帮助他们提高程序设计水平和培养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《国际大学生程序设计竞赛例题解》的第六册，编程所用的语言版本是 Microsoft Visual C++6.0。全书共分 6 章，收录了 2008—2009 年广东省大学生程序设计竞赛（GDCPC 2008—2009，即 ACM/ICPC 广东省赛）的全部试题、完整的测试数据和答案，以及第 33 届 ACM/ICPC 亚洲成都赛区预选赛（由西南民族大学承办、中山大学命题）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，供读者参考。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所有例题中完整的测试数据，以便于有更高、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。读者可利用《本书试题涉及知识点的说明》作为索引，根据知识点的分类进行学习，同时可根据目录中提供试题的难易程度，循序渐进地进行学习和训练。参与上述竞赛命题的有：陈实、张惠东、张子臻、王磊、汤振东、翁雨键、梁志荣、吴毅、刘祖立、涂德健、刘曦、熊鹰扬、张钊毅等。他们大部分是硕士研究生，都是参加过世界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，翁雨键、梁志荣、吴毅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。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。

本书所提供的题目全部为原创题，题目构思新颖、内容有趣，涉及的算法知识面广，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及训练题集，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，并适于作为中学省级及以上信息学奥林匹克优秀选手备战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
2010 年 4 月

目 录

本书试题涉及知识点的说明	1
第 1 章 第 11 届中山大学程序设计竞赛预选赛 (2008 年) 试题分析	3
1.1 矩阵归零 (难度: ★★☆☆☆)	3
1.1.1 试题	3
1.1.2 题目分析与算法实现	4
1.1.3 参考程序与程序分析	5
1.1.4 部分测试数据与输出结果	5
1.2 基因 (难度: ★★★★★)	6
1.2.1 试题	6
1.2.2 题目分析与算法实现	7
1.2.3 参考程序与程序分析	8
1.2.4 部分测试数据与输出结果	11
1.3 旅行 (难度: ★★★★★)	12
1.3.1 试题	12
1.3.2 题目分析与算法实现	13
1.3.3 参考程序与程序分析	14
1.3.4 部分测试数据与输出结果	16
1.4 单词缩写 (难度: ★☆☆☆☆)	16
1.4.1 试题	16
1.4.2 题目分析与算法实现	17
1.4.3 参考程序与程序分析	17
1.4.4 部分测试数据与输出结果	18
1.5 围棋 (难度: ★★★★★)	19
1.5.1 试题	19
1.5.2 题目分析与算法实现	24
1.5.3 参考程序与程序分析	25
1.5.4 部分测试数据与输出结果	28
1.6 字符串序列 (难度: ★★★★★)	32
1.6.1 试题	32
1.6.2 题目分析与算法实现	32
1.6.3 参考程序与程序分析	34
1.6.4 部分测试数据与输出结果	34
1.7 DNA 重组 (难度: ★★★★★)	35
1.7.1 试题	35
1.7.2 题目分析与算法实现	36

1.7.3	参考程序与程序分析	38
1.7.4	部分测试数据与输出结果	39
1.8	小实的问题 (难度: ★☆☆☆☆)	40
1.8.1	试题	40
1.8.2	题目分析与算法实现	40
1.8.3	参考程序与程序分析	41
1.8.4	部分测试数据与输出结果	41
1.9	总体题目分析与比赛情况	41
1.9.1	题目分析	41
1.9.2	比赛情况	42
第 2 章	第 6 届广东省大学生程序设计竞赛暨第 11 届中山大学程序设计竞赛 (2008 年)	
	试题分析	43
2.1	表面积 (难度: ★☆☆☆☆)	43
2.1.1	试题	43
2.1.2	题目分析与算法实现	44
2.1.3	参考程序与程序分析	44
2.1.4	部分测试数据与输出结果	44
2.2	读书 (难度: ★★☆☆☆)	45
2.2.1	试题	45
2.2.2	题目分析与算法实现	46
2.2.3	参考程序与程序分析	47
2.2.4	部分测试数据与输出结果	48
2.3	椰子 (难度: ★★★☆☆)	49
2.3.1	试题	49
2.3.2	题目分析与算法实现	51
2.3.3	参考程序与程序分析	51
2.3.4	部分测试数据与输出结果	53
2.4	烙饼 (难度: ★★★★★)	54
2.4.1	试题	54
2.4.2	题目分析与算法实现	55
2.4.3	参考程序与程序分析	57
2.4.4	部分测试数据与输出结果	59
2.5	子集 (难度: ★★★★★)	60
2.5.1	试题	60
2.5.2	题目分析与算法实现	61
2.5.3	参考程序与程序分析	63
2.5.4	部分测试数据与输出结果	66
2.6	风水 (难度: ★★★☆☆)	67
2.6.1	试题	67
2.6.2	题目分析与算法实现	69

2.6.3	参考程序与程序分析	70
2.6.4	部分测试数据与输出结果	72
2.7	危险游戏 (难度: ★★★★★☆)	73
2.7.1	试题	73
2.7.2	题目分析与算法实现	74
2.7.3	参考程序与程序分析	75
2.7.4	部分测试数据与输出结果	79
2.8	并集 (难度: ★★★★★☆)	80
2.8.1	试题	80
2.8.2	题目分析与算法实现	81
2.8.3	参考程序与程序分析	82
2.8.4	部分测试数据与输出结果	86
2.9	乒乓球 (难度: ★☆☆☆☆)	87
2.9.1	试题	87
2.9.2	题目分析与算法实现	89
2.9.3	参考程序与程序分析	89
2.9.4	部分测试数据与输出结果	90
2.10	单轮比赛 (难度: ★☆☆☆☆)	91
2.10.1	试题	91
2.10.2	题目分析与算法实现	92
2.10.3	参考程序与程序分析	92
2.10.4	部分测试数据与输出结果	93
2.11	总体题目分析与比赛情况	93
2.11.1	题目分析	93
2.11.2	比赛情况	94
第 3 章	第 12 届中山大学程序设计竞赛预选赛 (2009 年) 试题分析	95
3.1	SZ 斐波拉契数列 (难度: ★☆☆☆☆)	95
3.1.1	试题	95
3.1.2	题目分析与算法实现	96
3.1.3	参考程序与程序分析	96
3.1.4	部分测试数据与输出结果	96
3.2	无聊的 S (难度: ★★☆☆☆)	97
3.2.1	试题	97
3.2.2	题目分析与算法实现	98
3.2.3	参考程序与程序分析	99
3.2.4	部分测试数据与输出结果	100
3.3	杀手游戏 (难度: ★★★★★☆)	101
3.3.1	试题	101
3.3.2	题目分析与算法实现	102
3.3.3	参考程序与程序分析	103

3.3.4	部分测试数据与输出结果	105
3.4	切割矩形 (难度: ★★★★★☆)	107
3.4.1	试题	107
3.4.2	题目分析与算法实现	108
3.4.3	参考程序与程序分析	109
3.4.4	部分测试数据与输出结果	112
3.5	子序列 (难度: ★★★★★☆)	113
3.5.1	试题	113
3.5.2	题目分析与算法实现	114
3.5.3	参考程序与程序分析	115
3.5.4	部分测试数据与输出结果	116
3.6	N 皇后问题 (难度: ★★★★★☆)	117
3.6.1	试题	117
3.6.2	题目分析与算法实现	118
3.6.3	参考程序与程序分析	119
3.6.4	部分测试数据与输出结果	121
3.7	MAX-2-SAT (难度: ★★★★★★)	122
3.7.1	试题	122
3.7.2	题目分析与算法实现	123
3.7.3	参考程序与程序分析	123
3.7.4	部分测试数据与输出结果	128
3.8	平方数 (难度: ★★★★★☆)	129
3.8.1	试题	129
3.8.2	题目分析与算法实现	130
3.8.3	参考程序与程序分析	131
3.8.4	部分测试数据与输出结果	132
3.9	总体题目分析与比赛情况	132
3.9.1	题目分析	132
3.9.2	比赛情况	133
第 4 章	第 7 届广东省大学生程序设计竞赛暨第 12 届中山大学程序设计竞赛 (2009 年)	
	试题分析	134
4.1	航班管理 (难度: ★☆☆☆☆)	134
4.1.1	试题	134
4.1.2	题目分析与算法实现	135
4.1.3	参考程序与程序分析	135
4.1.4	部分测试数据与输出结果	135
4.2	人员调度 (难度: ★★★★★☆)	136
4.2.1	试题	136
4.2.2	题目分析与算法实现	137
4.2.3	参考程序与程序分析	138

4.2.4	部分测试数据与输出结果	141
4.3	积木游戏 (难度: ★★★★★☆)	141
4.3.1	试题	141
4.3.2	题目分析与算法实现	143
4.3.3	参考程序与程序分析	144
4.3.4	部分测试数据与输出结果	146
4.4	嘉年华 (难度: ★★★☆☆)	148
4.4.1	试题	148
4.4.2	题目分析与算法实现	150
4.4.3	参考程序与程序分析	151
4.4.4	部分测试数据与输出结果	154
4.5	H-index 因子 (难度: ★★☆☆☆)	156
4.5.1	试题	156
4.5.2	题目分析与算法实现	157
4.5.3	参考程序与程序分析	157
4.5.4	部分测试数据与输出结果	159
4.6	梦幻西游 (难度: ★★★★★)	160
4.6.1	试题	160
4.6.2	题目分析与算法实现	162
4.6.3	参考程序与程序分析	164
4.6.4	部分测试数据与输出结果	168
4.7	二十一点 (难度: ★★★★★☆)	169
4.7.1	试题	169
4.7.2	题目分析与算法实现	170
4.7.3	参考程序与程序分析	171
4.7.4	部分测试数据与输出结果	172
4.8	大话西游 (难度: ★★★★★☆)	173
4.8.1	试题	173
4.8.2	题目分析与算法实现	174
4.8.3	参考程序与程序分析	175
4.8.4	部分测试数据与输出结果	179
4.9	计数 (难度: ★★★★★)	181
4.9.1	试题	181
4.9.2	题目分析与算法实现	182
4.9.3	参考程序与程序分析	183
4.9.4	部分测试数据与输出结果	185
4.10	黑叔 (难度: ★★☆☆☆)	185
4.10.1	试题	185
4.10.2	题目分析与算法实现	186
4.10.3	参考程序与程序分析	187

4.10.4	部分测试数据与输出结果	188
4.11	总体题目分析与比赛情况	189
4.11.1	题目分析	189
4.11.2	比赛情况	189
第 5 章	第 33 届国际大学生程序设计竞赛亚洲成都赛区网络预赛试题分析	191
5.1	星星 (难度: ★★☆☆☆)	191
5.1.1	试题	191
5.1.2	题目分析与算法实现	192
5.1.3	参考程序与程序分析	192
5.1.4	部分测试数据与输出结果	194
5.2	单词接龙 (难度: ★★★☆☆)	195
5.2.1	试题	195
5.2.2	题目分析与算法实现	196
5.2.3	参考程序与程序分析	197
5.2.4	部分测试数据与输出结果	199
5.3	打包豆豆 (难度: ★★★★★☆)	200
5.3.1	试题	200
5.3.2	题目分析与算法实现	201
5.3.3	参考程序与程序分析	202
5.3.4	部分测试数据与输出结果	204
5.4	棋盘计数问题 (难度: ★★☆☆☆)	204
5.4.1	试题	204
5.4.2	题目分析与算法实现	205
5.4.3	参考程序与程序分析	207
5.4.4	部分测试数据与输出结果	207
5.5	Farey 序列卷土重来 (难度: ★★★★★☆)	208
5.5.1	试题	208
5.5.2	题目分析与算法实现	209
5.5.3	参考程序与程序分析	211
5.5.4	部分测试数据与输出结果	213
5.6	周游 (难度: ★★★☆☆)	214
5.6.1	试题	214
5.6.2	题目分析与算法实现	215
5.6.3	参考程序与程序分析	215
5.6.4	部分测试数据与输出结果	217
5.7	UNO 纸牌游戏 (难度: ★★★★★★)	218
5.7.1	试题	218
5.7.2	题目分析与算法实现	221
5.7.3	参考程序与程序分析	222
5.7.4	部分测试数据与输出结果	226

5.8	战争传说（难度：★★★★☆）	227
5.8.1	试题	227
5.8.2	题目分析与算法实现	228
5.8.3	参考程序与程序分析	229
5.8.4	部分测试数据与输出结果	233
5.9	碰撞检测（难度：★★☆☆☆）	234
5.9.1	试题	234
5.9.2	题目分析与算法实现	234
5.9.3	参考程序与程序分析	235
5.9.4	部分测试数据与输出结果	236
5.10	跳鼠（难度：★★☆☆☆）	237
5.10.1	试题	237
5.10.2	题目分析与算法实现	239
5.10.3	参考程序与程序分析	239
5.10.4	部分测试数据与输出结果	241
5.11	总体题目分析与比赛情况	242
5.11.1	题目分析	242
5.11.2	比赛情况	243
第6章	第33届国际大学生程序设计竞赛亚洲成都赛区现场赛试题分析	244
6.1	进程调度（难度：★★★★☆）	244
6.1.1	试题	244
6.1.2	题目分析与算法实现	245
6.1.3	参考程序与程序分析	247
6.1.4	部分测试数据与输出结果	248
6.2	套盒（难度：★★★★★）	249
6.2.1	试题	249
6.2.2	题目分析与算法实现	251
6.2.3	参考程序与程序分析	256
6.2.4	部分测试数据与输出结果	261
6.3	字符串印刷机（难度：★★★★☆）	262
6.3.1	试题	262
6.3.2	题目分析与算法实现	263
6.3.3	参考程序与程序分析	264
6.3.4	部分测试数据与输出结果	265
6.4	麦田怪圈（难度：★★★★☆）	266
6.4.1	试题	266
6.4.2	题目分析与算法实现	268
6.4.3	参考程序与程序分析	270
6.4.4	部分测试数据与输出结果	273
6.5	幻灯片（难度：★★☆☆☆）	274

6.5.1	试题	274
6.5.2	题目分析与算法实现	274
6.5.3	参考程序与程序分析	275
6.5.4	部分测试数据与输出结果	277
6.6	国王联盟 (难度: ★★★★★☆)	278
6.6.1	试题	278
6.6.2	题目分析与算法实现	279
6.6.3	参考程序与程序分析	281
6.6.4	部分测试数据与输出结果	285
6.7	盗宝 (难度: ★★★★★☆)	286
6.7.1	试题	286
6.7.2	题目分析与算法实现	287
6.7.3	参考程序与程序分析	288
6.7.4	部分测试数据与输出结果	290
6.8	玩具 (难度: ★★★★★★)	291
6.8.1	试题	291
6.8.2	题目分析与算法实现	292
6.8.3	参考程序与程序分析	296
6.8.4	部分测试数据与输出结果	300
6.9	公交搜索 (难度: ★★★★★☆)	300
6.9.1	试题	300
6.9.2	题目分析与算法实现	304
6.9.3	参考程序与程序分析	304
6.9.4	部分测试数据与输出结果	307
6.10	数方块 (难度: ★★☆☆☆)	308
6.10.1	试题	308
6.10.2	题目分析与算法实现	309
6.10.3	参考程序与程序分析	309
6.10.4	部分测试数据与输出结果	311
6.11	建塔 (难度: ★★★★★★)	312
6.11.1	试题	312
6.11.2	题目分析与算法实现	313
6.11.3	参考程序与程序分析	316
6.11.4	部分测试数据与输出结果	317
6.12	总体题目分析与比赛情况	318
6.12.1	题目分析	318
6.12.2	比赛情况	319
	参考文献	320
	作者简介	321

本书试题涉及知识点的说明

本书中题型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、数论、模拟、贪心法、动态规划、计算几何等方面的知识。

1. 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。递归是将问题表示为与原问题相似但规模更小的问题，然后通过函数的自调用来求解问题。而递推是从规模较小的问题逐步推导出规模较大问题的一个迭代过程。递归算法主要包含递归终止条件和递归方程两个要素，而递推算法主要包含递推初始条件和递推方程两个要素。本书涉及递归算法的题目有“1.6 字符串序列”、“3.1 SZ 斐波拉契数列”、“3.5 子序列”及“5.2 单词接龙”。递推在各种数学推导及动态规划中经常会被用到。

(2) 搜索、枚举及优化剪枝。搜索在所有算法中是既简单又复杂的算法。说它简单，是因为搜索本身的基本框架并不复杂，容易实现；说它复杂，是因为在处理具体问题时，为了有效地减少搜索量，需要对搜索过程进行合适的优化，避免出现超时问题。搜索技术主要包括广度优先搜索和深度优先搜索。ICPC 竞赛中出现的搜索题目主要是小规模 NPC 问题。NPC 问题是指满足以下两个条件的问题：第一，它本身是一个 NP 问题；第二，所有的 NP 问题都可以归属到它。NP 问题是指可以在多项式的时间里验证一个解的问题，以便考查选手充分发掘问题隐含的约束条件及设计剪枝优化的能力。另外，由于搜索本质上是对问题解空间的一个遍历，搜索过程中需要避免重复遍历同一个状态。搜索状态如何进行合适的编码和判重往往也是搜索问题的一个重要的考查点。本书涉及搜索算法的题目有“1.2 基因”、“2.6 风水”、“3.7 MAX-2-SAT”、“4.2 人员调度”及“5.6 周游”等。

(3) 动态规划（简称 DP）。动态规划的特点是能够把复杂的问题分解成一个个阶段进行处理，并使用递推的方法来计算各个阶段的解。动态规划必须符合两个特点，即无后效性（一个状态的抉择不影响更大规模的问题状态的抉择）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）。动态规划是竞赛题中常出现的类型，具有变形多（有线性 DP、环形 DP、树形 DP 等），难易跨度大，技巧性强等特点。动态规划算法的设计主要涉及状态表示、状态转移方程、初始状态、问题的解的表示等几个要素。本书中有关动态规划算法的题目有“1.7 DNA 重组”、“4.3 积木游戏”、“4.7 二十一点”、“5.4 棋盘计数问题”、“6.3 字符串印刷机”及“6.11 建塔”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。本书涉及贪心算法的题目有“4.2 人员调度”。

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，常常会使用各种数据结构。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。由此引伸出来的数据结构有双向链表，最小堆、AVL 树、Splay 树、并查集、线段树、哈希表等。它们能够在相对复杂的算法中提高处理速度。例如在本书中，“3.4 切割矩形”是一道矩形几何统计方面的问题，进行适当的转化后可以使用线段树等数据结构解决；而“4.6 梦幻西游”和“4.8 大话西游”是两道综合题目，在进行统计或者询问最值时，需要使用树状数组或线段树；“5.1 星星”一题考查哈希表数据结构针对具体问题的设计；“5.3 打包豆豆”一题需要巧妙的设计一个单调的双向队列；“6.2 套盒”一题考查 Splay 树在复杂问题上的应用。

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，一部分可以转化为基本图论问题，然后使用经典算法解决；而另一部分则可能需要选手设计算法。本书涉及图论的题目有“1.3 旅行”、“2.2 读书”、“2.5 子集”、“2.7 危险游戏”、“3.2 无聊的 S”、“4.1 航班管理”、“4.6 梦幻西游”、“5.8 战争传说”、“5.10 跳鼠”、“6.7 盗宝”及“6.9 公交搜索”等。

4. 数论

比赛中的数学题主要考查数论、组合数学、概率统计等知识。数学问题一般需要通过一定的推导计算，或者从问题中发掘数学规律，从而解决问题。本书与数学相关的题目有“1.1 矩阵归零”、“1.8 小实的问题”、“2.8 并集”、“2.10 单轮比赛”、“3.8 平方数”、“4.9 计数”、“5.5 Farey 序列卷土重来”、“6.6 国王联盟”、“6.8 玩具”及“6.10 数方块”等。

5. 计算几何

计算几何中的知识点也相当多。除了考查基本的几何知识，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，通常还会考查凸包相关的一些算法，包括最近点对，卡壳旋转等。另外，扫描线思想的应用也是计算几何的一个考查点。计算几何的题目通常会涉及浮点数运算，选手需要处理好运算中可能产生的浮点误差。本书与计算几何有关的题目有“2.1 表面积”、“2.4 烙饼”、“2.9 兵乓球”、“4.4 嘉年华”、“5.9 碰撞检测”、“6.4 麦田怪圈”及“6.5 幻灯片”等。

6. 模拟

模拟题也是比赛中常见的一类题目，通常这类题目的难度不大，但选手需要仔细阅读题目给出的条件，准确理解题目的要求，才可以完成好题目。本书涉及模拟的题目有“1.4 单词缩写”、“1.5 围棋”、“2.3 椰子”、“3.3 杀手游戏”、“4.5 H-index 因子”、“4.10 黑叔”、“5.7 UNO 纸牌游戏”及“6.1 进程调度”等。

7. 其他

此外本书试题还涉及其他不易归入上述类型的知识点，例如“3.6 N 皇后问题”用到随机算法。

第 1 章

第 11 届中山大学程序设计竞赛预选赛（2008 年）

试题分析

1.1 矩阵归零（难度：★★☆☆☆）

1.1.1 试题

Problem A

Zero Puzzling

(Standard Input / Standard Output)

There is a matrix with m rows and n columns. An element of the matrix has at most four adjacent elements (up, down, left, right). You can add a same number to a pair of adjacent elements of the matrix. By doing this operation repeatedly, can you make the matrix zero?

Input:

A line contains m and n indicating the rows and columns of the matrix. ($0 < m, n < 5$)

Each of the following m lines contains n integers ($-10^6 \leq \text{each integer} \leq 10^6$).

Input is terminated by one line contains two zeros.

Output:

If the matrix can be converted into zero, output "Yes" on a line otherwise output "No" on a line.

Sample Input:

```
2 2
1 1
1 0
```

```
2 2
0 0
1 1
```

```
0 0
```

Sample Output:

```
No
```

1.1.2 题目分析与算法实现

本题是比赛中较为简单的一题。主要考查编程者的数学思维能力。

本题的题意为：给定一个 m 行 n 列的数字矩阵，问能否通过反复使用给定的操作，使得最终矩阵里的所有数字都变成 0。题目允许的操作是对矩阵中某对相邻数都加上一个数值。这里矩阵中的两个数是否相邻定义为它们所在的方格是否有公共边，于是每个方格最多与其上下左右四个方格相邻。其中，矩阵中的数字可能存在负数，允许操作的加数也可以是负数。题目为多测试数据输入，每个测试数据第一行为 $m\ n$ ，表示该数据是一个 m 行 n 列的矩阵，接着是 m 行，每行 n 个数字，表示矩阵每个位置的数值。测试数据输入以 0 0 结束。

本题规模不大，行数和列数的上限都不超过 5。对于 4×4 的矩阵，总共有 24 个相邻对。因此可能部分选手会往搜索的方向思考这个问题。显然，对于同一个相邻对最多只需要做一次操作，因为多次的加法操作可以合并为一次。然而尽管如此，由于矩阵中的整数规模较大，盲目的搜索每次操作需要在较大范围内枚举一个加数，由于计算量太大将导致超时。

我们可将间隔的格子分别染成黑白两色，把矩阵染色成类似于国际象棋棋盘的样子，如图 1.1.1 所示。这样染色后，每对相邻的格子，一定是一个染黑色一个染白色。令矩阵中所有白色格子的数字和为 S_1 ，所有黑色格子的数字和为 S_2 ，比如 $S_1=a+c+f+h+i+k+n+p$ ， $S_2=b+d+e+g+j+l+m+o$ 。则每次操作都会在 S_1 和 S_2 加上同一个数，因此不会改变 S_1 与 S_2 的差。由于题目要求最终所有数字为零，应有 $S_1=S_2=0$ ，它们的差也为 0。根据 S_1 与 S_2 的差不变的性质，容易知道：若初始矩阵 S_1 与 S_2 的差不为零，则一定无法达到零矩阵这个目标。

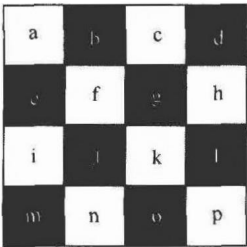


图 1.1.1 棋盘染色

这样就找到了判断能否到达零矩阵的一个必要条件。然而这是否也是充分条件呢？答案是肯定的。如果 $S_1=S_2$ ，存在简单的构造方法使用允许的操作使得矩阵中的数最终全部变为零。我们可以把数一步步的“集中”到一个格子 p 上去，假设格子 p 是白色的，此时其他格子都为零。因此有 $S_2=0$ ，根据 $S_1=S_2$ ，且白色的其他格子为零，故 p 这个格子也等于零，于是达到零矩阵的目标。

具体是这样“集中”的：对于第一行， a 与 b 同时加上 $-a$ ，则将第一个元素变为 0，第二个元素变为 $b-a$ ；然后 $b-a$ 与 c 同时加上 $a-b$ ，则将第二个元素变为 0，第三个元素变为 $c+a-b$ ；然后 $c+a-b$ 与 d 同时加上 $b-a-c$ ，则第三个元素变为 0，第四个元素变为 $d+b-a-c$ 。

类似的，对每行分别处理便可以使得只有最后一列的数字非零。然后再将最后一列按照同样的方式处理，可以将所有数“集中”到最后一个方格。根据上述证明，若 $S_1=S_2$ ，此时该方格也为零。