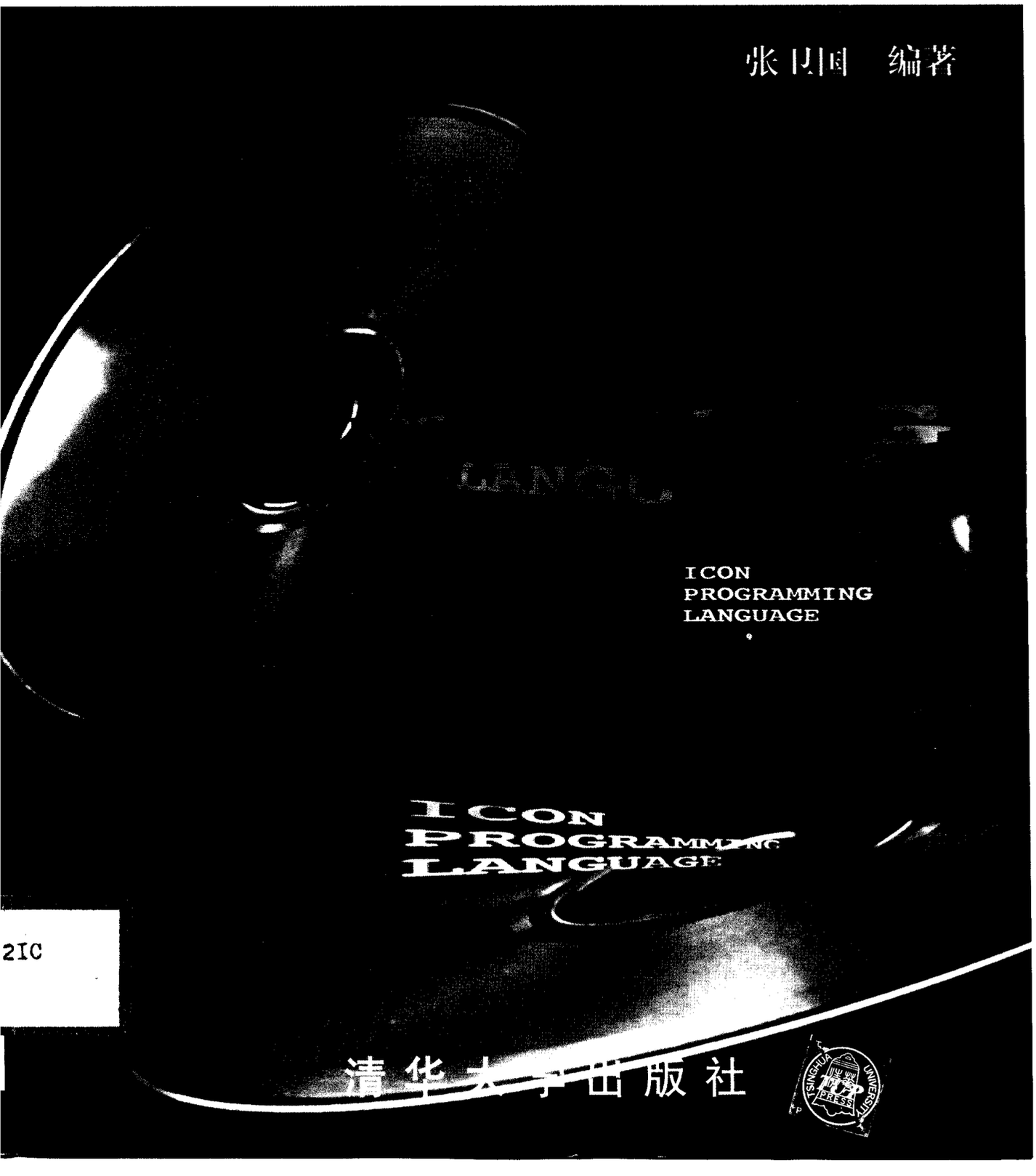
ICON语言教程

张国 编著



清华大学出版社

ICON指示灯

(京)新登字 158 号

内 容 简 介

ICON 语言是一种高级编程语言,具有强有力的字符串及结构分析和处理能力,特别适合于解决人文科学领域中的问题,所以在欧美的许多大学的文科院系里,把 ICON 语言作为必修的课程。本书第一次引进这一语言以适应我国文科院系学习计算机课程的需要。

本书全面系统地介绍了 ICON 语言,包括 ICON 语言的数据、算子、函数、过程、控制结构、程序的编写和 ICON 语言在汉语研究中的应用。书中有大量例程序,既是详细的说明和练习,又能实用,每章后均有练习,学以致用、学用结合。

ICON 语言程序可从因特网上自由下载,考虑到未上网络的读者的需要,书后附有一张软盘。

图书在版编目(CIP)数据

ICON 语言教程/张卫国编著. -北京:清华大学出版社,1998.1
ISBN 7-302-02807-9

Ⅰ.Ⅰ… Ⅱ.张… Ⅲ.程序语言,ICON 语言-教材 Ⅳ.TP 312

中国版本图书馆 CIP 数据核字(97)第 27693 号

出 版 者:清华大学出版社(北京清华大学校内,邮编:100084)
因特网地址: www. tup. tsinghua. edu.cn
责任编辑:徐培忠
印 刷 者:北京市清华园胶印厂
发 行 者:新华书店总店北京科技发行所
开 本:787×1092 1/16 印张:18 字数:372 千字
版 次:1998 年 3 月第 1 版 1998 年 3 月第 1 次印刷
书 号:ISBN 7-302-02807-9/TP·1466
印 数:0001~5000
定 价:36.00 元

序

《ICON 语言教程》是一本由文科出身的人写的介绍计算机程序设计语言及其应用的书，它的作者是中国人民大学语言文字研究所的张卫国副教授。语言文字学界的老前辈胡明扬教授希望我给这本书写一篇序，我也是文科出身，受命勉为其难。由此已经可以看出这部理科性质的书的文科特色了。

程序语言 ICON 是一种高级编程语言，它是美国亚利桑那大学计算机科学系著名的 Ralph E. Griswold 教授为首的 ICON 研制组开发研制的，在 70 年代末期就已经推出，至今已经得到广泛的应用。ICON 语言本身不断发展提高，陆续推出新的版本，目前已经升级为 ICON 9.2 版。欧美的许多大学的文科院系里，已经把它列为本科生或研究生的必修课程，介绍 ICON 语言的书已经有不少种，但是至今还没有中文译本，也没有介绍 ICON 的中文著述。ICON 语言本身在中国也极少有人了解和使用，更谈不到流行或者列入必修课。

程序语言 ICON 是一种适合于处理人文学科领域中的问题的高级编程语言，它有高水平的字符串分析、加工、处理能力以及文本的分析、加工、处理能力，虽然这并不是 ICON 独有的功能，但是 ICON 分析、加工、处理字符串的功能优于一般高级语言，特别是处理比较复杂的汉字字符串和汉字文本，ICON 会表现得更加简洁而有效。计算机从最初的以数值运算为主走向以逻辑运算为主，就注定为今天的语言文字信息处理奠定了基础。处理字符串的能力，首先和主要的也就是处理语言文字的能力，在中国，大量的和社会通用的字符串就是汉字串。人文学科的大量文献都是“汉字串”或者说“汉字文本”，利用电脑加工、处理、研究这些文献已经有了不少成功的尝试，今天，信息化、信息化社会、信息高速公路、国际互联网、“数字化生存”已经开启了一个新的局面，在整个人文科学领域，在人们的社会生活的一切方面，推广普及计算机的知识，利用计算机来处理“字符串”变得更加经常，更加重要，更加迫切。从这个角度说，ICON 语言是一种更适合于文科领域使用的语言，或者说是把人文社会科学推向电脑化、信息化的重要编程工具并不过分，所以在欧美的许多大学才会把它列入文科院系的必修课。

程序语言 ICON 是一种适合于文科人员掌握和使用的高级编程语言。现在已经有越来越多的面向各个专业领域的应用软件的开发提上议事日程，即便是计算机专业的人员开发这些领域的应用软件，也需要与该领域的专业人员结合，或者自己先学习和接受专业领域知识。随着形势发展的需要，有越来越多的不同领域的非计算机专业人员开始使用计算机编程，解决本领域的专业知识处理的电脑化问题。起初是非计算机专业的理工科的人员使用计算机进行本专业的程序设计，然后是财经、管理、图书馆、医农等等方面的专业人员使用计算机，最后是文科的哲学社会科学工作者使用计算机。由于要解决汉字在计算机上的输入输出问题，在 70 年代甚至更早已经有语言文字工作者参与计算机用字的定量、定形、定音、定序等方面的研究，而在汉字编码、汉语语音识别、汉字自动识别、汉字属性库、汉语词语属性库、汉语语料库的建设、汉语自动分词、自动标注词性、汉语理解、汉外－外汉机器翻译等方面，语言文字工作者的介入就更多。现在面向机器的语言文字研究和面向人的语言文字

研究已经受到人们的同等重视,《语文建设》、《语言文字应用》先后开辟专栏介绍“中文信息处理研究”,国家语言文字工作委员会设立中文信息司,许多大学中文系开设计算机基础课,越来越多的人,特别是年轻人希望学会并掌握一种编程语言。而用 ICON 语言编程相对来说比较简易、使用方便、处理字符串功能强,是非计算机专业特别是文科可选的理想的编程语言。

虽然,ICON 是适合文科领域开发的编程工具,也是文科人员相对容易学习使用的语言,但是,文科的人是不是一定要学会自己编程,是不是一定得自己编程? 我对这个问题的看法前后有一定变化,尽管到目前为止我还没有自己使用高级语言编程,也还没有打算自己编程。70 年代末期,在文科人员使用计算机的初期,那时计算机还只能处理西文的字符串,中文(汉字和汉语)如何进入计算机,进入计算机后又如何分析处理,都有许多的困难,常常是语言学工作者和计算机工作者结合起来解决这些难题,只是语言学工作者要懂得一点儿计算机原理和程序设计概要,计算机科学工作者也要粗通语言文字特别是汉语汉字知识,以便在研究工作中求得共同语言,互相配合。随着汉语信息处理技术的发展和计算语言学的深入,事情慢慢有了变化,一是一些计算机科学工作者从粗通语言文字专业知识到精通计算语言学和现代语言学,甚至参与或主持制定信息处理用语言文字规范;二是一些语言文字工作者特别是年轻的语言学专门人才,已不满足一般地基于语料库使用计算机来研究语言文字,更不愿意坐等计算机界的同志替自己设计一个应用软件(事实上,各行各业也不可能都等着懂软件设计的专家来做“救世主”),于是他们开始学习掌握一些高级编程语言,为在深层次开发使用语料库、为语言教学、语言研究甚至语言工程自己动手编写一些应用软件。这样,维护、修改、提高也方便。一般地说,大型语言工程项目的软件,还是要软件专业人才来主编,或者双方合作。当然,任何时候都会有少数人成为综合人才,成为精英,一个人精通两方面的知识,那毕竟是少数。但是,无论如何新一代的语言工作者和文科人才(特别是研究生)已经有必要、有可能进一步掌握适合文科的编程语言和编程技术。

事实上有一些年轻的语言学工作者已经掌握了一种或几种高级编程语言,例如 C、C++、PASCAL、FOXPRO 等等,近年来,也有一些文科毕业的研究生,通过各种渠道,拜理工科的学者为师,攻读理工科的博士学位。当然,已经掌握了其他的程序设计语言的人就不一定会再学 ICON,目前 ICON 也还没有流行。但是,此前流行的一些高级语言都是由计算机界的专家介绍进来的,因此恐怕难以顾及或者还无暇顾及人文学科的适用性。《ICON 语言教程》是一本主要写给文科人士看的书,身为个中人的作者了解文科的特点,了解文科的人士的知识结构,书中的用例、练习都是文科特别是搞语言研究或语言教学的人所熟悉的,相对于理工科人员写的同类书而言,作者的行文通俗易懂,深入浅出,因为他知道哪些必须使用的理工科的术语需要先做出界定或解释,他不忽略随文说明解释这些必备的理工科背景知识,否则,不是使人觉得像新名词轰炸,就是让人认为作者故作艰深,故意叫人读不懂。

ICON 语言是可以通过国际互联网自由下载的,如果有了这本相对通俗的书,又有可以自由下载的软件,那么,我们预期 ICON 也能在我国的人文科学领域发挥应有的作用。同时,我们也期望——这可能也是一个开端,即以文科人士为主要读者对象符合文科特点的这类理工科著述会越来越多,文科人士中能写这类书的人也越来越多。说一句绝对文科特点的话做结尾:果如是,则文科幸甚,文科人士幸甚。

张 普

1997,9,6

序　言*

早期的编程语言，如 FORTRAN，把重点放在数字运算方面。为商务运用设计的 COBOL 语言，对当时为数字计算而越来越复杂、有力的编程语言来说，有了显著的变化。尽管编程语言后来有了很大的变化，但编程语言的设计仍然以使数字运算更有效的机制为基础。语言设计者一般具有数学的背景，而用逻辑规则作为控制程序流程的基础。

计算机的结构对语言的设计也有巨大影响。为使语言在机器上运行更容易，语言的很多功能靠模仿计算机硬件的功能来实现，对程序执行速度的强调，以牺牲编程的便利和花费更多的时间为代价，反过来，更强化了这种趋势。

对于需要进行非数字运算的人来说，这样的编程语言在使用上有不少困难。试想对文本——字符构成的串——进行如下处理：

- 大多数编程语言没有提供很好的字符串处理的工具和手段，字符串只是被作为固定的或有限长度的字符数组来处理。
- 字符串处理的工具和手段不够有力；像字符串分析这样的操作，要靠复杂的、低水平的字符数组的操作来完成。
- 能够使用的操作在形式上并不符合人们所认为的或所设想的文本处理的形式。结果，造成了编程的无谓困难，本应由计算机来做的事情要由编程的人去做。

在计算机产生的初期，文本处理和数据间关系的处理只能在很低水平上才能办到。对这些数据处理的需要导致了 60 年代早期编程语言 SNOBOL 的开发。SNOBOL 里只有字符串一种数据类型，但却变得出乎意料地普及，尽管使用它的人的群体并不是很大。那时，在语言学和人文科学中，刚开始用计算机辅助进行研究，SNOBOL 正好满足人们当时的需要。

除注重字符串处理外，SNOBOL 的设计思想与同时期其他编程语言有很大不同。SNOBOL 强调语言使用上的便利和操作与所处理问题在性质上的和谐一致，如高水平的模式比配。SNOBOL 在计算机上的运行，虽然较为困难，但使得文本的处理比用其他语言容易得多。

SNOBOL 的一个关键的思想是，利用成功和失败控制程序的流程，而不是像传统做法那样使用“和”、“或”、“不”等逻辑条件语句。人类按照“成功”或“失败”的方式思考问题，在日常生活中不断地使用这种思考方式。SNOBOL 引入的一个与此有关思想就是定目标执行，在定目标执行中，一个尝试（如，模式比配）要是失败了，就尝试可替代的选择。这样的方法，同样是人类面对生活和周围世界的方法——人们并不是按照程序里的逻辑或循环表达式那样的方法与生活和世界打交道的。

SNOBOL 经历了多个版本的变化，功能越来越强，增加了像索引表这样精妙的数据结构。然而，SNOBOL 从一开始就显露出它的句法形式十分古怪，而缺少后来编程语言所具有的许多优点。而且，SNOBOL 的许多有力的工具和手段只限于文本处理。

ICON 语言，自 1980 年开发，吸取了 SNOBOL 语言中好的工具和手段并使之更简单、更通用，采用人们熟悉的句法，增添了大量处理工具和功能。成功、失败、定目标执行可以用

于任何运算，而不只限于文本处理。ICON 保留了这样的设计思想：编程容易，语言机能适合所要解决的问题而与在常规计算机硬件上运行的难易无关。

ICON 经历了一系列版本的发展，不断提高，不断增添新的特色，到了版本 9，已经成为成熟的、稳定的编程语言。

ICON 可广泛地用于不同的领域。在像语言学和人文科学这样的研究中，ICON 已在很大程度上取代了 SNOBOL4。另一方面，ICON 常被系统程序员用来进行文件分析和转换。看似不可思议的是，ICON 不仅特别适宜编写短小、只用于某一具体任务的程序，而且也十分适用于解决非常复杂的、相当困难的编程任务，例如快速原型法。随着 PC 机的普及，ICON 在从游戏到金融分析等广泛的领域里大有用武之地。

由于 ICON 具有其他编程语言所没有的思想和特色，学习使用它就要付出一定的努力。同时，要学会使用 ICON，还要乐意用与习惯不同的眼光看待编程，比如，要理解成功、失败、定目标执行的意义，而不是用逻辑条件和循环语句来代替它们。

学习 ICON 所付出的努力会得到优厚的回报。编程将更容易，得到结果将更快，用其他编程语言不容易做到的事情，用 ICON 很容易做到。用 ICON 编程，可以摆脱大多数编程语言对人们的拘束，得到无限的情趣。

越来越多的人将在工作和生意中使用计算机，我觉得，张卫国先生写的《ICON 语言教程》在把 ICON 语言介绍给中国人方面是很有价值的。我希望，ICON 会使中国用户感兴趣，并能使他们富有成果和兴趣盎然地使用计算机。

拉尔夫·格里斯华德
美国亚利桑那大学
校务委员会委员
计算机科学系教授

1997,11,17.

---

* 序言的作者格里斯华德(R. Griswold)教授是 ICON 语言的创始人及主要设计人。

Foreword

The earliest programming languages, of which FORTRAN remains, focused on numerical computation. COBOL, for business applications, was a notable departure from the trend toward more sophisticated and capable programming languages for numerical computation.

Even as programming languages became more varied, design still was based on mechanisms that work best for numerical computation. Language designers often had mathematical backgrounds, and the rules of logic were the basis for controlling program flow.

Computer architecture also strongly influenced language design. Language facilities mimicked hardware operations, making implementation easier. An emphasis on the speed of program execution reinforced this emphasis, albeit at the expense of ease of programming and the time it required.

Such programming languages present several problems for persons who need to do computation that is not numerical in nature. Consider processing text--strings of characters:

- Most programming languages do not provide good facilities for handling strings; they are treated as arrays of characters of fixed and often limited length.
- Facilities for processing strings are weak; operations such as string analysis (parsing) have to be cast in complicated, low-level manipulation of character arrays.
- The operations that are available are not in a form in which human beings think about and conceptualize text processing.

The result is unnecessary difficulty in programming. The human programmer is left to do what computers should do.

In the early days of computers, processing text and dealing with structural relationships among data was basely possible.

The need to process such data led to the development in the early 1960s of the programming language SNOBOL, whose sole data type was the string. SNOBOL was unexpectedly popular, albeit in a small programming community. At this time the use of computers to assist in re-

search in linguistic and humanistic studies was just beginning. SNOBOL fit their needs.

In addition to a focus on string processing, SNOBOL's design philosophy was very different from contemporaneous languages. SNOBOL focused on ease of use and operations natural to the problem domain, such as high-level pattern matching. The implementation was, of course difficult, but the result made text processing much easier than in other languages.

A key concept in SNOBOL was the use of success and failure to control program flow, rather than the conventional use of logical conditionals such as "and", "or", and "not". Human beings think in terms of success and failure and useit constantly in everyday life. A related idea SNOBOL introduced was goal-directed evaluation, in which if an attempted computation (such as a pattern match) fails, an alternative is tried. Again, this is the way human beings deal with life and the world around them--not in terms of logical expressions and loops.

SNOBOL evolved through a series of increasingly more powerful versions, adding sophisticated data structures such as associative tables. SNOBOL's early origins, however, marked it. It had an idiosyncratic syntax and lacked many of the amenities of later programming languages. In addition, SNOBOL's most powerful facilities were limited to processing text.

The Icon language, developed in 1980, took the best facilities of the SNOBOL languages, simplified and generalized them, adopted a familiar syntax, and added an extensive computational repertoire. Success, failure, and goal-directed evaluation could be applied to all kinds of computation, not just text processing.

Icon retained the design philosophy: ease of programming and language facilities that fit the problem domain regardless of the difficulty of implementation on conventional computer hardware.

Icon evolved through a series of versions, refining, unifying, and adding features, to Version 9, which is mature and stable.

Icon is widely used for surprisingly diverse applications. It has largely supplanted SNOBOL4 for research in areas like linguistics and the humanities. In contrast, Icon often is used by systems programmers for analyzing and transforming files. Paradoxically, Icon is best for short, "one-use", programs and for very complex and conceptually difficult programming tasks, such as rapid prototyping. As personal computers have proliferated, Icon has found a home for a variety of individual programming tasks ranging from games to financial analysis.

Since Icon has novel concepts and features not found in other programming languages, it is nec-

essary to make an investment in learning to use it. So, also, is a willingness to take a different view of programming than is customary--for example, to understand success, failure, and goal-directed evaluation instead of trying to translate them into logical conditionals and loops.

An investment in learning Icon pays off handsomely. Programming is easier, results can be obtained more quickly, and it is practical to do things that aren't practical in other programming languages. A bonus is that programming in Icon can be fun--a liberation from the straitjacket imposed by most programming languages.

I think that the book, written by Mr. Zhang, will be valuable in introducing Icon to Chinese users as more and more persons will be using computers in their office and businesses. I hope Icon will be interesting to Chinese people and allow them to use their computers in productive and interesting ways.

Ralph E. Griswold
Regent's Professor of Computer Science
The University of Arizona
Tucson, Arizona U. S. A.

1997,11,17.

前　言

本书是介绍计算机程序语言 ICON 及使用的教程。本书的读者是从事语言学、文学等人文科学方面的学习和研究的人员。ICON 语言，是美国亚利桑那(Arizona)大学计算机科学系以 Ralph E. Griswold 为首的项目研究组设计、开发的以字符串和结构处理为特色的一种高级计算机程序语言。

随着计算机科学的发展和计算机的普及，不仅计算机成为其他学科不可缺少的工具，而且计算机科学也促进了其他学科的发展。文科也不例外。计算机不但为语言学、文学等文科提供了有力的现代化工具，文科人员在学习、研究中越来越多地使用计算机，而且计算机科学的思想、方法等也为这些传统的文科注入新的活力，使这些学科有了长足的发展，展现出新的面貌，甚至产生了文理结合的新的学科，如计算语言学、语言风格的统计研究等。

计算机不只是计算工具，也不仅仅是编辑、打印文章的机器。计算机的最高价值在于对数据进行各种需要的处理。要处理就要有进行处理的程序和软件。虽然市场上有很多应用软件，但是用计算机要做的各种处理任务是无限的，而市场上现成的软件总是有限的，不可能完全满足每一个人、每一项具体处理工作的需要。所以，为更好地发挥计算机的潜力做好自己的工作，文科人员应该掌握程序语言，编制自己工作中需要的程序。

目前，多数文科人员还不能使用程序语言编程，有不少人只是拿计算机做文字处理机，做录入、编辑、打印的工作。造成这种情况的主要一个原因是，Basic、C、C＋＋ 等程序语言，一方面对文科人员来说不容易学习，一方面并不非常适合文科人员的使用。文科人员希望有一种相对来说比较容易学习、掌握又适宜自己学习、研究使用的程序语言。

ICON 是适合于文科人员学习、使用的一种计算机程序语言。一方面，ICON 可以字符串和结构处理为特色，同时又具有一般通用计算机程序语言具有的其他功能，特别适合以文字处理为主要内容的文科人员的工作性质的需要；另一方面，ICON 语言，没有非常复杂的句法，不需要使用者操心对计算机底层的控制，不需要变量类型的说明，进行动态的记忆和数据管理，等等，十分适合于没有很多数理知识的文科人员工作、研究情况和需要。ICON 不仅可以用来编制一些短小的程序解决工作中的具体问题，而且也可以设计较大的实用软件，解决一系列相关的问题。ICON 使用的实践证明，ICON 完全可以用在语言研究、文本分析和处理、语言信息研究、数据分析、过滤、转换处理、计算机辅助教学等多方面，是适合于文科的人员及研究生学习、使用的程序语言，是易于文科人员掌握的方便的编程工具。

作者是从事文科研究的人，近几年一直在学习、使用 ICON 语言，并作为一门学习和实践相结合的课程给文科研究生讲授。本书就是在作者 ICON 使用经验和授课讲义的基础上写成的。

本书分为 12 章：

第 3 章　控制结构
第 4 章　字符集和字符串
第 5 章　输出、输入
第 6 章　字符串分析和扫描
第 7 章　结构数据
第 8 章　编译选项、命令行参数和预处理指令
第 9 章　排序
第 10 章　多值式和协表达式
第 11 章　编程中的一些技术问题
第 12 章　程序的编写和调试

其中，第 1 章是 ICON 语言的概述和基础，第 2 到第 8 章是 ICON 语言有关部分的讨论，第 9、10 章是一些深入问题的讨论，第 11、12 章讨论一些 ICON 使用中有关的技术和技巧。从第 1 章到第 12 章，内容上是由浅入深，从语言本身的讨论到语言使用的讨论。与一般计算机程序语言的书不同，本书章节和内容的安排不完全以语言本身的内容为线索，而是以学习的过程和使用的需要为线索，同时考虑语言本身的内容，所以，从第 1 章到第 12 章，整个学习过程是从易到难、从解决简单问题到能解决较复杂问题的过程。使用本书时，要从第 1 章开始，按章节循序渐进。当然，在第 1 章到第 10 章的学习过程中，遇到与第 11、12 章的内容有关的问题，参阅第 11、12 章里的有关内容也是有益处的。

本书的主要参考文献有：

The Icon Programming Language
Ralph E. Griswold, Madge T. Griswold , Prentice Hall, 1990

Version 9.0 of the Icon Programming Language.
Ralph E. Griswold, Clinton L. Jeffery, and Gregg M. Townsend, IPD236, 亚利桑那大学计算机科学系研究报告。

Version 9.0 of Icon for MS－DOS
Ralph E. Griswold, IPD247, 亚利桑那大学计算机科学系研究报告。

Version 9.0 of Icon for MS－DOS 386/486 Platforms
Ralph E. Griswold, IPD248, 亚利桑那大学计算机科学系研究报告。

An Overview of the Icon Programming Language Version 9.0
Ralph E. Griswold, IPD266a, 亚利桑那大学计算机科学系研究报告。

本书能够出版，作者要感谢不少人。作者在 ICON 语言的学习、使用中，从 Ralph E. Griswold 和 Madge T. Griswold 教授的 The Icon Programming Language（《程序语言 ICON》）和有关研究报告里受益匪浅，本书里有些例子便是引自或改自该书或有关研究报告。Griswodd 教授还在百忙中抽出时间，特意为本书写了序言。所以，作者在这里向两位 Griswold 教授及他们的同事表示深深的感谢。

在本书的写作、出版过程中，胡明扬教授、黄昌宁教授、张普教授给了作者许多支持和帮助，张普教授还为本书写了序文。对这几位先生，作者在此再次表示衷心的感谢。作者还要感谢听过这门课的学生，他们提出了一些很好的意见，帮助找出了不少打印上的错误。

这本书出版后，如果能对读者有所帮助，作者将感到非常高兴。同时，由于作者水平上的局限，本书里错误和不足恐所难免。看了这本书后，如果读者有什么意见或建议，希望读者不吝赐教。

作　者

1997,10

目　录