HZ BOOKS

PEARSON
Addison Wesley

计算机程序设计艺术

第2卷 半数值算法

The Art of Computer Programming

Volume 2: Seminumerical Algorithms, Third Edition

(英文版·第3版)

(美) Donald E. Knuth 著
斯坦福大学

机械工业出版社
China Machine Press

计算机程序设计艺术

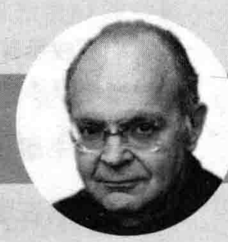
第2卷 半数值算法

The Art of Computer Programming

Volume 2: Seminumerical Algorithms, Third Edition

（英文版·第3版）

（美）Donald E. Knuth 著
斯坦福大学



机械工业出版社
China Machine Press

本书版权登记号：图字：01-2007-2437

图书在版编目（CIP）数据

计算机程序设计艺术　第2卷：半数值算法（英文版·第3版）/（美）克努特（Knuth, D. E.）著. —北京：机械工业出版社，2008.1

书名原文：The Art of Computer Programming, Volume 2: Seminumerical Algorithms, Third Edition

ISBN 978-7-111-22718-2

Ⅰ. 计…　Ⅱ. 克…　Ⅲ. 程序设计－英文　Ⅳ. TP311.1

中国版本图书馆CIP数据核字（2007）第171857号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：迟振春

北京瑞德印刷有限公司印刷·新华书店北京发行所发行

2008年1月第1版第1次印刷

170mm × 242mm · 48.75印张

定价：109.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近260个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”。为了保证这两套丛书的权威性，同时也为了更好地为学校和老师们服务，华章

公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这两套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzjsj@hzbook.com
联系电话：(010) 68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周克定	周傲英	孟小峰	岳丽华	范　明
郑国梁	施伯乐	钟玉琢	唐世渭	袁崇义
高传善	梅　宏	程　旭	程时端	谢希仁
裘宗燕	戴　葵			

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)

Character code:

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
␣	A	B	C	D	E	F	G	H	I	Δ	J	K	L	M	N	O	P	Q	R	Σ	Π	S	T	U

00 *1* No operation NOP(0)	**01** *2* $\mathrm{rA} \leftarrow \mathrm{rA} + \mathrm{V}$ ADD(0:5) FADD(6)	**02** *2* $\mathrm{rA} \leftarrow \mathrm{rA} - \mathrm{V}$ SUB(0:5) FSUB(6)	**03** *10* $\mathrm{rAX} \leftarrow \mathrm{rA} \times \mathrm{V}$ MUL(0:5) FMUL(6)
08 *2* $\mathrm{rA} \leftarrow \mathrm{V}$ LDA(0:5)	**09** *2* $\mathrm{rI1} \leftarrow \mathrm{V}$ LD1(0:5)	**10** *2* $\mathrm{rI2} \leftarrow \mathrm{V}$ LD2(0:5)	**11** *2* $\mathrm{rI3} \leftarrow \mathrm{V}$ LD3(0:5)
16 *2* $\mathrm{rA} \leftarrow -\mathrm{V}$ LDAN(0:5)	**17** *2* $\mathrm{rI1} \leftarrow -\mathrm{V}$ LD1N(0:5)	**18** *2* $\mathrm{rI2} \leftarrow -\mathrm{V}$ LD2N(0:5)	**19** *2* $\mathrm{rI3} \leftarrow -\mathrm{V}$ LD3N(0:5)
24 *2* $\mathrm{M(F)} \leftarrow \mathrm{rA}$ STA(0:5)	**25** *2* $\mathrm{M(F)} \leftarrow \mathrm{rI1}$ ST1(0:5)	**26** *2* $\mathrm{M(F)} \leftarrow \mathrm{rI2}$ ST2(0:5)	**27** *2* $\mathrm{M(F)} \leftarrow \mathrm{rI3}$ ST3(0:5)
32 *2* $\mathrm{M(F)} \leftarrow \mathrm{rJ}$ STJ(0:2)	**33** *2* $\mathrm{M(F)} \leftarrow 0$ STZ(0:5)	**34** *1* Unit F busy? JBUS(0)	**35** *1 + T* Control, unit F IOC(0)
40 *1* rA : 0, jump JA[+]	**41** *1* rI1 : 0, jump J1[+]	**42** *1* rI2 : 0, jump J2[+]	**43** *1* rI3 : 0, jump J3[+]
48 *1* $\mathrm{rA} \leftarrow [\mathrm{rA}]? \pm \mathrm{M}$ INCA(0) DECA(1) ENTA(2) ENNA(3)	**49** *1* $\mathrm{rI1} \leftarrow [\mathrm{rI1}]? \pm \mathrm{M}$ INC1(0) DEC1(1) ENT1(2) ENN1(3)	**50** *1* $\mathrm{rI2} \leftarrow [\mathrm{rI2}]? \pm \mathrm{M}$ INC2(0) DEC2(1) ENT2(2) ENN2(3)	**51** *1* $\mathrm{rI3} \leftarrow [\mathrm{rI3}]? \pm \mathrm{M}$ INC3(0) DEC3(1) ENT3(2) ENN3(3)
56 *2* $\mathrm{CI} \leftarrow \mathrm{rA(F)} : \mathrm{V}$ CMPA(0:5) FCMP(6)	**57** *2* $\mathrm{CI} \leftarrow \mathrm{rI1(F)} : \mathrm{V}$ CMP1(0:5)	**58** *2* $\mathrm{CI} \leftarrow \mathrm{rI2(F)} : \mathrm{V}$ CMP2(0:5)	**59** *2* $\mathrm{CI} \leftarrow \mathrm{rI3(F)} : \mathrm{V}$ CMP3(0:5)

General form:

C	*t*
Description OP(F)	

C = operation code, (5 : 5) field of instruction
F = op variant, (4 : 4) field of instruction
M = address of instruction after indexing
V = M(F) = contents of F field of location M
OP = symbolic name for operation
(F) = normal F setting
t = execution time; *T* = interlock time

25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55
V	W	X	Y	Z	0	1	2	3	4	5	6	7	8	9	.	,	(	)	+	-	*	/	=	$	<	>	@	;	:	'

04 *12*
rA ← rAX/V
rX ← remainder
`DIV(0:5)`
`FDIV(6)`
	05 *10*
	Special
	`NUM(0)`
	`CHAR(1)`
	`HLT(2)`
		06 *2*
		Shift M bytes
		`SLA(0)` `SRA(1)`
		`SLAX(2)` `SRAX(3)`
		`SLC(4)` `SRC(5)`
			07 *1 + 2F*
			Move F words from M to rI1
			`MOVE(1)`

12 *2* rI4 ← V `LD4(0:5)`	**13** *2* rI5 ← V `LD5(0:5)`	**14** *2* rI6 ← V `LD6(0:5)`	**15** *2* rX ← V `LDX(0:5)`
20 *2* rI4 ← −V `LD4N(0:5)`	**21** *2* rI5 ← −V `LD5N(0:5)`	**22** *2* rI6 ← −V `LD6N(0:5)`	**23** *2* rX ← −V `LDXN(0:5)`
28 *2* M(F) ← rI4 `ST4(0:5)`	**29** *2* M(F) ← rI5 `ST5(0:5)`	**30** *2* M(F) ← rI6 `ST6(0:5)`	**31** *2* M(F) ← rX `STX(0:5)`

36 *1 + T*
Input, unit F
`IN(0)`
	37 *1 + T*
	Output, unit F
	`OUT(0)`
		38 *1*
		Unit F ready?
		`JRED(0)`
			39 *1*
			Jumps
			`JMP(0)` `JSJ(1)`
			`JOV(2)` `JNOV(3)`
			also `[*]` below

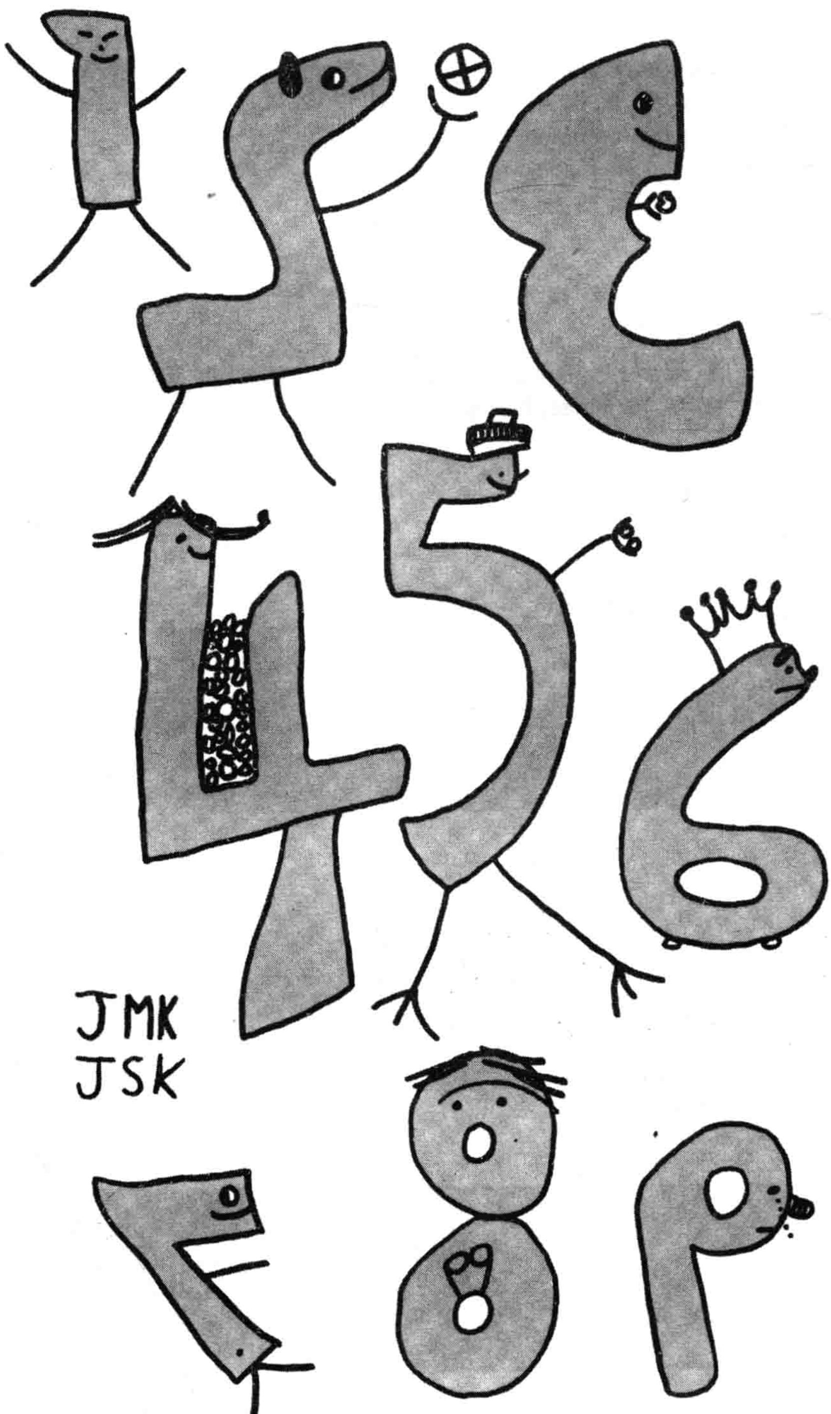
44 *1* rI4 : 0, jump `J4[+]`	**45** *1* rI5 : 0, jump `J5[+]`	**46** *1* rI6 : 0, jump `J6[+]`	**47** *1* rX : 0, jump `JX[+]`

52 *1*
rI4 ← [rI4]? ± M
`INC4(0)` `DEC4(1)`
`ENT4(2)` `ENN4(3)`
	53 *1*
	rI5 ← [rI5]? ± M
	`INC5(0)` `DEC5(1)`
	`ENT5(2)` `ENN5(3)`
		54 *1*
		rI6 ← [rI6]? ± M
		`INC6(0)` `DEC6(1)`
		`ENT6(2)` `ENN6(3)`
			55 *1*
			rX ← [rX]? ± M
			`INCX(0)` `DECX(1)`
			`ENTX(2)` `ENNX(3)`

60 *2* CI ← rI4(F) : V `CMP4(0:5)`	**61** *2* CI ← rI5(F) : V `CMP5(0:5)`	**62** *2* CI ← rI6(F) : V `CMP6(0:5)`	**63** *2* CI ← rX(F) : V `CMPX(0:5)`

rA = register A
rX = register X
rAX = registers A and X as one
rIi = index register i, $1 \le i \le 6$
rJ = register J
CI = comparison indicator

[*]:		[+]:
JL(4)	$<$	N(0)
JE(5)	$=$	Z(1)
JG(6)	$>$	P(2)
JGE(7)	$\ge$	NN(3)
JNE(8)	$\ne$	NZ(4)
JLE(9)	$\le$	NP(5)


JMK
JSK

CONTENTS