Golden Common LISP

HOPE

使用大全

北京希望电脑公司

■北京市新闻出版局
准印证号：891198
■订购单位：北京 8721 信箱资料部
■邮　　码：100080
■电　　话：2562329
■乘　　车：320、332、302路车至海淀黄庄下车
■办公地点：希望公司大楼 101 房间

Golden Common LISP使用大全

朱隽
唐汉
编译

北京希望电脑公司

一九九一年九月

序言

本书讨论如何用 LISP 进行程序设计．LISP 是一种使用表处理和符号操作以解决人工智能（AI）问题的高级语言．由于 LISP 的这些编程机制非常灵活，所以 LISP 自其 1958 年出现以来已成为开发 AI 应用程序的主要语言．LISP 不仅是当前仍在使用的第二种最老的计算机语言（居于 FORTRAN 之后），而且其应用也随着人们对 AI 重新产生兴趣而迅速发展．

对习惯于诸如 BASIC 或 PASCAL 这样的过程式语言的程序员来说，LISP 会使他们感到难以学习，因为 LISP 的操作方式与过程性语言十分不同．作为一种应用式语言，LISP 所使用的机制是函数作用于其参数，然后返回值至更高级的函数．这样的函数调用可被嵌套至任意深度．因此，整个 LISP 程序就类似于一种层次结构．这样一种方法能够很好地克服 AI 的复杂性，因为人们在解决一现实问题所采取的认知步骤可被直接反映成程序的结构．

关于本书

本书的目的是帮助你学习 LISP，它既可作为大学课本，也可作为自学基础．

为了使本书适于自学，我们作出了特别的努力．在许多 LISP 教材的一开始都有类似于这样的句子："本书来源于课堂讲义..."．我们这本书也许可以这么说："本书来源于试图理解来源于课堂讲义的教材所遇到的困难..."．只针对计算机专业学生的教材在为其它人们用作自学课本时会有许多问题，因为它们总假定读者已掌握一定的初步知识；而这正是他们所缺少的．

为此，本书提供了几个介绍性的章节以逐步介绍 LISP 程序设计的艺术．有经验的程序员或计算机科学专业的大学生可能会发现这些内容有些简单了，他们完全可以跳过它们．一旦读者了解了这些章节中的内容并加足马力就会看到后面的内容正是建立于前面这些内容之上的．

本书的组织方式是将较为常用也较短的内容单独组织一章．这样就可在一章中专门讨论一个主要问题（如流，包，串等），而不是将许多无关的材料组织在一起．

由于人们普遍认为学习程序设计的最好方法是在一交互式的环境中，所以我们假定你手上有一 LISP 解释器可用．本书所基于的是 Golden Common LISP 的 1.1 词法域版，这是支持 Common Lisp 的一个很大子集的 MS-DOS 实现．但本书和其它 LISP 实现一起使用效果也会很好，只要这些实现的内容与 Common LISP 标准相差不是太大．

一般来说，当例子前出现与 GCLISP 提示符 * 时，就意味着你可将例子敲入实际系统中，并会满意地看到所说明的结果．与此同时，我们也希望你能自己进行尝试以增进对系统的了解．

在大多数章的末尾都有十个练习涉及了该章所讨论的概念．如果你能独立地解决每章所有的练习，就说明你已掌握了该章的基本内容．有练习的答案都在本书最后给出。

本书还涉及了通常讨论 LISP 解释器的内部性质时未讨论的一些细节，如符号数据对象在计算机内存中的表示方式，建立及维护对象表的方式，等等．对解释器内部工作机制的了解将会极大地提高你对 LISP 的理解．

本书组织

本书的前五章介绍了符号及符号操作的概念，将表用作这种符号操作的方式，LISP 函数之性质及其如何作用于其变元．各章标题及内容如下：

第一章　智能与符号处理

本章使用了一简单的日常生活中的例子来说明我们在解题时无意识地使用及处理符号的方式．对符号的各种用途都进行了探讨：作为表示现实世界对象或对象性质的名字，或作为要被执行的过程的描述．

第二章　LISP解释器

本章对LISP解释器进行初步介绍，包括可能会出现的各种错误情况，以及如何对之进行处理．

第三章　LISP 函数

作为一种应用式语言，LISP取决于函数调用才能完成一切．本章讨论了函数调用的语法，说明函数是如何作用到其变元上的，以及如何嵌套以反映一 LISP 应用程序的各层细节的．

第四章　符号数据对象

在与LISP打交道的过程中，如果你能想像你正在使用的数据结构的性质，则会很有助益的．本章主要讨论符号，说明符号被作为数据对象而建立的方式，以及一符号可能具有的各种属性．

第五章　表的建立及表的操作

链接表是将符号及其它数据对象组合起来以反应其功能及，或其它关系的基本方式．本章描述了链接表的结构，并说明了创建及处理表的方式．

上面这些预备章节介绍了 LISP 的基本核心：其应用式性质，它用来解决问题的符号与表，以及这些数据对象在计算机内存中内部表示的方式．一旦理解了这些基本概念，就能容易地掌握本书后面的 LISP 编程细节．

下面四章讨论了在 LISP 中定义过程的方法，在一词法域实现中将值约束至变量的方法，作为测试方法的许多谓词的性质，以及将这些测试结果用作 if / then / else 类判断之基础的方法．

第六章　函数定义

作为一应用式语言，LISP的性质是将一函数作用到其变元上．所以，整个LISP程序就由一组函数调用的层次结构而组成的．本章说明如何利用 LISP 的内部函数及其它用户定义函数来定义你自己的函数．

第七章　变量作用域

将一个LISP系统与另一个LISP系统区分开来的一个重要性质就是决定如何将值约束至符号上的规则系统．在本章中，我们用具体例子来说明在一词法域实现中是如何建立这种约束的．

第八章　谓词及布尔操作符

谓词被用来测试一数据对象是否满足某些持定的条件或者两个对象之间是否有着某些特定的关系．测试结果为假时谓词返回 NIL，为真时返回非 NIL 值．这些值可被用在 LISP 的条件表达式中的控制分支判断．

第九章　分支操作

LISP的if／then／else分支判断是通过诸如cond，if和when这样的各种条件结构而完成的．此外还有 ifn，unless 和 case 结构．

如同在任何高级语言中，信息在系统中输入输出的方式既基本又重要．下面二章就探讨 LISP 中用于这些目的的机制，包括以特殊方式来格式化输出的性质．

第十章　输入函数

LISP系统中进行输入的主要机制是read，它从键盘或其它输入源中一次读取一个 LISP 式子．其它主要的变形包括 read-char（从一输入流中读取单个字符）；read-line（将整个一行输入作为一字符串而读取）；以及 read-from-string（从一串中读取字符）．

第十一章　输出函数

LISP系统中进行输出的主要机制是print，它将一LISP数据对象的打印表示输出至终端或打印机这样的输出设备上．和流一起使用时，数据还能被传至诸如文件这样所选择的输出场所．

第十二章　格式打印输出

在LISP中有各种方法可用来将正文串及数据一起进行格式化输出：基本print函数的变体以及 backquote．另外，专门的 format 函数可用来定义将数据输出至屏幕上的方式．

许多计算机操作都要求多次执行某些活动或直至某些测试被满足．LISP 提供了各种函数以执行这样的重复．此外，还提供了其它可执行功能很强的递归操作的方法．专门的映射函数则可将一函数连续地作用到一表的各元素上．

第十三章　迭代程序设计

基本的迭代函数do提供了执行重复的一种灵活方法，包括初始化及在每一遍中对变量进行修改．其它象 dolist 和 dotimes 这样的变形只允许对表和整数序列有效地执行简单迭代操作．

第十四章　递归函数

LISP的一个更为有力的性质是其递旧性质，在这里一函数可在其自己的函数定义中对其本身进行调用．该性质允许将一问题分解成其自身的较小部分，并常常提供了解决一问题的比迭代方法更有效的方法．

第十五章　映射函数

如mapcar和maplist这样的映射函数提供了一种连续地将一函数作用到一表中的各元素并将结果累积起来的方法．其它象 mapcan 和 mapcon 这样的变形能够对表中各元素进行一次测试，并从累积结果中过滤掉不满足测试的那些元素．

下面几章探讨 LISP 的更为高级的性质．我们讨论了由不立即计算其变元而带来的灵活性的宏的使用．我们还讨论块的建立以及以一词法(return)或动态(catch 和 throw)方式从块中退出的方法．另外还描述了产生及处理多元值的方法，和作为存储及检索数据机制的特性表与联结表的性质．

第十六章　宏的建立与使用

宏提供了建立LISP中一般化过程的一种模式．宏的灵活性在于宏并不计算其变元，而是在计算所扩展的表达式之前先执行一遍扩展．

第十七章　块、退出及多元值

block机制允许从该块词法域中的任何地方返回；catch和throw则提供了更大的灵活性：它们允许以动态的方式在程序中的任何地方退出．LISP 提供了产生由一函数返回的多元值的方法，并有各种专门函数来处理这些值．

第十八章　特性表及联结表

LISP的符号常被用来表示现实世界中的对象．特性表可将一特性名表与其相应的值联系起来．另一种更为一般的查询机制则由联结表所提供：在这里，一关键字与一值相联．

在 LISP 中，还有各种其它的数据结构可用在不同的目的上．流提供了解释器与输入

/输出设备间的必要接口．字符被用在其 ASCII 性质及各种宏操作中．串则在屏幕显示及自然语言处理中得到广泛的应用．各种整数及浮点数也为 Common LISP 所支持．数组和结构提供了将表用作数据存储及检索的新方法．在下面四章中，我们将探讨对这些数据结构的使用．

第十九章　流

流这一数据对象提供了与外设及文件的接口．输入流一般从键盘、文件或其它地方读取数据；输出流则将数据打印至屏幕、打印机或其它外设上．各种与流有关的函数在自然语言处理中对串进行操作时特别有用．

第二十章　字符内幕

除了支持标准的ASCII字符集，GCLISP还支持对与Ctrl和Alt键相联的:control和:meta 位的使用．另外，还支持了 Common LISP 宏字符的一个很大子集，且可为任何字符定义宏函数．

第二十一章　串

本章说明了用来对串进行处理的各种函数，包括将其它LISP数据对象转换成串以及将一串转换成流以作为从中可读取字符的一输入源．其它函数还允许抽取子串，串比较及串合并等．

第二十二章　数、数组及结构

本章提供了各种函数以对整数及浮点数进行指数、对数、三角函数及位操作其它函数还提供了对数组和向量的创建，以及从一数组中检索数据．最后，defstruct 和有关的函数可用来建立用户定义的数据结构．

GCLISP 还提供了各种其它 Common Lisp 性质，包括与文件系统进行接口的 pathname 函数．还有各种工具可用来进行程序的错误处理、中断、调试及跟踪．包提供了将一用户与其它用户隔离的一种方法，这样就能防止意外的符号名冲突．GCLISP 的其它与实现有关的性质还包括在机器语言层次上与系统打交道的函数．这些内容在下面四章中讨论．

第二十二章　与文件系统的接口

Common LISP中的路径名提供了用来创建及存取文件的一标准化方法．如 with-open-file 这样的函数可使文件在使用后被自动关闭；另外的函数还提供了对文件的重新命名及删除．还有各种用于测试及/或返回一路径名之组成的函数．

第二十五章　包

在多用户系统中可能会由于不同用户为不同目的却使用了相同的符号名而出现的冲突．包的建立则为每个用户都提供了该用户自己所专用的符号查找表．在他们各自的

LISP 和 KEYWORD 包中维护着基本的 LISP 基元和关键字.

第二十六章　GCLISP的其它性质

GCLISP还提供了各种实用函数以分配内存及控制垃圾回收活动．在机器语言层次上还可用低级函数与系统打交道，并可在特定的 MS-DOS 基／偏移地址处对数据进行存取．最后，还能产生 CPU 中断以用作诸如测定时间这样的目的.

最后四章提供了有关 GMACS 编辑器及 LISP 一般性质(包括解释器的工作结构，read／eval／print 循环的内部工作细节，以及动态域和词法域解释器之间的本质差别)的进一步信息.

第二十七章　GMACS 编辑器

为了开发程序，使用GMACS编辑器是必不可少的，因为在其中程序被作出修改之后可被立即进行测试．本章概述了此编辑器的所有主要性质，包括光标移动，正文块移动，向前／向后搜索，使用子窗口以用与编辑等.

第二十八章　LISP解释器内幕

为了很好地理解LISP的机制，熟悉解释器的一般工作方式是很有用的，包括解释器的结构，为不同目的而对内存进行分配的方式，以及对象表是如何被创建及维护的.

第二十九章　read／eval／print循环

基本read／eval／print机制在这里被详细详论，包括read是怎样相应LISP的输人而建立一内部数据对象的，eval 是怎样执行计算过程的，以及 print 是怎样有效地建立 eval 所返回的 LISP 数据对象的打印表示的.

第三十章　词法域及动态域

LISP的一个较为费解之处是动态域解释器与词法域解释器之间的不同之处．我们为各个类型建立了两个很小的解释器以比较处理约束环境的方式．闭包的使用也在这里加以说明.

附录 A 含 GCLISP 支持的所有 Common LISP 函数的索引．除说明这些函数在本书中出现的章节，还给出了交叉引用信息以便于在 Guy Steele 的＜＜Common LISP: The Language＞＞（简称 CLRM）一书中查找，＜＜Golden Common LISP 参考手册（1.0 及 1.1 版）＞＞则被简称为 GCLRM.

附录 B 含在本书（包括正文及练习）中所开发的所有用户定义函数的索引.

最后一部分为练习答案.

目 录