AutoLISP 语言程序设计

郭秀娟　于全通　范小鸥　主　编

化学工业出版社

·北 京·

本书主要介绍了 AutoCAD 内部的编程语言 AutoLISP，系统而详细地介绍了 AutoLISP 的各类函数和语法规则。从 AutoLISP 的数据类型和程序结构入手，循序渐进地介绍了 AutoLISP 的基本函数、AutoLISP 绘图功能、AutoLISP 建筑设计制图程序范例等内容。

本书列举了丰富的程序实例，图文并茂，清晰易懂。

本书可作为高等院校建筑学及相关专业的教材，也可供其他从事设计绘图工作、学习 AutoLISP 语言的有关人员参考，还可作为 AutoLISP 培训学习的教材。

图书在版编目(CIP)数据

AutoLISP 语言程序设计 / 郭秀娟，于全通，范小鸥主编.
北京：化学工业出版社，2008.6
ISBN 978-7-122-03236-2

Ⅰ. A… Ⅱ. ①郭…②于…③范… Ⅲ. 计算机辅助设计-应用软件，AutoLISP-程序设计 Ⅳ. TP391.72

中国版本图书馆 CIP 数据核字（2008）第 099737 号

责任编辑：李彦玲 鲍晓娟　　装帧设计：韩 飞
责任校对：王素芹

出版发行：化学工业出版社（北京市东城区青年湖南街 13 号 邮政编码 100011）
印　　刷：大厂聚鑫印刷有限责任公司
装　　订：三河市宇新装订厂
787mm×1092mm 1/16 印张 14½ 字数 359 千字　　2008 年 8 月北京第 1 版第 1 次印刷

购书咨询：010-64518888（传真：010-64519686） 售后服务：010-64518899
网　　址：http: // www. cip. com. cn
凡购买本书，如有缺损质量问题，本社销售中心负责调换。

定　　价：29.00 元

第 1 章 AutoLISP 语言概述

1.1 AutoLISP 语言简介

1.1.1 开发 AutoCAD 的重要工具

LISP（List Processing Language）是一种计算机的表处理语言，是在人工智能学科领域广泛应用的一种程序设计语言。AutoLISP 语言是嵌于 AutoCAD 内部的计算机语言，它是 AutoCAD 开放式体系结构的具体表现，它是 LISP 语言和 AutoCAD 有机结合的产物。使用 AutoLISP 可直接调用几乎全部的 AutoCAD 命令。AutoLISP 语言既具备一般高级语言的基本结构和功能，又具有一般高级语言所没有的强大图形处理功能，是当今世界上 CAD 软件中被广泛采用的语言之一。

美国 AutoDesk 公司在 AutoCAD 内部嵌入 AutoLISP 的目的是让用户充分利用 AutoCAD 进行二次开发，实现直接增加和修改 AutoCAD 命令，扩大图形编辑功能，建立图形库和数据库，并对当前图形进行直接访问和修改，开发 CAD 软件包等。在 AutoCAD 为用户提供的 AutoLISP、ARX、VBA 等开发工具中，AutoLISP 是一种简便易学的解释性语言，具有很强的数据表处理功能，是开发 AutoCAD 的一种重要手段。

AutoLISP 语言最典型的应用之一是实现参数化绘图程序设计，包括尺寸驱动程序，鼠标拖动程序等。尺寸驱动是指通过改变实体标注的尺寸值来实现图形的自动修改；鼠标拖动即利用 AutoLISP 语言提供的（GRREAD[<track>]）函数，让用户直接读取 AutoCAD 的输入设备（如鼠标），任选项追踪光标移动存在且为真时，通过鼠标移动光标，调整所需的参数值而达到自动改变屏幕图形大小和形状。

到目前为止，大多数参数化程序都是针对二维平面图编制的。实际上，立体图同样可以实现参数化绘图，在 AutoCAD 中编制实体的立体图参数化程序比其平面三视图程序更简单，而且立体图生成后，可以很方便地生成三视图、剖面图和轴侧图等。

AutoLISP 语言还能够利用 PDB 函数驱动 DCL（Dialog Control Language）文件创建自己的对话框。

自从在 AutoCAD 中嵌入 AutoLISP 以后，使仅仅作为交互式图形编辑软件的 AutoCAD 变成能真正进行计算机辅助设计、绘图的 CAD 软件。由于 LISP 灵活多样，又易于学习和使用，因此使 AutoCAD 成为功能很强的工具性软件。

1.1.2 AutoLISP 的特点

AutoLISP 具有如下特点：

① AutoLISP 语言是在普通 LISP 语言基础上，扩充了许多适用于 AutoCAD 应用的特殊

功能而形成的计算机语言，是一种仅能以解释方式运行于 AutoCAD 内部的程序设计语言。

② AutoLISP 语言表达式形式为前缀式表达式。

③ AutoLISP 语言中的一切成分都是以函数的形式给出的，它没有语句概念或其他语法结构。执行 AutoLISP 程序就是执行一些函数，再调用其他函数。

④ AutoLISP 语言把数据和程序统一表达为表结构，即 S-表达式，故可把程序当作数据来处理，也可把数据当作程序来执行。

⑤ AutoLISP 语言中的程序运行过程就是对函数求值的过程，是在对函数求值的过程中实现函数的功能。

⑥ AutoLISP 语言比较典型的程序结构是递归方式。由于递归方式的使用，使得程序设计简单易懂。

1.2 AutoLISP 数据类型

AutoLISP 语言主要用到如下数据类型：

整型数	（INT）
实型数	（REAL）
符号	（SYM）
字符串	（STR）
表（及用户定义的函数）	（LIST）
文件描述符	（FILE）
AutoLISP 的内部函数	（SUBR）
AutoCAD 的选择集	（PICKSET）
AutoCAD 的实体名	（ENAME）
函数分页表	（PAGETB）

本节只介绍前五种数据类型，其他类型将在后面相应的章节中介绍。

在上述数据类型中，前四种称为原子（ATOM），原子中包括数值原子（整型数和实型数）、符号原子和字符串原子。

所以 AutoLISP 语言最基本的数据类型是原子和表，它们又总称为符号表达式（Symbolic-Expression），也称为 S-表达式见图 1.1。其中原子（ATOM）是表的最基本元素，它本身只是一个值或符号。

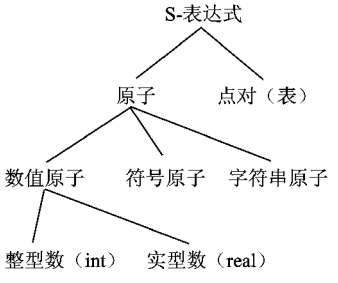

图 1.1 S-表达式

1.2.1 原子（ATOM）

（1）整型数（INT）

整数是由 0、1、2、…、9、+、–等字符组成，整数的大小与所使用的计算机系统有关。AutoLISP 支持 32 位有符号整数，范围从–2 147 483 648 到+2 147 483 647，如整数大小超出此范围，计算机将提示出错信息。但是（GETINT）函数只能取得 16 位有符号整数，范围从–32 728 到+32 767。

注意：在实际应用中，若设定和计算结果超出 AutoLISP 语言整数范围时，可改用实型数。

（2）实型数（REAL）

AutoLISP 支持双精度实数，并且至少有 14 位的精度，即整数后跟小数。如果实数的绝

对值小于 1，小数点前必须加 0，不能直接以小数点开头，否则被误认为“点对”而出错。

双精度实数（DOUBLE），是以 8 个字节存储的实数，共有 64 个位。实数在计算机内是以指数形式存储的。实型数的数值范围比整型数大得多，实型数范围约为$-1.797\ 693\times10^{308}$到$+1.797\ 93\times10^{308}$它不易超界，故用户可以尽量采用实型数。

数据在内存中的存储方式见表 1.1。

表 1.1　数据在内存中的存储方式

位	63	62	61~52	51~0
用途	符号位（+/–）	指数符号位（+/–）	指数位	基数位

其中：指数部分有 10 位 ，即$2^{10}=1024$，因此，最大的指数值是2^{1024}约等于$10^{308.55}$，所以实数的范围约在$\pm1.79\times10^{308}$之间。而基数用 52 位存储一个 0 到 1 之间的纯小数，即2^{52}约等于$10^{15.65}$，所以有效位数大约为 16 位，这就是实数运算产生误差的主要原因。

如果运算后产生实数的绝对值大于1.79×10^{308}时，AutoLISP 会返回±1.#INF（正或负无限大），这种情况称为 Overflow（高溢）。

AutoLISP 最小的正实数约为2.228×10^{-308}，如果运算后产生实数的绝对值小于这个数，便会出现错误，这种情况称为 Underflow（低溢）。

测试实数范围，可用（EXPT）函数实现。

例如：

命令：(EXPT 10.0 308)　　返回 1.0e+308

命令：(EXPT 10.0 308.55)　　返回 1.#INF；Overflow（高溢）

命令：(EXPT 10.0 –308)　　返回 错误：出现异常：0xC0000093

警告: 忽略展开异常

实型数也可采用科学记数法表示，如0.12×10^{19}可表示为 0.12E+19。

注意：双精度实数的有效位可达 16 位，但实际中用 14 位，这里是考虑误差的因素，而 AutoLISP 指令行响应的一般为 6 位有效数字。

（3）符号(SYMBOL)

① 符号(SYMBOL)包括除左右圆括号 “( )”、小数点“.”、单引号“'”、 双引号“"”、分号“;”及全部由数字组成的字符之外的任何可打印字符。

② 符号原子的长度没有限制，命名时要以能够表达清楚变量的含义为主，但尽量不要超过 6 个，否则要占用额外的内存，降低运行速度。

③ 在 AutoLISP 中符号的大小写是等效的，如以下的符号原子都是合法的。

A　　A12　　PC　　X-38-6　　*A

④ AutoLISP 中的任何符号都是有值的，即符号都要赋以一定的数值，或者说符号总是约束在一定值上。一般用赋值函数 SETQ 进行赋值。

例如：(SETQ　x　25.0)

含义是将 25.0 赋给 x，这时 x 的当前约束值即为 25.0。一个符号在使用前如没有赋以任何值，则该符号的值为 NIL(空)，它不占用内存空间。

⑤ 符号名避免使用 AutoLISP 的内建函数、常量名称，AutoCAD 的命令、系统变量，acad.pgp 文件内定义的外部命令等。

注意：为区别起见，常用术语“符号”来指存储静态数据的一个符号名，例如内建式函数和用户定义函数名是一个符号。

用术语“变量”来指存储程序数据的符号名，如上述(SETQ x 25.0)中的变量名为 x，它的值为 25.0。AutoLISP 程序中每一个变量都要消耗少量内存，故当变量值不再有用时，重复使用变量名或将变量值设置成 NIL 是良好的程序设计习惯。符号名或变量名不能包含空格字符或分隔符，并总是以字母开头。

⑥ 常量。在程序运行过程中其值保持不变的量称为常量；AutoLISP 有 4 个内建常量，用户在设定变量或自定义函数时，要避免和这 4 个常量同名。

T/t　　逻辑真值；

NIL/nil　　逻辑假值，同时也代表空值(或空表)；

Pi　　圆周率π值，约等于 3.141592654；

Pause　　双反斜线“\\”字符，用于(COMMAND)函数等待用户输入。

（4）字符串(STR)

字符串(STR)是由包含在一对双引号内的一组字符组成的。

例如：

"ABC"　"135"　"Ab C"　" "

字符串可以包括任何可打印的字符。字符串中字母的大小写及空格都是有意义的。若字符串中没有任何字符，则为空串""。

当用户在 AutoLISP 表达式中直接使用引号引起来的字符串时，该值被称为字符串常量。

例如："string 1"和"\n Enter first point: "都是有效的字符串。

在用引号引起来的字符串中，用反斜杠“\”字符可以添加控制字符（或换码代码），即反斜杠“\”作为控制字符，与其后的字符组成，控制字符及含义见表 1.2。

表 1.2　控制字符及含义表

控 制 字 符	含　义	用 ASCII 码表示
\\	表示反斜线“\”字符	\114
\"	表示双引号“"”	\042
\e	表示换码字符（Esc）	\033
\n	表示换行	\012
\r	表示回到行首	\015
\t	表示移到下一个定位(Tab）	\011
\nnn	表示八进制码为 nnn 的字符	

注意：“\”后面的字符 e、n、r、t 必须为小写字母。

1.2.2　表和点对

（1）表(LIST)

在 AutoLISP 语言中，表作为一种基本数据类型，它有如下特点：

① 表是指放在一对相匹配的左、右圆括号中的一个或多个元素的有序集合。

② 表中的每一个元素可以是任何类型的 S-表达式，既可以是数字、符号、字符串，也可以是表。

③ 表中元素与元素之间至少要用一个空格隔开，而元素与括弧之间可不用空格，因为括弧本身就是有效的分隔号。

例如：　　(15　(a　b)　c　d)

在此例中，表内有 4 个元素，即 15、(a　b)、c 和 d，其中第二个元素又是一个表。

④ 表是可以任意嵌套的，上例表中即嵌套了一个表(a　b)。表可以嵌套很多层，从外层向里依次称为 0 层(也称顶层)、1 层、2 层、…我们所说表中的元素是指表的顶层元素，即 0 层元素。

⑤ 表中元素是有顺序的，为便于对表中元素进行存取，每个元素都有一个序号。从左向右，第一个元素序号为 0，第二个元素序号为 1，第 i 个元素序号为 i–1。

⑥ 表的大小为表的长度，即表中顶层元素的个数。没有任何元素的表称为空表。空表用()或 NIL 表示。在 AutoLISP 语言中，NIL 是一个特殊的符号原子，它既是原子又是表。

⑦ 表有两种基本类型：标准表和引用表。

a．标准表：标准表是 AutoLISP 程序的基本结构形式，AutoLISP 程序就是由标准表组成的。标准表是用于函数的调用，其中第一个元素必须是系统内部函数或用户定义的函数，其他的元素为该函数的参数，如上面提到的赋值函数的调用，即采用标准表的形式。

(SETQ　x　25.0)

表中第一个元素 SETQ 为系统内部定义的赋值函数，x 和 25.0 均为 SETQ 的参数。

b．引用表：这种表第一个元素不是函数，即不作为函数调用，常作为数据处理，在程序中以如下两种形式存在：

'(a　d　b)或(QUOTE (a　d　b))

引用表的一个重要应用是表示图中点的坐标。当表示点的坐标时，表中的元素是用实型数构成的。

表示二维点的坐标是用两个实型数构成的表，如(20.0　30.5)，其中第一个元素表示点的 X 轴坐标，第二个元素表示点的 Y 轴坐标。三维点的坐标表示，是用三个实型数构成的表，如(20.0　82.5　1.0)，其中三个元素依次表示点的 X 轴坐标、Y 轴坐标和 Z 轴坐标。

（2）点对(DOTTED　PAIR)

点对也是一种表，该表中只有两个元素，两元素中间为一圆点“.”，且圆点与元素之间必须用空格分开。

例如：(A　.　B)就是一个点对，A、B 与圆点均用空格分开，其中第一个元素 A 为该点对的左元素，第二个元素 B 为点对的右元素。点对亦可任意嵌套。当使用点对时，切记要注意它的书写格式。

例如：(X　.　(B　.　(Y　.　Z)))　为合法点对，而(X　.(B　.　Y) .　Z)即为非法的。

点对常用于构造关联表。

1.3　AutoLISP 的程序结构

AutoLISP 语言没有“语句”这一术语，AutoLISP 程序一般是由一个或一系列按顺序排列的标准表所组成。

例如：(SETQ　x　25.0)

是上面提到的标准表，又可以看作是一个 AutoLISP 程序。

建议按照如下原则书写 AutoLISP 源程序：

① 先写出对称的括号，再填入其内的函数和参数。

② 适当的缩排，使程序各标准表间的主从关系明确化。

③ 做必要的空行，以区分程序中的各个单元。

④ 尽量加上注释，最好连程序流程、变量的类型与用途、用到的 AutoCAD 系统变量与命令都加以说明，方便程序维护。程序注释可以使用任何语言，但不影响程序的执行。

首先给出几个基本问题实例。

【例 1】 在图形屏幕上，画一个圆心在（5　5），半径为 8 的圆。

```
(DEFUN　mm( )
(SETQ　r　(GETREAL "\n 半径:" ))
(SETQ　p　(GETPOINT "\n 中心点:"))
(COMMAND　"circle"　p　r )
)
```

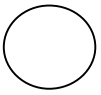

图 1.2　圆

程序执行结果见图 1.2。

【例 2】 编程：随机输入两点坐标 p1、p2，求两点距离及两点连线的方位角，并画出此线段见图 1.3，程序文件名为 PROG1.lsp。

图 1.3　线段

```
( DEFUN　mm1()
    (SETQ　p1 (GETPOINT "\n p1:")
              p2 (GETPOINT "\n p2:")
     )
    (SETQ　d　(DISTANCE　p1 p2))
    (SETQ　ang　(ANGLE　p1　p2))
    (PRINT　d)
    (PRINT　ang)
    (COMMAND　"line"　p1　p2　"")　　；在 p1 与 p2 两点间画一条直线
)
```

执行结果：

```
命令：(LOAD　"C:PROG1.LSP")　　返回 mm1
命令：(mm1)
命令：p1:30,45
命令：p2:123,89
      139.461
      0.439063　nil
```

【例 3】 文件名为 PQ.lsp 的 AutoLISP 文件是由以下程序组成的。

```
(DEFUN　mm2()
    (SETQ　x　25.0)
    (SETQ　y　12.2)
    (+ (*　x　y)　x)　；表示 x*y+x
)
```

以上是由三个标准表组成的程序，每个标准表的第一个元素均为系统提供的函数（如：SETQ，+，*）称为系统的内部函数。SETQ 为赋值函数，+为加函数，*为乘积函数。标准表中的其他元素为相应函数的参数。这个程序是将 25.0 赋给变量 x，将 12.2 赋给变量 y，求变量 x 和 y 的值的乘积，再求此乘积与 x 的和。

AutoLISP 语言源程序的书写格式有如下特点：

① 由于 AutoLISP 语言的一切成分都是函数，而所有函数又以表结构形式存在，所以

AutoLISP 程序的所有括号都需要左右匹配。

② AutoLISP 程序阅读函数时，按从左到右的规则进行。

③ 函数必须放在表中第一个元素的位置，如赋值函数 SETQ、算术运算函数+、*等应为表中第一个元素，即放在操作数之前，而不是放在它们的中间，这与算术运算的书写格式不同，初学者可能会感到不习惯。表中的函数与参数，各参数之间均至少要一个空格来分开。

④ 两个表之间和表内的多余空格和回车是不需要的，故一个表可占多行，一行也可写多个表，如 PQ.LSP 程序可写成如下形式：

```
(DEFUN  mm2()
(SETQ  x  25.0)  (SETQ  y  12.2) (+ (*  x  y)  x)
)
```

⑤ AutoLISP 程序中可以使用分号“;”作注释。注释的作用是对程序作解释。AutoLISP 求值器总是忽略每一行中分号以后的部分，且注释可放在程序中的任何地方。

⑥ AutoLISP 源程序是以扩展名为“.LSP”的 ASCII 码文本文件。

执行 AutoLISP 程序就是对一个个 AutoLISP 函数的调用。函数是 AutoLISP 语言处理数据的工具，学习掌握 AutoLISP 语言，核心就是要掌握 AutoLISP 函数。AutoLISP 函数分为系统内部函数和用户定义的外部函数。AutoLISP 提供了大量的系统内部函数，以满足编程的需要。

AutoLISP 对函数的调用是通过标准表来实现的。如前所述，AutoLISP 程序的基本结构就是由一系列有序标准表构成的。AutoLISP 程序的运行，就是对标准表依次进行求值。标准表或者说函数调用的一般格式如下：

(函数名[<参数 1>][<参数 2>]……[<参数 n>])

标准表中的第 1 个元素必须是函数名，以后的各元素为该函数的参数，参数的类型及数目取决于函数。为了便于读者更好地了解和掌握函数调用的格式，本书对所用的符号特做如下的约定。

<……>　尖括号中的内容表示函数所要求的参数的类型，它是必须存在的。如：
　　<参数 1 >

[……]　方括号的内容是任选项，它可以存在，也可省略，但该项只能出现一次。如：
　　[参数 1]

……　省略号，表示省略前面同样的参数，其数目不限。

|　表示两边的元素只可选其一。如：
　　参数 1 |参数 2

学习 AutoLISP 的系统内部函数时，必须掌握以下的基本内容。

① 函数调用格式：即函数名、函数要求的参数个数和类型。

② 函数的功能：即该函数的功能和作用，以及它对其参数如何进行处理。

③ 函数的求值情况：即哪些参数要求值，哪些不被求值。

④ 函数求值结果的返回值类型：这点很重要，因为大多数函数的返回值都要被其他函数接受，而每个函数所需要的参数都有特定的类型。因此只有清楚被调用函数的返回值的类型，才不会因用错函数的参数而出错。在本书以后各章中将分别介绍 AutoLISP 系统内部函数。

1.4 AutoLISP 的运行环境

AutoLISP 没有单独的运行环境，因为 AutoLISP 程序工具是附在 AutoCAD 的软件包内，安装 AutoCAD 时会一并被安装进去。所以，使用 AutoLISP 并不需要增加任何软硬件配备，只要能执行 AutoCAD 的计算机即可满足 AutoLISP。

1.5 AutoLISP 的内存分配

AutoLISP 的变量（整型、实型、字符串等）、用户定义的函数以及标准函数，在 AutoLISP 的编辑过程中存储在计算机的内存中。当运行 AutoLISP 程序时，它还需要两个很大的内存区域。

① HEAP（堆区域），它存储所有的函数和变量。因此程序使用的函数和变量越多，或函数越复杂，则“HEAP”空间占得也就越多。

② STACK（栈区域），它存储函数的变量和局部结果。因此“嵌套”的函数越深，或函数执行的递归的次数越多，那么所用的栈空间也越多。

AutoLISP 隐含的堆和栈空间大小为：

HEAP=25000B

STACK=20000B

在 AutoCAD 下运行 AutoLISP 不能扩展它的堆或栈空间。如果用户定义的函数和变量太多，以至于用光了所有的堆空间，AutoLISP 将显示下列信息：

Insufficient node space

并且中止当前函数的执行。如果在执行 AutoCAD 时，没有足够的内存装入 AutoLISP，则显示下列信息：

Insufficient memory—AutoLISP disabled.

直到有足够的内存后 AutoCAD 重新启动时，AutoLISP 才能恢复其功能。

1.6 AutoLISP 程序的执行过程

对于很短的 AutoLISP 程序，只由一至两个表所组成，如简单的数值函数的运算，或用 DEFUN 函数定义的简单用户函数，可直接在 AutoCAD 环境中的“命令:”提示符下输入即可，返回结果立即显示在文本屏幕上。

对于一般的 AutoLISP 应用程序，需采用 Visual LISP 编辑器进行编辑。在编辑器下编辑好的.LSP 程序（如下面的 TMP.LSP），当回到 AutoCAD 环境下，用 LOAD 函数装载后便可以执行了。执行时，如果程序中没有 DEFUN 函数，系统便边装入边运行；若有 DEFUN 定义的命令或函数，装载后只需再在“命令:”提示符下键入 DEFUN 函数定义的命令名或函数名即可运行相应的命令或函数。

AutoLISP 程序编辑及运行过程如下：

① 启动 AutoCAD；

② 打开 AutoCAD 的工具菜单，选择 AutoLISP（S）的 Visual LISP 编辑器（V）；

③ 打开 Visual LISP 编辑器（V）的文件菜单，选择新建文件进入编辑文件窗口，便可编写程序；

④ 程序编写完成后，保存程序；

⑤ 打开 Visual LISP 编辑器（V）的文件菜单，选择加载所保存的文件；

⑥ 在_$提示符下，用 LOAD 函数加载，其格式为：

（LOAD"文件名.LSP"）

加载成功返回，函数名

⑦ 在_$符号下或在 AutoCAD 的“命令：”提示符下执行函数。

【例 4】 在 NOTEPAD 下编辑下列 LISP 程序。

```
(DEFUN  C:TESTLISP( )
(PRINT  "\n Please Use COMMAND：TESTLISP\n"
(ALERT "AutoLISP Test  Function!")  ;在显示一个警告框，并在框内显示如图 1.4 所示信息
  )
)
```

将上述程序保存为 C:TMP.LSP。

启动 AutoCAD，在“命令:”提示符下输入

(LOAD "C:TMP.LSP")

图 1.4　AutoCAD 信息显示

按 Enter 键后，即执行第 1 句并返回最后一个 DEFUN 函数定义的函数名：

C:TESTLISP

装载完后再运行 DEFUN 定义的函数，在“命令：”提示符下输入(C:TESTLISP)或 TESTLISP，即执行该程序段。

注意：

① 在 AutoCAD 工作目录下有一个 ACAD.LSP 文件，它是当 AutoCAD 启动、新建文件（NEW）、打开文件（OPEN）时自动装载的 AutoLISP 程序。用户可以修改它，实现一定的目的。例如，用户想要在 AutoCAD 启动时自动装入自己定义的函数或程序，则可以在 ACAD.LSP 程序中加入（DEFUN XXX()）程序段或(LOAD "XXX")函数。

② 在装入 ACAD.LSP 文件时若出现一条 AutoLISP 错误，剩余的文件会被忽略而不装入，并提示出错。如果一个 LOAD 函数的调用是成功的，它返回被加载文件中最后的那个表达式的值。

练习题

1．AutoLISP 语言主要有哪些特点？

2．AutoLISP 语言的基本数据类型有哪些？

3．AutoLISP 语言符号的命名规则是什么？

4．AutoLISP 程序书写的 4 个基本原则是什么？

5．AutoLISP 语言规定了哪几个内部常量，其含义是什么？

6．摄氏温度与华氏温度换算的计算式，如何以 AutoLISP 语句表示？

F=9/5 * C + 32

C=5/9 (F – 32)

7．写出下列表达式的 AutoLISP 语言表达式。

（1）$\dfrac{88-52\times 63}{18+47+3}$

（2）$\dfrac{2\sin\dfrac{\pi}{4}}{\mathrm{e}}$

（3）$G\cdot\dfrac{m_1\cdot m_2}{r^2}$

（4）$2\alpha^{r}\sqrt{\dfrac{m}{k}}$

（5）$\dfrac{V_0^2\sin^2\alpha}{2g}$

（6）$D\cdot\dfrac{i\cdot A/N}{1-(i/n+1)^{-n\cdot y}}$

（7）$\dfrac{R_1+1}{\dfrac{1}{R_2}+\dfrac{1}{R_3}+\dfrac{1}{R_4}}$

（8）$109887\times\left(\dfrac{1}{2^2}-\dfrac{1}{n^2}\right)$

（9）$x^3-9.6x^2+25.837x-19.25$

（10）$\sqrt[5]{(-a)^2+(-b)^2}$

（11）$a+\dfrac{b}{\tan x+1}$

（12）$\dfrac{\sin 45^\circ-\left[(2x+3)+5\right]}{(5+x)(x-8)}$

第 2 章　数值函数

数值函数是 AutoLISP 最基本的函数之一。

数值函数用于处理整型数和实型数，它包括：基本标准函数、三角函数以及布尔运算函数。数值函数总是返回数的数值，返回值的数据类型取决于参数表中参数的数据类型。

数的运算遵循以下规则：

① 若参数表中的所有参数都为整型数，则求值器对参数表中的参数做整数运算，返回整数值。例如(/　17　3)返回值为 5 而不是 5.6667。

② 若参数表中有一个实型数，则对参数表中的参数进行浮点数学运算，返回实型数。例如(/　17　3.0)返回值为 5.6667。

③ 若参数表中的参数多于两个，则从前到后，遵循前两条规则，每两个参数进行数值运算，再把运算结果与下一个参数进行运算。例如(/　17　3　2.0)求值器先进行 17 除以 3，得 5，再用 5 除以 2.0 得 2.5，该函数最后返回值为 2.5。

注意：AutoCAD 提供一个数学求值器，此求值器可在 AutoCAD 的“命令:”后使用，如上述函数使用如下。

命令: (/　17　3　2.0)
　　2.5　　；屏幕显示运算结果 2.5

2.1　计算函数

2.1.1　(＋　<数 1><数 2>…)

本函数返回所有<数 i>的和。其中的数可以是整型的，也可以是实型的，如果所在的<数 i>都为整型数，其结果也为整型数。如果其中有一个是实型的，那么其他整型数都将被转换为实型的，结果是实型数。如果本函数仅提供了一个<数>，则函数返回<数>与零相加的结果。如果不提供数，则返回零。注意函数返回值的范围。

例如：　(+　4)　　返回 4
　　(+　2　3)　　返回 5
　　(+　1　2　3　4.5)　　返回 10.5000
　　(+　1　2　3　4.0)　　返回 10.0000

2.1.2　(－　<数 1> <数 2>…)

此函数将第一个数减去以后所有数之和，并返回最后结果。若仅给出一个数，即返回零减这个数。或不提供数，则返回零。

函数中的<数 i>可以是实型数或整型数，按标准规则进行类别转换。例如：

(– 50 40) 返回 10
(– 50 40.0) 返回 10.0000
(– 50 40 10.0) 返回 0.0000
(– 8) 返回 –8

2.1.3 (1+ <数>)和(1– <数>)

其参数只有一个<数>，“1+”、“1–”必须连写，中间无空格，数加 1 或减 1，并返回最后结果。

函数中的<数>可为实型数或整型数，返回值类型取决于<数>的类型。例如：

(1– –3) 返回-4
(1+ 3) 返回 4
(1– 3) 返回 2
(1+ 2.0) 返回 3.0000
(1– 2.0) 返回 l.0000

2.1.4 (* <数 1> <数 2>…)

函数返回所有<数 i>的乘积，返回值类型取决于参数类型，只要参与运算的<数 i>中有一个是实型数，则结果为实型数；只有所有参与运算的数全部为整型数，其结果才为。如果本函数仅提供了一个<数>，则函数返回<数>与 1 相乘的结果。若不提供<数>，则返回零。

注意：函数返回值的范围。

例如：

(* 3) 返回 3
(* 3 2 7) 返回 42
(* 3 (+ 1.0 4)) 返回 15.000

2.1.5 (/ <数 1> <数 2>…)

本函数返回<数 1>除以<数 2>，再除以<数 3>…依次做除法运算的结果。如果仅提供了一个<数>，则返回<数>除以 1 的结果。若函数不提供<数>，则返回零。

各个<数>类型不同，计算结果不同，返回值类型也不同。

例如：

（/ 9 2） 返回 4
（/ 9 2.0） 返回 4.5000
(/ 9 (/ 2 3)) error：divide by zero
(/ 9 (/ 2.0 3)) 返回 13.5000
(/ 4) 返回 4 （相当于 4÷1=4）

2.1.6 (REM <数 1> <数 2>…)

函数返回<数 1>除以<数 2>的余数，若参数多于两个，则将<数 1>除以<数 2>的余数再除以<数 3>得余数……即为运算结果。

例如：

(REM 42 12) 返回 6
(REM 42 12.0) 返回 6.0000
(REM 20 2) 返回 0

(REM　36　5　2)　　返回 1，此式相当于(REM(REM　36　5) 2)
(REM　3)　　返回 3
注意：余数的符号与被除数的符号相同。
(REM　–13　5)　　返回–3
(REM　13　5)　　返回 3
(REM　13　–5)　　返回 3
(REM　–13　–5)　　返回–3

2.1.7　(GCD　<数 1> <数 2>)

该函数返回<数 1>，<数 2>的最大公约数(greatest common factor)。
注意：
<数 1>，<数 2>必须为正整数。
例如：
(GCD　81　57)　　返回 3
(GCD　81　80)　　返回 1
(GCD　12　20)　　返回 4
(GCD　81)　　提示：错误: 参数太少

2.1.8　(MAX　<数 1>　<数 2>…)
(MIN　<数 1>　<数 2>…)

该函数返回<数 i>(i=1，2，3，…)中的最大者或最小者。
例如：
(MAX　4.07　–2)　　返回 4.0700
(MAX　4　9.0　2)　　返回 9.0000
(MIN　3　2　–1)　　返回–1
(MAX　4　9.0　2)　　返回 9.0000
(MIN　73.0　2　48　5)　　返回 2.0

2.1.9　(EXP　<数>)

该函数返回 e 的<数>次幂的值。
注意：返回值为实型。
例如：
(EXP　1.0)　　返回 2.718282(即 e^{1})
(EXP　0)　　返回 1.000000（即 e^{0}）
(EXP　–0.4)　　返回 0.670320(即 $e^{-0.4}$)
(EXP　2.2)　　返回 9.02501（即 $e^{2.2}$）

2.1.10　(EXPT　<底数>　<幂>)

该函数返回<底数>的<幂>次方，如果底数和幂都是整数，其结果也是整数；否则，结果为实数。
例如：
(EXPT　3　2)　　返回 9
(EXPT　3.0　2)　　返回 9.0000

或 (EXPT 3 2.0) 返回 9.0000
(EXPT 3 -2) 返回 0
(EXPT 3.0 -2) 返回 0.111111

2.1.11 (LOG <数>)

该函数是 EXP 的反函数，返回值为<数>的自然对数值，其数据类型为实型数。
例如：
(LOG 3) 返回 1.098610
(LOG 1) 返回 0.00000
(LOG 1.22) 返回 0.198850
(LOG –90) 返回错误：没有为参数定义函数: –90

2.1.12 (SQRT <数>)

该函数返回<数>的平方根，其数据类型总为实型数。要求<数>大于等于零。
例如：
(SQRT 9) 返回 3.0000
(SQRT 9.0) 返回 3.0000
(SQRT(/ 4 2.0)) 返回 1.4142
(SQRT –9) 返回错误: 没有为参数定义函数: –9
注意：<数>为大于零的正数。

2.1.13 (ABS <数>)

该函数返回<数>的绝对值，其中<数>可为实型数或整型数。
例如：
(ABS 0.0) 返回 0.0
(ABS 100) 返回 100
(ABS –100) 返回 100
(ABS –2.1) 返回 2.100000

2.1.14 (MINUSP <数>)

函数检查一个数是否是负数，若<数>为负数，则函数返回 T；否则，返回 NIL。
例如：
(MINUSP -2) 返回 T
(MINUSP 3.1) 返回 NIL
(MINUSP 0) 返回 NIL

2.1.15 (ZEROP <数>)

函数检查一个<数>的求值是否为零，若为零，则返回 T；否则，返回 NIL。
例如：
(ZEROP 0) 返回 T
(ZEROP 0.0) 返回 T
(ZEROP 0.0001) 返回 NIL

2.1.16　(NUMBERP　<项>)

函数检查<项>是否是一个实型数或整型数。如果<项>是一个实型数或是一个整型数，该函数返回 T；否则，返回 NIL。

例如：

(NUMBER　4)　　返回 T
(NUMBER　B)　　返回 NIL
(NUMBER　4.0)　　返回 T
(NUMBER (EVAL B))　　返回 T
(NUMBER　A)　　返回 T
(NUMBER　"HU")　　返回 NIL

2.1.17　(FLOAT　<数>)

该函数将一个<数>转换成实型数后返回。此函数有时非常有用，如在除法函数中，通过 FLOAT 函数强制把数转换为实型数，从而使可能为整除的运算变为浮点除运算。

例如：

(F1OAT　3)　　返回 3.0
(FLOAT　3.75)　　返回 3.75
(FLOAT (–　34　2　3.7))　　返回 28.3

2.1.18　(FIX　<数>)

该函数忽略实数<数>的小数部分，将<数>的整数部分返回。

例如：

(FIX　3)　　返回 3
(FIX　3.7)　　返回 3
(FIX　–3.99)　　返回–3
(FIX (/　34.67　23))　　返回 1

注意：如果<数>大于最大可能的整数或小于最小可能的整数（在 32 位的平台上分别是+2 147 483 647 和–2 147 483 648），FIX 返回截去小数部分后得到的实数（尽管 AutoLISP 和 AutoCAD 之间的整数传送被限制在 16 位值范围内）。

2.2　布尔运算函数

2.2.1　(LOGAND　<整数>　<整数>…)

该函数返回一个整型数表的各数按位逻辑与(AND)的结果。当 LOGAND 函数表中不含参数时，则返回 0。

例如：

(LOGAND　7　15　3)　　返回 3
(LOGAND　2　3　15)　　返回 2
(LOGAND　8　3　4)　　返回 0

注意：各数以二进制形式按位与。

2.2.2　(LOGIOR　<整数>　<整数>…)

该函数返回一个整型数表的各数按位逻辑或(OR)的结果。当 LOGIOR 函数表中不含参数