

TURING

图灵程序设计丛书

Short Coding

短码之美

编程达人的心得技法

- 短码编程的尖峰时刻
- 编程效率的极致追求
- 北大POJ AMC经典题目解析

[日] Ozy 著

[日] やねうらお 审校

李秀银 赛丽娜 译



人民邮电出版社

POSTS & TELECOM PRESS

TURING

图灵程序设计丛书

Short Coding

短码之美

编程达人的心得技法

[日] Ozy 著

[日] やねうらお 审校

李秀银 赛丽娜 译

人民邮电出版社

北 京

图书在版编目 (C I P) 数据

短码之美：编程达人的心得技法 / (日) 冈田佑一
著；李秀银，赛丽娜译. -- 北京：人民邮电出版社，
2011.1

(图灵程序设计丛书)

ISBN 978-7-115-24044-6

I. ①短… II. ①岡… ②李… ③赛… III. ①程序设
计 IV. ①TP311

中国版本图书馆CIP数据核字(2010)第211280号

内 容 提 要

本书主要介绍短码编程，首先介绍了短码编程的三要素：数据结构和算法、执行环境以及短码编程的精神。书中以北京大学的 PKU Judge Online 在线评分系统为编程环境，选取了该系统中比较典型的问题，如海狸啃木头、旋转拼图、葡萄酒买卖，利用数据结构和算法的知识加以分析，最后编程实现，并尝试使用各种编程语言，不断地将源代码缩短。同时还介绍并分析了一些钻安全漏洞的代码，旨在通过短码编程帮助程序员集中注意力，锻炼忍耐力，开拓视野。

本书适合所有对短码编程感兴趣的人。

图灵程序设计丛书

短码之美：编程达人的心得技法

-
- ◆ 著 [日] Ozy
审 校 やねうらお
译 李秀银 赛丽娜
责任编辑 杨海玲
执行编辑 李 静
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
中国铁道出版社印刷厂印刷
- ◆ 开本：800×1000 1/16
印张：22
字数：533千字 2011年1月第1版
印数：1-3000册 2011年1月北京第1次印刷
- 著作权合同登记号 图字：01-2010-0832号
ISBN 978-7-115-24044-6
-

定价：59.00元

读者服务热线：(010)51095186 印装质量热线：(010)67129223

反盗版热线：(010)67171154

版 权 声 明

Short Coding-Shokunin-tachi no Giho by Ozy, supervised by Urao Yane Copyright © 2007 Ozy.

All rights reserved.

Original Japanese edition published by Mainichi Communications Inc.

This Simplified Chinese edition is published by arrangement with Mainichi Communications Inc. in care of Tuttle-Mori Agency, Inc, Tokyo through Bardon-Chinese Media Agency, Taipei.

本书中文简体字版由人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

原版编者的话

当初随口向好友 Ozy 先生提及是否可以写一本书的时候，我多少有些缺乏自信，不知道什么时候能见到这本书。可能有人会说说话很不负责任。但是，计算机图书出版行业中大家经常说：“只要工夫深，铁杵磨成针。”在出版行业，我们需要跟各种各样的人打交道，并且要有各种策划方案在手。这个激烈竞争的年代迫使你联系所有能写书的人。当然，并不是所有的书都有市场，也不是所有的策划思想都能实现。但是，Ozy 先生给了我明确的承诺。

这本书远远超出了我的期望。一般而言，如果内容不是特别多，写一本书大概需要 3 个月左右的时间。如果需要再长的时间，那么作者不如转行更有利可图。而这本书却说，即使为了减少 1 字节代码也要极尽辛苦，投入大量时间去实现，由此可见，写这本书真是个费力不讨好的活。“真是太棒了，竟然能做到！”这是我现在的真实感觉。

这本书毫无疑问是一本短码编程者的圣经。其实，岂止短码编程者喜欢这本书，对于普通编程者来说，这也可以称得上是圣经级的著作。

以 Ozy 先生为代表的短码编程者自己可能也会有怀疑，短码编程对实际的代码编写工作是否真的有帮助，但我对这个问题相当肯定。

一般而言，为了使事物不断地提升，一定要有某种形式的反馈。最好是在尽量短的时间获得强烈反馈。如果是艺术作品，制作需要花费很多时间，即使完成了，由于每个人的喜好千差万别，也很难得到（正面的）反馈，这个时候只能相信自己心中的价值。在短码编程中，代码量的多少有明确的指标，而且是对是错可以根据在线评分系统瞬时做出判断。在这方面，可以在非常短的时间内得到很好的反馈。相信在提高编码水平上没有比这更好的方法了。

在以短码编程为目标的过程中，人们能渐渐写出错误较少的代码，这个观点要展开的话可能要用 50 页左右的篇幅来阐述。但是，给短码编程这本书写的序言如果不短的话，就会违反这本书本身的宗旨，我想还是就此停笔吧。

やねうらお
2007 年 7 月

致 谢

首先，感谢一直支持我的各位短码程序员。除了 5.3 节介绍过的人之外，我还得到过其他人的帮助。特别感谢 PKU Online Judge 的开发者应甫臣，感谢北京大学的各位同学。

本书的审校工作承蒙各位短码写手与 fkm 的帮助，特别是 hinoe 用心地检查了每一页，让人切实感受到“短码魂”的热血与专注，再次感谢你们！

像我这种无名的程序设计者，能以短码写手自居，完成本书的写作，要感谢やねうらお的帮助与付出，真诚地谢谢你。此外还要感谢身在天国，仍然写着源代码的 namasute，真的希望能再与你来一场程序设计大战。也要感谢爽快地允许我在书中提到 namasute 的伯父伯母。

最后，要特别感谢在我写作的这半年间，一直默默付出的爱妻。

Ozy
2007 年

目 录

第 1 章 入门

1

1.1 引言	2
1.1.1 什么是短码编程	2
1.1.2 短码编程的三要素	3
1.1.3 热情、聪明的短码编程者	4
1.1.4 短码编程者的传奇	4
1.2 战场：在线评分系统	4
1.2.1 什么是在线评分系统	4
1.2.2 世界各地的在线评分系统	4
1.2.3 PKU Judge Online	5
1.3 动手试试：短码编程的基础	6
1.3.1 编程环境	6
1.3.2 使用的编程语言和编译器	6
1.3.3 注册 POJ 账号	6
1.3.4 基本规则	7
1.3.5 微妙的规则	8
1.3.6 道德上的规则	8
1.4 热身	10
1.4.1 关于运行环境	10
1.4.2 首先从简单的加法运算开始	10
1.4.3 不简单加法运算	16
1.4.4 计算平均值	21
1.4.5 推出重叠的牌	24
1.4.6 开始模仿编程	28
1.4.7 热身结束	31

第 2 章 必读！绝妙技巧

33

2.1 精通规则	34
----------	----

2.1.1	醉汉看守	34
2.1.2	阶段	43
2.1.3	连续数的和	49
2.2	Quick Sort Hack	55
2.2.1	快速排序	55
2.2.2	qsort 函数	55
2.2.3	Hack 风暴	56
2.2.4	挑战变态排序一	58
2.2.5	挑战变态排序二	65
2.2.6	永远的 Hack	69
2.3	熟练运用库函数	69
2.3.1	方便的库函数	69
2.3.2	好用的 strtol 函数	74
2.3.3	strtol 与 itoa 天生一对	77
2.3.4	转换乐园	82
2.3.5	海狸啃木头	84
2.3.6	系绳	87
2.4	精减 1 字节的靠山	91
2.4.1	好用的 ~ 运算符	91
2.4.2	好用的 % 运算符	94
2.4.3	分组建立长期政权	99
2.4.4	ISBN 码	106
2.4.5	非常活跃的数学问题	110
2.5	按位运算的超级魔法	114
2.5.1	超级魔法的二叉树	114
2.5.2	超级魔法的密码	119
2.5.3	超级魔法的极致	123
2.6	精通指针	129
2.6.1	短码编程中的指针	129
2.6.2	旋转拼图的智力游戏	130
2.6.3	超长加法	135
2.6.4	指针和数组的密切关系	138
2.7	数学问题	142
2.7.1	不要依赖公式	142
2.7.2	越界 (数列篇)	143

2.7.3	越界（排列组合篇）	146
2.7.4	与周期有关的问题	150
2.7.5	伪随机数的周期性	151
2.7.6	三个周期	153
2.7.7	斐波那契数列	158
2.8	活用算法	163
2.8.1	高速、节省内存是编写短码的关键	163
2.8.2	三角形	163
2.9	数字的新常识	171
2.9.1	关键在“三”	171
2.9.2	整理电话号码	176
2.10	字节的热情	184
2.10.1	附上答案	184
2.10.2	汉诺塔	184

第 3 章 短码编程研究 193

3.1	更高的编程技巧	194
3.1.1	开始	194
3.1.2	关于表示法	194
3.2	精简循环	194
3.2.1	精通之后就能成为顶尖的短码编程者	194
3.2.2	简单的例子	195
3.2.3	常用结构	197
3.2.4	短码编程的基本类型	197
3.2.5	重要的短码语法	199
3.2.6	取数字游戏	200
3.2.7	葡萄酒买卖	202
3.2.8	难解：精简多层循环	204
3.3	强大的扩展语法	208
3.3.1	对短码编程者有用的扩展语法	208
3.3.2	条件运算符	208
3.3.3	数字根	209
3.3.4	扩展左值	212
3.3.5	扩展关系运算符	213

3.4 宏能不能缩短代码	213
3.4.1 基于短码编程的宏	213
3.4.2 while 语句的可能性	214
3.4.3 数组的可能性	217
3.5 神奇的 main 递归	219
3.5.1 main 递归可以最大限度地缩短代码	219
3.5.2 从单纯循环到 main 递归	219
3.5.3 破解难题的 main 递归	220
3.5.4 基于递归算法的 main 递归	232
3.5.5 传说中的 1145	233
3.5.6 小结	239
3.6 Quick Sort Hack II	240
3.6.1 征服二进制等于征服短码编程	240
3.6.2 缩短汇编代码	244
3.6.3 进一步缩短	248
3.7 作弊代码和安全性	251
3.8 挑战难题	254
3.8.1 短码编程的灵魂	254
3.8.2 组成最强阵容	254

第 4 章 语言对决

277

4.1 短码编程不限语言	278
4.1.1 各种程序设计语言	278
4.1.2 本章结构	278
4.2 Java 与 C	278
4.2.1 Java 是有缺陷的程序设计语言	278
4.2.2 使用 Java 的地方	279
4.2.3 多精度时就交给 Java	279
4.2.4 C 语言仍然强大	285
4.3 Pascal 与 C	292
4.3.1 遇到麻烦的 Pascal 语言	292
4.3.2 用 Pascal 写出比 C 语言更短的代码	293
4.3.3 复利计算	293
4.4 C++ 与 C	294

4.4.1 不经常使用的 C++	294
4.4.2 基本技巧	295
4.4.3 检查顺序	296
4.4.4 G++扩展运算符与变态 GCC 源代码的对比	299
4.5 更多的挑战	306
4.5.1 成为一名真正的短码编程者	306
4.5.2 世界各地的短码编程者的活动	306
4.5.3 挑战 Code Golf	307
4.5.4 尝试用所有语言编写短码	312

第 5 章 磨练自己 313

5.1 致亲爱的读者	314
5.1.1 向短码编程者学习	314
5.1.2 源代码的可读性	314
5.1.3 希望写出错误少的程序	315
5.1.4 更大程度地提高	316
5.1.5 有人情味、富有魅力	317
5.2 珍贵的参考资料	318
5.2.1 珍贵的书籍	318
5.2.2 网站介绍	323
5.3 精明强干的短码编程者	325
5.3.1 致谢	325
5.3.2 短码高手们(敬称略,按字母顺序)	325
5.4 最后	327

附录 331

A.1 问题一览	332
A.2 ASCII 码表	337
A.3 运算符的优先级与结合规则	338

第 1 章

入 门

```
main(n){gets(&n); putchar(n%85+5);}
```

- 1.1 引言
- 1.2 战场：在线评分系统
- 1.3 动手试试：短码编程的基础
- 1.4 热身

1.1

引言

1.1.1 什么是短码编程

```
main(n){gets(&n);putchar(n%85+5);}
```

以上是读取两个用空格分隔的整数，并输出它们的 C 语言源代码。

看到这样的代码，就可以预想到会有许多质疑声。

比如：

“输入输出是 2 位数的时候就不能运行吧！”

“是负数的时候不一定能运行吧！”

“不要使用 gets 函数！”

“没有写类型名！”

“漏掉了头文件#include<stdio.h>啦！”

“忘记了 return 语句啦！”

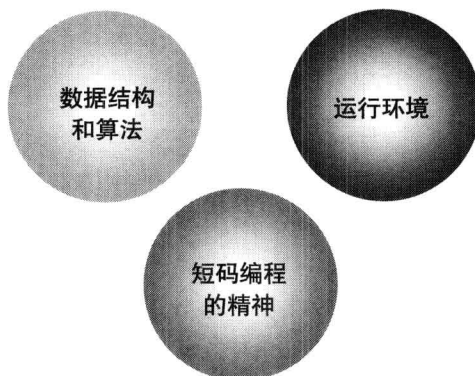
“这个真的能运行吗？写这样难懂的源代码有意义吗？”

“根本通不过编译啊。”

确实如此，但是一般情况下，在短码编程中会优先考虑积极的想法。当然，我们不可能讨论完全运行不了的源代码。这段源代码在特定的环境中是可以正确运行的（详细内容见 1.4 节）。不要单纯考虑能否运行，首先要确定“n%85+5”能否计算出结果。不要仅停留在争论代码能否被理解，而应对能产生这样的创意而感到惊奇和高兴。所以，请抛开不必要的成见，培养你丰富的好奇心。

所谓短码编程，就是从数据结构、算法、运行环境的角度去探索编写尽可能短的源代码。缩短程序源代码或者特意编写成难以理解的代码，这种做法不知道是从什么时候开始的，反正很早以前就有了。不过这样的事情，现在可能仍有人不知道。即使知道，人们的印象大概只是一小撮热衷于数字游戏的狂热者们出于兴趣所做的非营利性的事情而已。很难说以上这些知识对日常工作和生活有没有帮助。但是短码编程的行为和精神，却包含了程序员的思维方法和哲学的本质。

1.1.2 短码编程的三要素



1. 数据结构和算法

在编程中，考虑数据结构和算法是非常重要的。短码编程者为了缩短源代码而反复进行调试，最终得到最短源代码时，它不只是一段短小而难懂的源代码，还是一段运行速度最快、占用内存最少的数据结构和算法组成的源代码。为了使问题尽可能简单化，尽可能地寻找可缩短的语句，必须深入研究才能最大限度地缩短源代码，从而使执行速度和内存使用效率得到最大的发挥。在这里，最重要的事情就是要深入地分析问题，认真地解决问题。

2. 运行环境

随着抽象化程度越来越高，人们即使不清楚底层是如何实现的，也可以编写代码。但是，许多程序员也会经常遇到开发平台或编译器（以下统称“运行环境”）的问题。有些人也一定尝试过发布在自己的运行环境中编写的源代码，但却被别人说“运行不了”。自己认为是很普通的一段源代码却出现不能运行，甚至连编译都通不过的情况。短码编程就是要把目光投向因为运行环境不同而导致的差异上。为此，有时可能会写出违反内存存取规范的危险源代码。可是，为了加深对运行环境知识的了解，这是有必要的。仅仅因为高水平的程序员说过“要这样写”，所以也就模仿着写了源代码，对于初学者来说这可能是好事。但是，有时也要尝试着用怀疑的眼光去看那些既有的编码方法。当你勇敢地尝试着说“不”的时候，说不定就会发现一些平时没有注意到的事情。

3. 短码编程的精神

在短码编程中最重要的是它的精神。掌握了数据结构、算法、运行环境等丰富知识的程序员应该是能干的程序员。可是仅具备这些知识是写不出短代码的。一定还要相信无论怎样的源代码都能写得更短，并且要有不放弃、不气馁的坚强意志力。总是去想“做这样的事情，有什么意义”这种问题的人就是逃兵。一旦决定了要编写出比谁都短的源代码，就一定要坚持到底，永不放弃。这就是短码编程的精神。

1.1.3 热情、聪明的短码编程者

将源代码编写到最短的人，称为短码编程者（Short Coder）。即使只能缩短 1 字节，短码编程者也会为此而努力战斗。当战斗结束后，短码编程者们会相互间讨论缩短源代码过程中的想法，交换自己发现的黑客技巧，并会分享知识，然后继续迎接下一个新的挑战。

虽然大多数短码编程者都拥有明晰的头脑，但是通常一个人是不可能完成短码编程的，在与其他短码编程者切磋沟通的过程中，也就生成了最短源代码，同时也使自己得到了锻炼。

1.1.4 短码编程者的传奇

程序员是一边参照设计文档，一边从网上和参考书上得到所需的信息来编写程序的。然而通常设计文档本身也会有不完整的时候。就算它是完整的，可能也会在库文件的实现过程中出现问题。当然，也不能保证网上内容或者是参考书都是100%正确，大多数情况下都有可能包含少量的错误。

这时可以请教有经验的程序员去解决问题，去互联网上检索也肯定能找到相应源代码的例子。但是，不是所有的问题都可以这样解决的。有时，会碰到不能轻易判断出对错的问题，这时候应该怎么办呢？

本书通过热情、快乐、认真地描述短码编程者勇敢挑战似乎不可能完成的任务的故事，揭示出如何掌握更高的编程技巧、如何探索未知的问题，以期培养编程者的热情。

1.2

战场：在线评分系统

1.2.1 什么是在线评分系统

在线评分系统（Online Judge System）先在互联网上公布问题，程序员通过网页或者电子邮件的形式将解答问题的源代码上传到服务器，由服务器端接收后自动进行编译、执行，最后判断是否正确。世界上有许多在线评分系统，其中有详细测定程序的运行速度、内存使用量的系统，也有能利用这个系统去研究程序的运行效率和内存占用的系统。通过该系统，世界各地的程序员一起思考相同的问题，可以增进交流和信息互换。

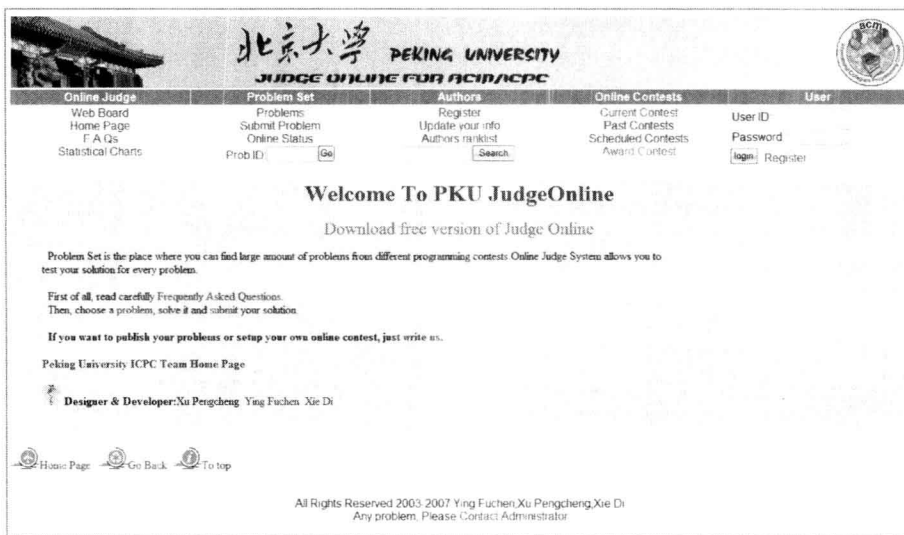
1.2.2 世界各地的在线评分系统

美国名为 TopCoder 的程序设计竞赛网站，会定期举行世界级的竞赛，它受到了谷歌等大公司的关注。自愿者不仅可以在同公司密切相关的网站里磨练编程技术，还可以与很多有能力的程

序员交流技术信息。通过这种方式，会使自己的编程水平不断地提高。除了 TopCoder 网站以外，还有许多其他在线评分系统。

顺便说一下，你知道 ACM 吗？ACM 是 Association for Computing Machinery（美国计算机协会）的缩写，是世界上最大的计算机协会。国际大学生程序设计竞赛（ACM-ICPC/ICPC）是由 ACM 主办的一项年度竞赛，旨在展示全世界大学生的创新能力和团队精神，以及在压力下编写程序、分析和解决问题的能力。该竞赛还可以促进国内大学之间甚至是国际间交流。为了在预选赛中胜出，全世界的大学生们每天都在训练。有一些大学运营的在线评分系统可以作为参赛学生们的训练场。在欧洲，有西班牙巴利亚多利德大学主办的称为 The UVa Online-Judge 的最大规模的在线评分系统。在亚洲，北京大学的 PKU Judge Online 在线评分系统是规模最大（在所有大学运营的系统）、世界上品质最高的在线评分系统。

1.2.3 PKU Judge Online



PKU Judge Online (<http://acm.pku.edu.cn/JudgeOnline/>)

PKU Judge Online (POJ)^① 是为想参加 ACM/ICPC 的学生制作的网站。在这个网站里可以编译、运行用 C、C++、Pascal (Free Pascal)、Java 语言编写的源代码。特别是关于 C/C++ 的源代码，可以选择用微软的 Visual C++ 或 GNU C/C++ 作为编译器。虽说支持丰富的语言种类是它的魅力所在，但是最吸引人的还是它的题库的数量。到 2007 年 5 月为止，POJ 公开了 2 000 多个问题。大多数问题是在过去的 ACM/ICPC 或是其他程序设计比赛中出现过的，因此即使没有参加

① 取 PKU Judge Online 的首字母是 PJO。但是在中国，一般是“大学名+OJ”。在日本，因为北京大学的知名度非常高，所以 PKU（北京大学的缩写）和 POJ 在使用上完全一样，可以混用。

过竞赛的人也能接触到这些真题。在此之上，还有 POJ 原创的问题，其独创性也使得人们对该系统始终保持热情。并且，这些问题不是简单地追加到数据库中，每次追加到数据库时都会举行小型的竞赛，让人们不时地感受竞赛的氛围并保持高度的干劲和紧张感。更值得一提的是，其提出的评价源代码的方法非常不错，是以执行速度、执行时使用的内存、源代码的代码量三个方面作为评价标准的。

1.3

动手试试：短码编程的基础

1.3.1 编程环境

POJ 系统是运行在 Windows 系统上的。如果已经导入了 GNU C/C++ 或者是 Microsoft Visual C++ 编译器，就可以直接运行了。如果没有准备好环境的话，可以下载 MinGW 等能在 Windows 环境下执行的 GNU toolchain，或者从 POJ 的主页上下载^①免费的评分系统。在免费的评分系统中含有与 POJ 使用版本相同的 gcc 与 fpc 等编译器。即使不将运行环境建立起来，也可以直接复制其中所包含的编译器来使用。

当使用 MinGW 或者 POJ 评分系统包含的编译器时，如果安装 MSYS 这个在 Unix 环境下经常使用的编译工具组的话会更方便。但是，最理想的状况是安装 MinGW、Microsoft Visual C++ 与 Cygwin 环境。Cygwin 里包含 GNU 的编译器工具组，而且比 MSYS 的更丰富。因为还可以比较^② MinGW 和 Cygwin 的 GNU C/C++，所以建议大家安装 Cygwin。

1.3.2 使用的编程语言和编译器

虽然基本上是使用 C 语言来编写源代码，但是在 POJ 上也提供了 C++、Java、Pascal 语言的编译器。另外，对于 C/C++ 来讲，也可以选择 GNU C/C++ 编译器和 Microsoft Visual C++ 编译器。要注意即使用 C 语言来编写同样的源代码，根据其使用的编译器的不同，其执行结果也会有略微的不同（稍后会详细叙述）。

1.3.3 注册 POJ 账号

为了在 POJ 上提交自己编写的源代码并编译、执行，先要在 POJ 上注册，取得账号。只要输

^① 虽然系统链接在 POJ 的主页上，但是因为版本升级等原因有时会有链接中断的情况。这时，请通过讨论区或是邮件来联系管理员。

^② MinGW 的 GNU C/C++ 作为运行时库（runtime library），使用的是 msvcrt.dll。而 Cygwin 的 GNU C/C++ 使用的是 cygwin.dll。根据使用的运行时库的不同，程序执行结果略有不同。通过比较它们之间的不同特征，可以丰富程序员的知识。