THE CLASSIC WORK
NEWLY UPDATED AND REVISED

The Art of Computer Programming

VOLUME 2
Seminumerical Algorithms
Third Edition

计算机程序设计艺术

第2卷 半数值算法

(第3版)

[美] DONALD E. KNUTH 著

(英文影印版)

Pearson Education

清华大学出版社

THE CLASSIC WORK

NEWLY UPDATED AND REVISED

The Art of Computer Programming

VOLUME 2
Seminumerical Algorithms
Third Edition

计算机程序设计艺术

第 2 卷　半数值算法

（第 3 版）

[美] DONALD E.KNUTH　著

（英文影印版）

清华大学出版社

（京）新登字 158 号

The Art of Computer Programming Volume 2：Seminumerical Algorithms (Third Edition).
Donald E.Knuth

北京市版权局著作权合同登记号：01-2001-5312

书　　名：计算机程序设计艺术　第 2 卷：　半数值算法（第 3 版）
作　　者：Donald E.Knuth
出 版 者：清华大学出版社（北京清华大学学研大厦，邮编 100084）
　　http://www.tup.tsinghua.edu.cn
印 刷 者：北京市耀华印刷有限公司
发 行 者：新华书店总店北京发行所
开　　本：787×1092　1/16　印张：48.5
版　　次：2002 年 9 月第 1 版　2002 年 9 月第 1 次印刷
印　　数：0001~ 3000
书　　号：ISBN 7-302-05815-6/TP • 3440
定　　价：一套（三册）总定价：248 元

PREFACE

O dear Ophelia!
I am ill at these numbers:
I have not art to reckon my groans.
— Hamlet (Act II, Scene 2, Line 120)

THE ALGORITHMS discussed in this book deal directly with numbers; yet I believe they are properly called *seminumerical*, because they lie on the borderline between numeric and symbolic calculation. Each algorithm not only computes the desired answers to a numerical problem, it also is intended to blend well with the internal operations of a digital computer. In many cases people are not able to appreciate the full beauty of such an algorithm unless they also have some knowledge of a computer's machine language; the efficiency of the corresponding machine program is a vital factor that cannot be divorced from the algorithm itself. The problem is to find the best ways to make computers deal with numbers, and this involves tactical as well as numerical considerations. Therefore the subject matter of this book is unmistakably a part of computer science, as well as of numerical mathematics.

Some people working in "higher levels" of numerical analysis will regard the topics treated here as the domain of system programmers. Other people working in "higher levels" of system programming will regard the topics treated here as the domain of numerical analysts. But I hope that there are a few people left who will want to look carefully at these basic methods. Although the methods reside perhaps on a low level, they underlie all of the more grandiose applications of computers to numerical problems, so it is important to know them well. We are concerned here with the interface between numerical mathematics and computer programming, and it is the mating of both types of skills that makes the subject so interesting.

There is a noticeably higher percentage of mathematical material in this book than in other volumes of this series, because of the nature of the subjects treated. In most cases the necessary mathematical topics are developed here starting almost from scratch (or from results proved in Volume 1), but in several easily recognizable sections a knowledge of calculus has been assumed.

This volume comprises Chapters 3 and 4 of the complete series. Chapter 3 is concerned with "random numbers": It is not only a study of various ways to generate random sequences, it also investigates statistical tests for randomness,

as well as the transformation of uniform random numbers into other types of random quantities; the latter subject illustrates how random numbers are used in practice. I have also included a section about the nature of randomness itself. Chapter 4 is my attempt to tell the fascinating story of what people have discovered about the processes of arithmetic, after centuries of progress. It discusses various systems for representing numbers, and how to convert between them; and it treats arithmetic on floating point numbers, high-precision integers, rational fractions, polynomials, and power series, including the questions of factoring and finding greatest common divisors.

Each of Chapters 3 and 4 can be used as the basis of a one-semester college course at the junior to graduate level. Although courses on "Random Numbers" and on "Arithmetic" are not presently a part of many college curricula, I believe the reader will find that the subject matter of these chapters lends itself nicely to a unified treatment of material that has real educational value. My own experience has been that these courses are a good means of introducing elementary probability theory and number theory to college students. Nearly all of the topics usually treated in such introductory courses arise naturally in connection with applications, and the presence of these applications can be an important motivation that helps the student to learn and to appreciate the theory. Furthermore, each chapter gives a few hints of more advanced topics that will whet the appetite of many students for further mathematical study.

For the most part this book is self-contained, except for occasional discussions relating to the MIX computer explained in Volume 1. Appendix B contains a summary of the mathematical notations used, some of which are a little different from those found in traditional mathematics books.

Preface to the Third Edition

When the second edition of this book was completed in 1980, it represented the first major test case for prototype systems of electronic publishing called TEX and METAFONT. I am now pleased to celebrate the full development of those systems by returning to the book that inspired and shaped them. At last I am able to have all volumes of *The Art of Computer Programming* in a consistent format that will make them readily adaptable to future changes in printing and display technology. The new setup has allowed me to make many thousands of improvements that I have been wanting to incorporate for a long time.

In this new edition I have gone over every word of the text, trying to retain the youthful exuberance of my original sentences while perhaps adding some more mature judgment. Dozens of new exercises have been added; dozens of old exercises have been given new and improved answers. Changes appear everywhere, but most significantly in Sections 3.5 (about theoretical guarantees of randomness), 3.6 (about portable random-number generators), 4.5.2 (about the binary gcd algorithm), and 4.7 (about composition and iteration of power series).


The Art of Computer Programming is, however, still a work in progress. Research on seminumerical algorithms continues to grow at a phenomenal rate. Therefore some parts of this book are headed by an "under construction" icon, to apologize for the fact that the material is not up-to-date. My files are bursting with important material that I plan to include in the final, glorious, fourth edition of Volume 2, perhaps 16 years from now; but I must finish Volumes 4 and 5 first, and I do not want to delay their publication any more than absolutely necessary.

I am enormously grateful to the many hundreds of people who have helped me to gather and refine this material during the past 35 years. Most of the hard work of preparing the new edition was accomplished by Silvio Levy, who expertly edited the electronic text, and by Jeffrey Oldham, who converted nearly all of the original illustrations to METAPOST format. I have corrected every error that alert readers detected in the second edition (as well as some mistakes that, alas, nobody noticed); and I have tried to avoid introducing new errors in the new material. However, I suppose some defects still remain, and I want to fix them as soon as possible. Therefore I will cheerfully pay $2.56 to the first finder of each technical, typographical, or historical error. The webpage cited on page iv contains a current listing of all corrections that have been reported to me.

Stanford, California D. E. K.
July 1997

When a book has been eight years in the making,
there are too many colleagues, typists, students,
teachers, and friends to thank.
Besides, I have no intention of giving such people
the usual exoneration from responsibility for errors which remain.
They should have corrected me!
And sometimes they are even responsible for ideas
which may turn out in the long run to be wrong.
Anyway, to such fellow explorers, my thanks.
— EDWARD F. CAMPBELL, JR. (1975)

'Defendit numerus,' [there is safety in numbers]
is the maxim of the foolish;
'Deperdit numerus,' [there is ruin in numbers]
of the wise.
— C. C. COLTON (1820)

NOTES ON THE EXERCISES

THE EXERCISES in this set of books have been designed for self-study as well as classroom study. It is difficult, if not impossible, for anyone to learn a subject purely by reading about it, without applying the information to specific problems and thereby being encouraged to think about what has been read. Furthermore, we all learn best the things that we have discovered for ourselves. Therefore the exercises form a major part of this work; a definite attempt has been made to keep them as informative as possible and to select problems that are enjoyable as well as instructive.

In many books, easy exercises are found mixed randomly among extremely difficult ones. This is sometimes unfortunate because readers like to know in advance how long a problem ought to take — otherwise they may just skip over all the problems. A classic example of such a situation is the book *Dynamic Programming* by Richard Bellman; this is an important, pioneering work in which a group of problems is collected together at the end of some chapters under the heading "Exercises and Research Problems," with extremely trivial questions appearing in the midst of deep, unsolved problems. It is rumored that someone once asked Dr. Bellman how to tell the exercises apart from the research problems, and he replied, "If you can solve it, it is an exercise; otherwise it's a research problem."

Good arguments can be made for including both research problems and very easy exercises in a book of this kind; therefore, to save the reader from the possible dilemma of determining which are which, *rating numbers* have been provided to indicate the level of difficulty. These numbers have the following general significance:

Rating *Interpretation*

00 An extremely easy exercise that can be answered immediately if the material of the text has been understood; such an exercise can almost always be worked "in your head."

10 A simple problem that makes you think over the material just read, but is by no means difficult. You should be able to do this in one minute at most; pencil and paper may be useful in obtaining the solution.

20 An average problem that tests basic understanding of the text material, but you may need about fifteen or twenty minutes to answer it completely.

30 A problem of moderate difficulty and/or complexity; this one may involve more than two hours' work to solve satisfactorily, or even more if the TV is on.

40 Quite a difficult or lengthy problem that would be suitable for a term project in classroom situations. A student should be able to solve the problem in a reasonable amount of time, but the solution is not trivial.

50 A research problem that has not yet been solved satisfactorily, as far as the author knew at the time of writing, although many people have tried. If you have found an answer to such a problem, you ought to write it up for publication; furthermore, the author of this book would appreciate hearing about the solution as soon as possible (provided that it is correct).

By interpolation in this "logarithmic" scale, the significance of other rating numbers becomes clear. For example, a rating of *17* would indicate an exercise that is a bit simpler than average. Problems with a rating of *50* that are subsequently solved by some reader may appear with a *45* rating in later editions of the book, and in the errata posted on the Internet (see page iv).

The remainder of the rating number divided by 5 indicates the amount of detailed work required. Thus, an exercise rated *24* may take longer to solve than an exercise that is rated *25*, but the latter will require more creativity.

The author has tried earnestly to assign accurate rating numbers, but it is difficult for the person who makes up a problem to know just how formidable it will be for someone else to find a solution; and everyone has more aptitude for certain types of problems than for others. It is hoped that the rating numbers represent a good guess at the level of difficulty, but they should be taken as general guidelines, not as absolute indicators.

This book has been written for readers with varying degrees of mathematical training and sophistication; as a result, some of the exercises are intended only for the use of more mathematically inclined readers. The rating is preceded by an *M* if the exercise involves mathematical concepts or motivation to a greater extent than necessary for someone who is primarily interested only in programming the algorithms themselves. An exercise is marked with the letters "*HM*" if its solution necessarily involves a knowledge of calculus or other higher mathematics not developed in this book. An "*HM*" designation does *not* necessarily imply difficulty.

Some exercises are preceded by an arrowhead, "▸"; this designates problems that are especially instructive and especially recommended. Of course, no reader/student is expected to work *all* of the exercises, so those that seem to be the most valuable have been singled out. (This is not meant to detract from the other exercises!) Each reader should at least make an attempt to solve all of the problems whose rating is *10* or less; and the arrows may help to indicate which of the problems with a higher rating should be given priority.

Solutions to most of the exercises appear in the answer section. Please use them wisely; do not turn to the answer until you have made a genuine effort to

solve the problem by yourself, or unless you absolutely do not have time to work this particular problem. *After* getting your own solution or giving the problem a decent try, you may find the answer instructive and helpful. The solution given will often be quite short, and it will sketch the details under the assumption that you have earnestly tried to solve it by your own means first. Sometimes the solution gives less information than was asked; often it gives more. It is quite possible that you may have a better answer than the one published here, or you may have found an error in the published solution; in such a case, the author will be pleased to know the details. Later editions of this book will give the improved solutions together with the solver's name where appropriate.

When working an exercise you may generally use the answers to previous exercises, unless specifically forbidden from doing so. The rating numbers have been assigned with this in mind; thus it is possible for exercise $n + 1$ to have a lower rating than exercise n, even though it includes the result of exercise n as a special case.

Summary of codes:			
		00	Immediate
		10	Simple (one minute)
		20	Medium (quarter hour)
▸	Recommended	*30*	Moderately hard
M	Mathematically oriented	*40*	Term project
HM	Requiring "higher math"	*50*	Research problem

EXERCISES

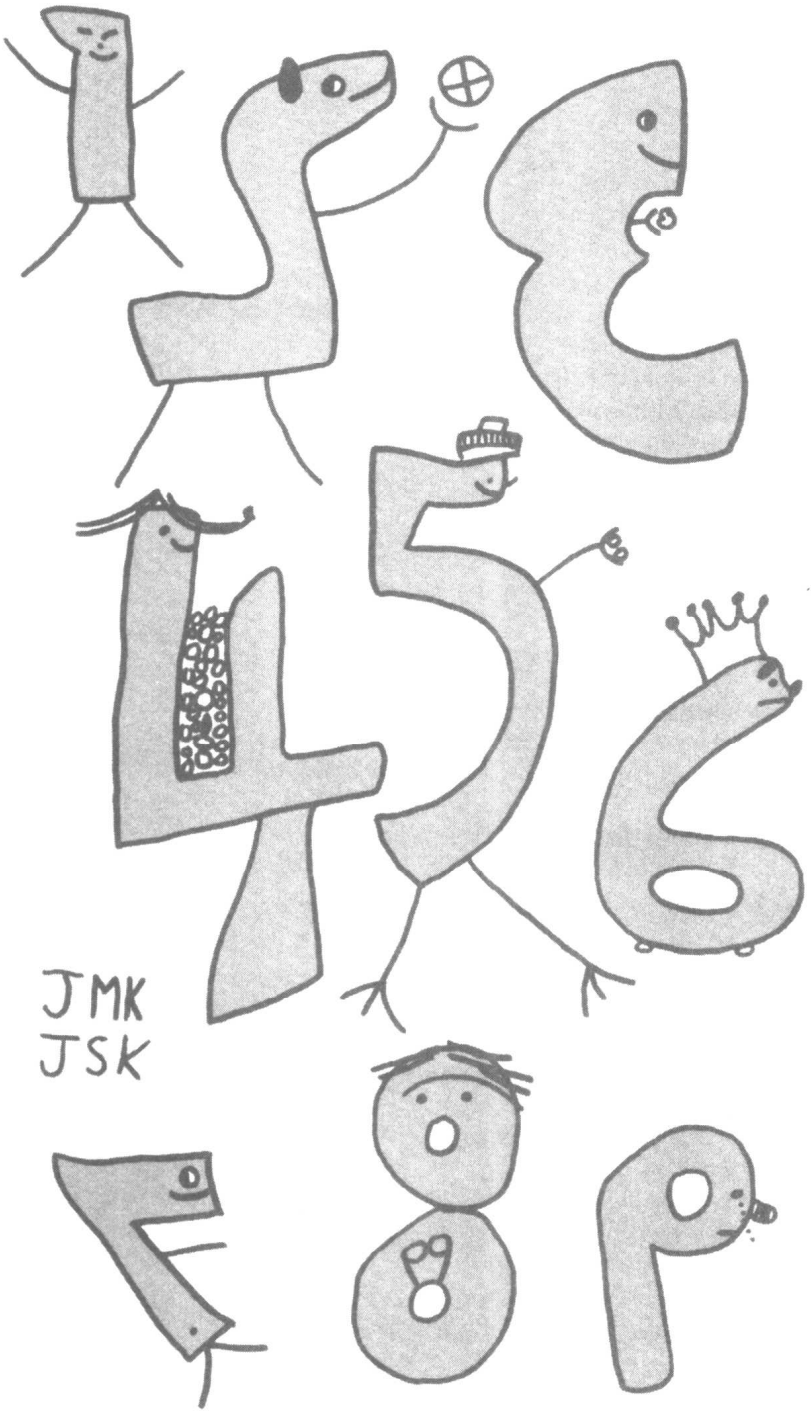
▸ **1.** [*00*] What does the rating "*M20*" mean?

2. [*10*] Of what value can the exercises in a textbook be to the reader?

3. [*34*] Leonhard Euler conjectured in 1772 that the equation $w^4 + x^4 + y^4 = z^4$ has no solution in positive integers, but Noam Elkies proved in 1987 that infinitely many solutions exist [see *Math. Comp.* **51** (1988), 825–835]. Find all integer solutions such that $0 \le w \le x \le y < z < 10^6$.

4. [*M50*] Prove that when n is an integer, $n > 4$, the equation $w^n + x^n + y^n = z^n$ has no solution in positive integers w, x, y, z.

Exercise is the beste instrument in learnyng.
— ROBERT RECORDE, *The Whetstone of Witte* (1557)


JMK
JSK

本套书由3卷组成

第1卷 基本算法

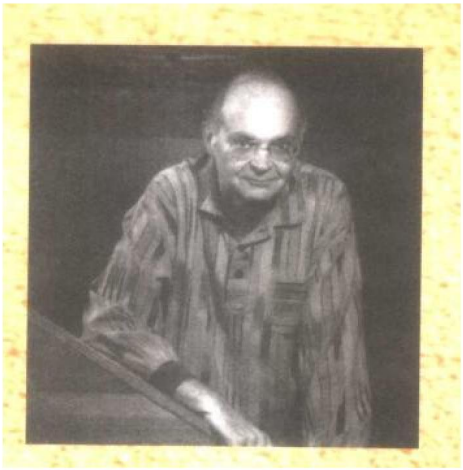
第1卷首先介绍编程的基本概念和技术，然后详细讲解信息结构方面的内容，包括信息在计算机内部的表示方法、数据元素之间的结构关系，以及有效的信息处理方法。此外，书中还描述了编程在模拟、数值方法、符号计算、软件与系统设计等方面的初级应用。新版本增加了数十项简单但重要的算法和技术，并根据当前研究发展趋势在数学预备知识方面做了大量修改。

第2卷 半数值算法

第2卷对半数值算法领域做了全面介绍，分"随机数"和"算术"两章。本卷总结了主要算法范例及这些算法的基本理论，广泛剖析了计算机程序设计与数值分析间的相互联系。第3版中最引人注目的是，Knuth对随机数生成器进行了重新处理，对形式幂级数计算作了深入讨论。

第3卷 排序和查找

这是对第3卷的头一次修订，不仅是对经典计算机排序和查找技术的最全面介绍，而且还对第1卷中的数据结构处理技术作了进一步的扩充，通盘考虑了大小型数据库和内外存储器。它遴选了一些经过反复检验的计算机方法，并对其效率做了定量分析。第3卷的突出特点是对"最优排序"一节作了修订，对排列论原理与通用散列法作了全新讨论。



Donald.E.Knuth(唐纳德.E.克努特,中文名高德纳)是算法和程序设计技术的先驱者,是计算机排版系统 T_EX 和 METAFONT 的发明者,他因这些成就和大量创造性的影响深远的著作(19 部书和 160 篇论文)而誉满全球。作为斯坦福大学计算机程序设计艺术的荣誉退休教授,他当前正全神贯注于完成其关于计算机科学的史诗性的七卷集。这一伟大工程在 1962 年他还是加利福尼亚理工学院的研究生时就开始了。Knuth 教授获得了许多奖项和荣誉,包括美国计算机协会图灵奖(ACM Turing Award),美国前总统卡特授予的科学金奖(Medal of Science),美国数学学会斯蒂尔奖(AMS Steele Prize),以及 1996 年 11 月由于发明先进技术而荣获的备受推崇的京都奖(Kyoto Prize)。Knuth 教授现与其妻 Jill 生活于斯坦福校园内。

访问 Addison-Wesley 网站可以获得有关杰出的科学家和作者 Knuth 教授的更多信息:

www.aw.com/cseng/authors/knuth

访问 Knuth 教授的个人主页,可以获得有关本书及本系列其他未出版图书的更多信息:

www-cs-faculty.stanford.edu/~knuth

THE ART OF COMPUTER PROGRAMMING

THIRD EDITION

DONALD E. KNUTH *Stanford University*

ADDISON–WESLEY

CONTENTS

Chapter 3 — Random Numbers 1

3.1. Introduction 1
3.2. Generating Uniform Random Numbers 10
3.2.1. The Linear Congruential Method 10
3.2.1.1. Choice of modulus 12
3.2.1.2. Choice of multiplier 16
3.2.1.3. Potency 23
3.2.2. Other Methods 26
3.3. Statistical Tests 41
3.3.1. General Test Procedures for Studying Random Data 41
3.3.2. Empirical Tests 61
*3.3.3. Theoretical Tests 80
3.3.4. The Spectral Test 93
3.4. Other Types of Random Quantities 119
3.4.1. Numerical Distributions 119
3.4.2. Random Sampling and Shuffling 142
*3.5. What Is a Random Sequence? 149
3.6. Summary 184

Chapter 4 — Arithmetic 194

4.1. Positional Number Systems 195
4.2. Floating Point Arithmetic 214
4.2.1. Single-Precision Calculations 214
4.2.2. Accuracy of Floating Point Arithmetic 229
*4.2.3. Double-Precision Calculations 246
4.2.4. Distribution of Floating Point Numbers 253
4.3. Multiple Precision Arithmetic 265
4.3.1. The Classical Algorithms 265
*4.3.2. Modular Arithmetic 284
*4.3.3. How Fast Can We Multiply? 294
4.4. Radix Conversion 319
4.5. Rational Arithmetic 330
4.5.1. Fractions 330
4.5.2. The Greatest Common Divisor 333
*4.5.3. Analysis of Euclid's Algorithm 356
4.5.4. Factoring into Primes 379

4.6. Polynomial Arithmetic 418
4.6.1. Division of Polynomials 420
*4.6.2. Factorization of Polynomials 439
4.6.3. Evaluation of Powers 461
4.6.4. Evaluation of Polynomials 485
*4.7. Manipulation of Power Series 525

Answers to Exercises 538

Appendix A — Tables of Numerical Quantities 726

1. Fundamental Constants (decimal) 726
2. Fundamental Constants (octal) 727
3. Harmonic Numbers, Bernoulli Numbers, Fibonacci Numbers . . . 728

Appendix B — Index to Notations 730

Index and Glossary 735

CHAPTER THREE

RANDOM NUMBERS

Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin.

— JOHN VON NEUMANN (1951)

Lest men suspect your tale untrue, Keep probability in view.

— JOHN GAY (1727)

There wanted not some beams of light to guide men in the exercise of their Stocastick faculty.

— JOHN OWEN (1662)

3.1. INTRODUCTION

NUMBERS that are "chosen at random" are useful in many different kinds of applications. For example:

a) *Simulation.* When a computer is being used to simulate natural phenomena, random numbers are required to make things realistic. Simulation covers many fields, from the study of nuclear physics (where particles are subject to random collisions) to operations research (where people come into, say, an airport at random intervals).

b) *Sampling.* It is often impractical to examine all possible cases, but a random sample will provide insight into what constitutes "typical" behavior.

c) *Numerical analysis.* Ingenious techniques for solving complicated numerical problems have been devised using random numbers. Several books have been written on this subject.

d) *Computer programming.* Random values make a good source of data for testing the effectiveness of computer algorithms. More importantly, they are crucial to the operation of *randomized algorithms*, which are often far superior to their deterministic counterparts. This use of random numbers is the primary application of interest to us in this series of books; it accounts for the fact that