超值奉送

Auto LISP 入门

Auto LISP 是在 AutoCAD 内部允许的一种内嵌式程序设计语言。LISP（List Processing Language）是人工智能领域中广泛采用的一种表处理语言，具有较强的表处理功能，主要用于人工智能、机器人、专家系统、博弈、定理证明等领域。LISP 也被称为符号式语言，因为它处理的对象是符号表达式。

Auto LISP 可以直接调用几何所有的 AutoCAD 命令，为 AutoCAD 提供了使用高级语言开发编程的途径，使得用户能充分利用它对 AutoCAD 进行二次开发。

感谢：

感谢您选购本图书。

本部分是《AutoCAD 2007 中文版自学手册——入门提高篇》图书超值赠送的一部分，仅供读者个人参考使用。未经许可，任何人不得将此部分用于其他商业用途。

欢迎您对我们的图书进行监督，并对我们的工作（图书质量、装帧设计、封面、印刷等）提出改进意见或建议。

网站：http://www.fr-cad.net

E-mail: editor.liu@gmail.com

QQ 群：9843746 （CAD/CAM/CAE 应用方向）

您的支持是我们前进的动力，您的需要是我们努力的方向！

冯如设计在线

2006-10-09

A.1 Auto LISP 基本函数

自从 AutoCAD RA.01 开始，出现了 Visual LISP 语言，它是 Autodesk 公司为增强 Auto LISP 程序开发能力而设计的软件工具，为 Auto LISP 增加了许多新的函数。Visual LISP 的集成开发环境提供很多特性，使用户创建和修改源代码、测试和调试程序更加方便。

自从 Auto LISP 嵌入 AutoCAD 之后，使仅仅作为交互式图形编辑软件的 AutoCAD，通过编程使之能真正进行计算机辅助设计、绘图。由于 LISP 灵活多变、易于学习和使用，因而使 AutoCAD 成为功能很强大的工具性软件。

随着 AutoCAD 版本的升级，Auto LISP 的功能不断得到扩充和增强，主要功能如下：

- 能够把数据和程序统一表达为表结构，即 S-表达式，因此可以把程序当作数据来处理，以可以把数据当作程序来执行；
- 有图形处理和程控方面的函数，可以直接调用 AutoCAD 的所有命令；
- 扩充了 AutoCAD 应用的特殊功能；
- 主要控制结构采用递归方式，使得程序设计简单易懂。

A.1.1 赋值函数

Auto LISP 提供了大量系统预定义的函数。函数的一般描述格式为：

（函数名 <变元> [<变元> ……]）

表中第一个元素为函数名，它一般指明函数的功能。其余各元素为函数的变量（称为变元）。[ ]表示任选项；……表示任意多项；<>表示变元类型。函数调用后显示一个执行该函数的结果，称为返回值。下面是 Auto LISP 的基本函数。

1. (setq 符号 表达式[符号 表达式]……)

将一个或者多个符号设置为相应表达式的值，返回最后一个表达式的值，例如：

（setq a 123 b 6.0） ；符号 a 返回 6.0

（setq s “it” x ‘(a b) ） ；符号 s 被赋值为”it”，符号 x 被赋值为（a b）,返回（a b）

2. (set ‘符号 表达式)

如果符号前有一个单引号，该函数则将符号设置为表达式的值，此用法与 setq 函数的用法等价，例如：

(set ‘ a 18.0) ;符号 a 被赋为 18.0，返回 18.0

Set 和 setq 函数都需要将符号作为它的第一个参数。但是 set 函数接受返回结果为符号的表达式作为参数，setq 却不可以。

3.（eval 表达式）

返回对表达式求值的结果，例如：

（setq a 24 b 45） ；返回 45

（eval 10.5 ） ；返回 10.5

（setq c (+1 2 3)） ；

（eval c ）；返回 6

A.1.2 数值计算函数

这类函数有一个共同特点：参数既可以是整数，也可以是实数，如果所有的数都是整数，返回的结果就是整数；如果有的是实数，返回的结果也就是实数。

1.（+[数 数]……）

返回所有数之和。如果只有一个数，则返回它本身；如果不提供数，则返回 0。例如：

（+ 2 3 4）； 返回 9

2.（-[数 数]……）

返回第一个数减去其它所有数之和的差。如果只提供一个数，则返回它本身。如果不提供数，则返回 0。例如：

（- 50 30.0 5.5）； 返回 A.5

3.（*[数 数]……）

返回所有数的乘积。如果调用本函数只提供一个数，则返回它本身；如果不提供数，则返回 0。例如：

（* 5 3 4.0）； 返回 60.0

4.（/ [数 数]……）

返回第一个数除以其它所有数乘积的商。如果只提供一个数，则返回这个数除以 1 的结果；如果不提供数，则返回 0。例如：

(/130 13 2); 返回 5

5.（1+ 数）

返回一个数加 1 后的结果。例如：

（1+ 4）； 返回 5

6.（1- 数）

返回一个数减 1 后的结果。例如：

（1- 10）； 返回 9

7.（abs 数）

返回一个数的绝对值。

8.（atan 数 1[数 2]）

如果只提供数 1，则该函数返回数 1 的反正切值（单位为弧度）。如果提供了数 1 和数 2，则该函数返回数 1/数 2 的反正切值。如果数 2 为 0，则该函数返回正的或负的 1.570796 弧度，其正

负取决于数 1 的正负。该函数返回角度的范围是-PAI/2~PAI/2(弧度)。例如：

(atan 0.5); 返回 0.463648 弧度

(atan 2.0 3.0); 返回 0.58803 弧度

9.（COS 角度）

返回角度的余弦值，角度的单位为弧度。例如：

(cos 0.2) ; 返回 0.980067

10.（exp num）

返回常数 e(2.718282……)的指定幂，参数 num 为实数。例如：

(exp 1.0) ; 返回 2.718282

11.（expt number power）

返回以 number 为底数的 power 次幂的值。例如：

(expt 3 3) ; 返回 27

12.（fix 数）

返回将数截去小数部分的结果。例如：

(fis 4.5) ; 返回 4

A.（gcd 正整数 1 正整数 2）

返回两个正整数的最大公约数。例如：

(gcd 30 20) ; 返回 5

14. (log num)

返回 num 的自然对数。参数 num 为正数，返回值是实数。例如：

（log 4.5 ）；返回 1.50408

15. (max [数 数……])

返回给定各数中的最大者。如果没有提供数，则返回 0，例如：

（max 1 3 5 9 ） ；返回 9

16. (min [数 数……])

返回给定各数中的最小者。如果没有提供数，则返回 0，例如：

（min 1 3 5 9 ） ；返回 1

17. (rem [数 数])

返回第 1 个数除以第 2 个数的余数。例如：

（rem 40 6 ） ；返回 4

18. (sin 角度)

返回角度（单位为弧度）的正弦值。例如：

（sin 1.0） ；返回 0.841471

19. (sqrt　数)

返回一个数的平方根。例如：

（sqrt　9　）；返回 3

A.1.3 关系函数

1. （＝ [数或者字符串] [数或者字符串]）

在该函数中，判断数字或者字符串的参数是否相等。如果所有的参数都相等，返回 T；否则返回 nil。如果只提供一个参数，返回 T。如：

（＝ 5.20 5.2）；　　返回 T

（＝ “HELLO”　“hello”）；　　返回 nil

2. （/＝ [数或者字符串] [数或者字符串]）

在该函数中，判断数字或者字符串的参数是否不等。如果相邻两个数或者字符串的值不相等，返回 T；否则返回 nil。如果只提供一个参数，返回 T。如：

（/＝ 4　5.2　6）；　　返回 T

（/＝ 4　5　5）；　　返回 nil

3. （< [数或者字符串] [数或者字符串]）

在该函数中，判断数字或者字符串的参数是否小于。如果每个参数的值都小于它右边参数的值，返回 T；否则返回 nil。如果只提供一个参数，返回 T。如：

（< 9　18）；　　返回 T

（< “l”　“ly”）；　　返回 nil

4. （> [数或者字符串] [数或者字符串]）

在该函数中，判断数字或者字符串的参数是否大于。如果每个参数的值都大于它右边参数的值，返回 T；否则返回 nil。如果只提供一个参数，返回 T。如：

（> 1009　901　）；　　返回 T

5. （<＝ [数或者字符串] [数或者字符串]）

在该函数中，判断数字或者字符串的参数是否小于等于。如果每个参数的值都小于或等于它右边参数的值，返回 T；否则返回 nil。如果只提供一个参数，返回 T。如：

（<＝ 9.01　9.010　10.09）；　　返回 T

（<＝ “love”　“hate”）；　　返回 nil

6. （>＝ [数或者字符串] [数或者字符串]）

在该函数中，判断数字或者字符串的参数是否大于等于。如果每个参数的值都大于或等于它右边参数的值，返回 T；否则返回 nil。如果只提供一个参数，返回 T。如：

（>＝ 27　18　9）；　　返回 T

（>= 603 1003 825）； 返回 nil

A.1.4 输入函数

Getxxx 函数是 AutoLISP 提供的用户输入函数，每个用户输入该函数时，都暂停下来等待用户输入该函数指定类型的数据值，并且返回输入的数据，在该函数暂停之前，应用程序可以指定一些要显示的任选的提示。

1. （getint [msg]）

以交互方式等待用户输入一个整型数，并将此值赋给“（）”前的变量，参数 msg 为提示信息（下同）。如：

（setq n （getint “请输入一个整数: ”））

> 注意：用户不能在提示信息中输入另外一个 AutoLISP 表达式来响应 getint 函数的请求。下同。

2. （getreal [msg]）

以交互方式等待用户输入一个实数（浮点数），并将此值赋给“（）”前的变数。如：

（setq y（getreal “请输入一个实数: ”））

3. （getstring [cr] [msg]）

等待用户输入一个字符串，并将此字符串赋给“（）”前的参数。参数 cr 控制空格键是否代替<Enter>键，如果该参数不为 nil，那么输入的字符串可以包括空格，此时只能按<Enter>键结束输入。字符串最大长度为 132 位，如果超出 132 位，则只返回前 132 位字符。如：

（setq w （getstring y “do you love me ? ”））

do you love me ?Yeah!

" Yeah! "

4. （getpoint [pt] [msg]）

等待用户输入一个点，并将此点赋给“（）”前的变数。用户可以使用光标来拾取点，也可以在命令行以坐标形式输入。参数 pt 是以现行 UCS 表示第一个 2D 或 3D 基点的坐标值（下同）。如：

（setq x（getpoint “第一点: ”））

1，2

（2.0 3.5 6.9）

5. （getdist [pt] [msg]）

以交互方式等待用户输入一段距离。用户可以输入一个数值，也可以输入两个点来指定距离。该函数要求有一个基于现行 UCS 的基点变量，当用户在屏幕上移动十字光标时，它会从这个基点开始绘制一个矩形。如：

（setq　len（getdist　“输入长度:”））

6. （getangle　[pt]　[msg]）

等待用户输入一个方位角，并将此角度值以弧度方式赋给“（）”前的变量。该函数以逆时针方向测量由用户指定的方位角或两点所确定的直线与 X 轴的夹角。

了解输入的角度和 getangle 函数所返回的角度之间的区别，是非常重要的。传送给该函数的角度，是基于 AutoCAD 的系统变量 ANGDIR 和 ANGBASE 的现行设置而确定的。然后，一旦输入了一个角度，则它就以 ANGBASE 的现行设置为 0 弧度逆时针方向来测量（忽略 ANGDIR 的设置）。

如：

（setq　ang（getangle　“拾取点:”））

A.2　数据类型

数据类型是说明程序可以运算和处理的数据范围。AutoCAD 支持以下几种数据类型。

- 整型数：不含分数且小数点后的值必需是零的正整数或者负整数。通常使用 0 或 1 来表示某些系统变量的当前状态，通过设置 0 和 1 的值可以对这些变量加以控制。整形数值 AutoLISP 中以 16 位传输，取值范围为-32768 ～ +32767，其中符号“+”可以任选。
- 实型数：实型变量所使用的是一个含有分数且小数点后的值不为零点正数或者负数，整型数可以包含着实数内。对于小于 1 的实型数，小数点前必须用 0 作为开始。实型数可以用科学计数法表示；如用 9.01E+02 表示 901。
- 字符串：字符串是指用双撇号括起来的一串字符序列。Auto LISP 用它来做提示信息、命令选择项、调用命令名和尺寸文字等。

字符串中的字符个数为字符串长度。“”为空串，其长度为零，可以表示回车。“　”是一个空格字符，长度为 1。字符串常数的最大长度为 100 个字符。Auto LISP 字符串区分大小写形式，如“HELLO”不等于“hello”。

字符串中可用反斜杠“\”引入其后的控制字符。如，使用“\e”表示“ESC”。

- 表：由一个或多个任意类型的数据构成，用于 AutoLISP 的特殊变量。使用时将它们并排地放置在一划圆括号内，各数据间使用一个空格来分割。同一个变量可被 AutoLISP 多次定义使用，但是只有最后一次定义才是当前有效的。AutoLISP 变量名称可以与 AutoCAD 系统变量名称相同，它们之间是相互独立的。

> 注意：AutoLISP 变量名称不是所有的字符均能够设置，除了 AutoLISP 的保留字以外，一切产生歧义的字符也均不能使用。AutoLISP 的保留字是 AutoLISP 的专用字，如圆点（.）、双引号（"）、单引号（′），圆括号（()）、空格等，产生歧义的如控制符号（∧）等。

- 符号：符号是指以字符开头，其后有显示字符的字符序列。通常用它做变量名、函数名、命令名等。

用户定义的符号名，即变量名、用户函数名等必须以字母开头，中间不含空格，大小写形式

等效，且不能与函数和变量同名，也不要使用已定义的专用符号。

常见的专用符号见表 13-1 所示。

表 13-1　　数字区和相关参数的说明

控制符号	说明	控制符号	说明
（　）	表的定界符	“	字符串的定界符
;	注释标志符	‘	quote 函数的缩写
.	点对	!	求值符
T	逻辑常数“真”	nil	逻辑常数“假”，空或空表

A.3　函数与函数控制

本节主要介绍如何定义、调用 Auto LISP 函数。

A.3.1 定义 Auto LISP 函数

Defun 函数可以用来定义函数，其格式如下。

```
(defun 函数名（[函数参数] [/变量表……]）表达式……)
```

例如定义一个加 3 函数，程序源代码如下。

```
(defun mul 3 (x)
(setq x (+ 3 x))
)
```

A.3.2 调用 Auto LISP 函数

Auto LISP 以表的形式调用函数，其格式如下。

```
（函数名[函数参数]……）
```

这里的函数参数的数量可能为 0，也可以为任意多个，这取决于具体的函数。

每个参数还可以是表达式，返回的是表达式的最终计算结果。

每调用一个函数都会得到函数的结果，即函数的返回值。有的函数返回的是数值，有的函数返回的是逻辑常数 T 或者 nil，这取决于函数本身。如果不希望返回任何结果，则可以用无变元的 princ（打印字符）函数，比如（princ）。

调用自定义函数的方法与调用系统提供的函数的方法相同，例如：

```
(div 9 3 )
```

A.3.3 调用 AutoCAD 命令

Auto LISP 用 command 函数调用 AutoCAD 命令，其格式如下。

```
(command “AutoCAD 命令” “命令所需的数据”)
```

例如：

```
(command “line” “6,9” “18,27” “ ”)
```

即指绘制一条起点为（6，9），终点为（18，27）的直线。

> 注意：因为绘制直线的时候用空回车或者空格相应的“Specify next point or [Undo]:”提示才能够结束，所以在最后一个点的后面增加两个双引号 “” 以表示一个空格，这样 line 命令才能结束。

A.3.4 定义 AutoCAD 命令

定义 AutoCAD 命令用 defun 函数，格式如下。

```
（defun C: AutoCAD 命令名（/ 局部变数表）表达式……）
```

例 13-1　编写程序来绘制正弦函数

编写如下所示的程序。其中，使用 LP 代表曲线的左侧端点，N 表示正弦的周期。

```
(defun c: drawsin (/ lp lpx lpy N x step)
  (initget 1)
  (setq  lp (getpoint  "\n Left point:")
        lpx  (car lp) lpy (cadr lp))
  (initget 7)
(setq  N (getint "\Nnumber of cyles:"))
(setq  x 0 step 0.02)
(command "pline" )
(While (< x (* 2 N pi))
   (command (list (+ (car lp) x)
(+ (cadr lp)(sin x)))
   )
(setq x (+ x step))
)
(command "line" lp (list (+ lpx (* 2 N pi)) lpy) "")
(command "line" lp (list lpx (+ lpy 2))(list lpx (- lpy 2)) "")
(command)
)
```

运行程序如下：

命令：drawsin

Left Point:

Number of cycles:2

A.4　Visual LISP 环境

Visual LISP 是新一代 Auto LISP 语言，它运行在 AutoCAD 之外的一套窗口中。

启动 Visual LISP 有以下两种方法。

- 命令：VLISP
- 菜单命令：“工具”→“Auto LISP”→“Visual LISP 编辑器”

执行该命令后，弹出如图 13-1 所示“Visual LISP 为 AutoCAD”的程序编写窗口。

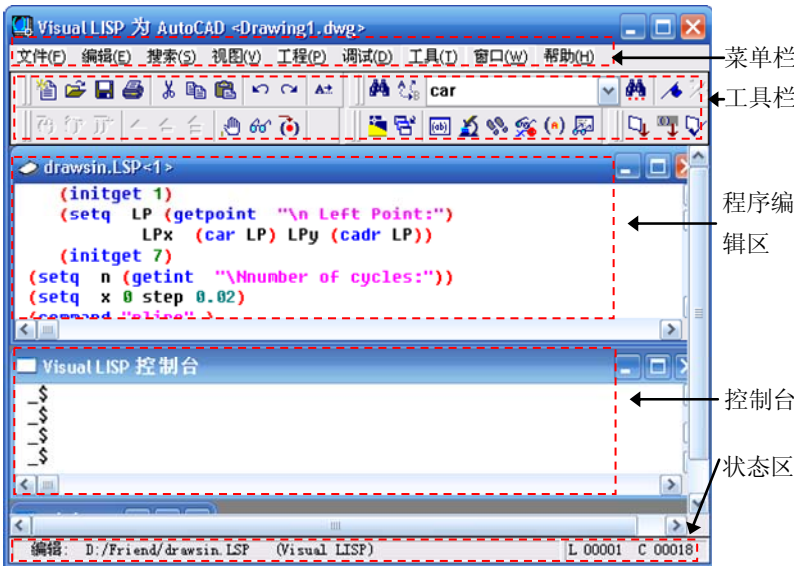

图 13-1

该环境窗口几个部分主要说明如下。

- 程序编辑区：Visual LISP 本身包括一个集成的文本编辑器，在此编辑 LISP、VLISP 和 DCL 等程序源代码文件。
- 控制台窗口：这是一个在主 Visual LISP 窗口接口上独立的窗口。在次窗口中，用户可以输入 Auto LISP 命令，这与在 AutoCAD 的命令行中输入的效果相同。

其他选项含义和普通 Windows 程序命令相似，不再详细说明。

A.4.1 菜单项

用户可以选择各种各样的菜单项来调用 Visual LISP 命令。菜单的内容随着启动窗口的不同而变化，单击一个窗口的标题栏，或者单击窗口的任意区域即可启动该窗口。如图 13-2 所示。

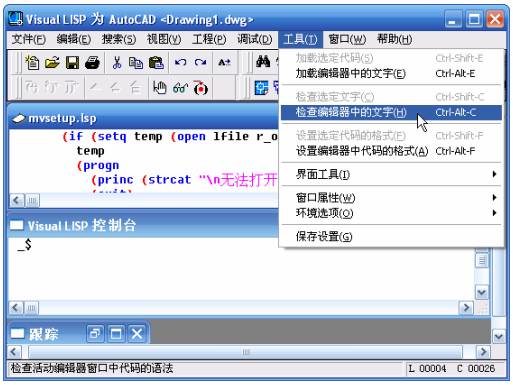

图 13-2

其它各菜单项的功能如下。

- 文件：管理应用程序文件，可以用来生成一个新的 Visual LISP 程序文件、打开和编辑已有的文件、保存程序文件代码、打印程序以及退出 Visual LISP 程序。
- 编辑：提供对程序代码的编辑命令，可以用力复制、粘贴文本恢复对程序所做的改动、在文本编辑器或控制台窗口中选择文本、在表达式中进行括号匹配以及重新显示在控制台窗口提示符下输入的下一个命令。
- 搜索：提供查询功能，可以用力查找和替换文本字符串、设置书签以及通过书签在程序中的应用。
- 视图：提供视图功能，包含一些在 Visual LISP 程序中查找和替换变量及符号值的命令
- 工程：提供应用程序工程功能，包含一些用来管理项目和编译程序的命令。
- 调试：提供程序调试功能，允许用户在程序中设置和删除断点，以及在程序运行时单步执行程序，还可以检查变量的状态和表达式的结果。
- 工具：提供设置 Visual LISP 选项文本格式、环境变量的工具等。
- 窗口：提供窗口管理功能，允许用户组织在当前 Visual LISP 进程中的窗口显示，以及启动 AutoCAD 窗口。
- 说明：提供 Visual LISP 帮助，包括在线帮助功能等。

A.4.2 常用工具栏

常用工具栏由“标准”、“搜索”、“工具”、“调试”和“视图”5 部分组成。虽然这些功能都可以在菜单中找到，但是利用它们将可以缩短调用命令的时间。此外还可以使用快捷键来帮助调用命令。例如在 Windows 下按<Ctrl+C>组合键可以进行复制，按<Ctrl+V>组合键可以进行粘贴，按<Ctrl+X> 组合键可以进行剪切。表 13-2 是一些在 Visual LISP 开发过程中经常用到的快捷键。

表 13-2　　编写 VLISP 时常用的快捷键

快 捷 键	操 作 环 境	作 用
Tab	控制台窗口	取出刚输入的前一条命令
Shift+Tab	控制台窗口	反转取出命令的方向
Esc	控制台窗口	清除控制台目前命令行中的文字
Shift+Esc	控制台窗口	保留目前命令行文字并打开一新命令行
Ctrl+[	程序编辑窗口	到对应的左括号
Ctrl+]	程序编辑窗口	到对应的右括号
Ctrl+E	程序编辑窗口	调出编辑需要的快捷菜单
Ctrl+Shift+Space	程序编辑窗口	使用“自动匹配”来寻找指定文字
Ctrl+Alt+E	程序编辑窗口	加载编辑窗口内的文字
Ctrl+Alt+F	程序编辑窗口	格式化编辑窗口内程序代码
F9	程序编辑窗口	在目前位置添加/删除断点
F8	调试时	跳到下一括号表达式内
Shift+F8	检测时	略过一括号表达式
F6	任意	显示控制台
Ctrl+W	任意	将变量添加到添加监视窗口
Ctrl+Shift+W	任意	显示添加监视窗口

A.4.3 程序编辑区

Visual LISP 的程序编辑区是其编程环境的核心元素，如图 13-3 所示。

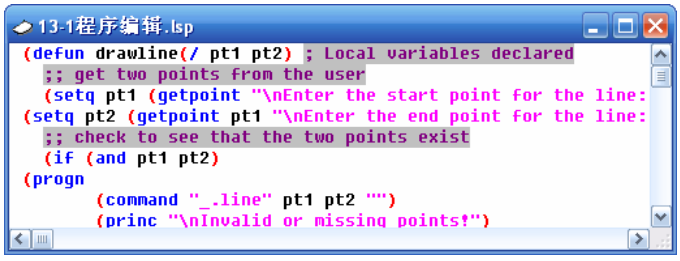

图 13-3

Visual LISP 编辑区的各种元素主要有以下功能。

- 彩色代码显示：Visual LISP 的程序编辑区为 Auto LISP 程序指定不同的颜色，便于用户很方便的查找程序元素和拼写错误。
- 执行 Auto LISP 表达式：可以帮助用户脱离程序编辑区，运行一个表达式或几行程序代码，以得到运行结果。
- 多文件查找：可以帮助用户在只运行一个命令的情况下，在多个文件中查找一个字货殖一个表达式。
- 语法检查：Visual LISP 的程序编辑器具有语法检查的功能。

A.4.4 控制台窗口

除了 Visual LISP 主操作窗口中的下拉式菜单、工具条与快捷键以外，在控制台窗口中输入命令也一样可以发出 Visual LISP 命令。

Visual LISP 的控制台是一个类似 AutoCAD 程序编辑窗口区的工具。在 VLISP 程序的开发过程中，有效地使用控制台窗口可以提高开发的效率。在控制台窗口中可以直接计算输入的表达式。如果希望有一个临时性的变量，那么也可以在这里直接将变量值纳入目前的程序文本中，当然这个变量是暂时性的，当程序文本关闭时也就结束了。此外，还可以储存控制台窗口中所输入的文本与文本执行后的输出结果。

例如可以将程序源代码分段拷贝到控制台中执行，以查看它的结果，这样程序的调试效率会比较高。

虽然 VLISP 控制台窗口和 AutoCAD 程序编辑窗口区所提供的功能相近，但是完成同一功能的操作有些不同。例如为了显示 Auto LISP 变量的目前值，在 VLISP 控制台窗口中，只需输入变量名称并按下<Enter>键即可；但是在 AutoCAD 程序编辑窗口区中，则必须在变量名称前加一个感叹符号（!）。

在 AutoCAD 程序编辑窗口区中，按空格键将对表达式求值。而 VLISP 控制台窗口只有在按下<Enter> 键后，才处理在控制台提示处所输入的表达式。

在换行处按<Ctrl+Enter>组合键可以在下一行继续输入程序文本的表达式。

在按<Enter>键之前可以输入多个表达式，如：_$（setq a 1）（setq b 2）。

用 Tab 键或者<Shift+Tab>组合键可以重复前一条命令。

在控制台窗口右击，或者按<Shift+F10>组合键，会显示一个如图 13-4 所示的关联菜单。

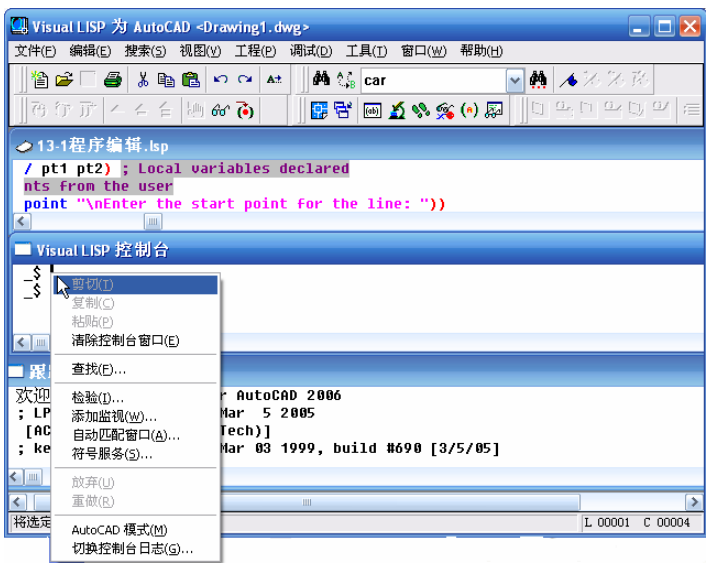

图 13-4

这个菜单包含部分 Visual LISP 命令，用户可以利用这个它在控制台窗口的命令行上进行复制文本、检验程序、切换 AutoCAD 模式等操作。

A.5 撰写 Auto LISP 程序

Auto LISP 允许用户把每一条 Auto LISP 语句有机地组合起来，以文件形式来执行其功能。按以下步骤可以进行整个 Auto LISP 程序的编写与执行。

操作步骤

（1）启动 AutoCAD 2007，选择“工具”→“Auto LISP”→“Visual LISP 编辑器”选项。

（2）在“程序编辑窗口”中输入以下文本（注意：以下每条程序前的数字，如（1）、（2）等是为了方便稍后的语法说明，非程序正文文本）。

```
(1);;;Hello World Auto LISP Program-------firstlisp.lsp
(2);;;function:draw a circle and print "Hello World"
(3)
(4)(defun c:firstlisp(/ c Enter pt radius)
(5)(setvar "cmdecho" 0)
(6)(setq c<Enter>pt (list 100 100 0 ))
(7)(setq radius 50)
(8)(command "circle" c Enter pt radius)+
(9)(prompt "Hello World! ")
(10)(princ)
(11) )
```

分析：程序的第（1）、（2）行是批注，以分号“;”开头的就表示此行为批注。批注行是不执行的，它是用来提醒或说明该程序的设计重点或设计内容的。第（4）行是命令的定义语法，后面括号内斜线后的变量名称表示该变量是暂时性的变量。第（5）行用来设定“cmdecho”系统变量为 0，“cmdecho”系统变量将用来控制当 Auto LISP 的（Command）函数执行时，AutoCAD 是否显示提示与输入。

第（6）、（7）行的语法表示要将圆的圆心与半径的值分别分派给变数 c Enter pt 与 radius。第（8）行则是调用 AutoCAD 的 circle 命令，并根据 c Enter pt 与 radius 值来执行画圆动作。第（9）行则表示要在程序编辑窗口区中打印出“Hello World!”字样。第（10）行是第（9）行的动作执行命令。第（11）行则是第（4）行的对称括号。

例如在 AutoCAD 中使用 LISP 命令，要从点“0,0”到“100,100”画出一条直线，其命令流程如下。

```
命令: line <Enter>
指定第一点: 0,0 <Enter>
指定下一点或 [复原(U)]: 100,100 <Enter>
```

```
指定下一点或[复原(U)]: <Enter>
```

如果要在 AUTO LISP 中使用 Command 函数来执行一样的动作，则为：

```
(command"line""0,0""100,100""")
```

其中"line"对应 line <Enter>

"0,0"对应 0,0 <Enter>

空双引号 "" 代表<Enter>

读者应理解并掌握 Command 函数的用法。

A.6 加载和调试 Auto LISP 程序

撰写好程序后，就需要加载程序调用相应的命令来执行编写的程序。

A.6.1 加载 Auto LISP 程序

存储在磁盘上的 Auto LISP 程序，需要在 AutoCAD 状态下加载后才能运行。一般有以下加载方法。

1. 使用命令行调用 Load 函数

在 AutoCAD 命令行输入 Load 命令加载.lsp 文件。格式如下：

```
命令: (load " 文件名 ")
```

被加载的程序文件没有错误，则返回最后一个用户函数名，表示该文件已被加载，否则显示错误信息，需调试完成后再重新加载。其中“文件名”包括文件的磁盘位置和文件名称。

2. 利用 acad.lsp 文件自动加载

AutoCAD 里允许有一个 acad.lsp 文件。每当系统启动 AutoCAD 系统时，将自动查找是否存在 acad.lsp 文件，如果找到该文件，将自动加载。

3. 使用对话框方式加载文件

输入 Appload 命令或者选择“工具”→“AutoLISP”→“加载应用程序”命令，打开“加载/卸载应用程序”对话框，选择需要加载的.lsp 文件，然后单击 加载(L) 按钮即可加载该程序，如图 13-5 所示。

图 13-5

A.6.2 调试 Auto LISP 程序

与大多数程序开发工具一样，Visual LISP 提供了强大的程序调试功能。一般情况下，初次编写的程序都不能按照预期的方式运行，得到的结果也往往是错误的。这就需要用户进行程序的调试和修改工作。在 Visual LISP 中提供了许多调试功能，以帮助用户检查程序中的错误。一般而言，程序出错有以下两种情况。

（1）语法错误：没有按程序规定的语言写程序，这是低级错误。

（2）逻辑错误：程序员错误地理解了计算机所要完成的任务，这是高级错误，对于一个程序员来说要尽量避免。无意输入错误变量名对计算机而言也属于这种错误。

一个正确的程序不能有上述两种错误。现在的高级程序语言的集成编译环境一般都提供语法错误的检查，但是并不能告诉程序中是否存在着逻辑错误。Auto LISP 的集成编译环境 VLIDE 提供了一系列调试手段，以供检查程序中的逻辑错误。通常出错的程序是上述两种错误的组合，如表 13-3 所示。

表 13-3　　发生错误的组合

情　况	语法错误	逻辑错误
1	有	无
2	无	有
3	有	有

下面用实例来介绍如何调试 Auto LISP 程序。

正确的范例程序为：

```
1 (defun c:firstlisp (/ c<Enter>pt radius)
```

```
2 (setvar "cmdecho" 0)
3 (setq c<Enter>pt (list 109 50 0))
4 (setq radius 50)
5 (command "circle" c<Enter>pt radius)
6 (prompt "Hello World!")
7 (princ)
8 )
```

功能：以（109,50）为圆心画一个半径为 50 的圆，并在命令行打印“Hello World!”字样。

范例 1：

```
1 (defun c:firstlisp ( / c<Enter>pt radius)
2 (setvar "cmdecho"0)
3 (setq c<Enter>pt (list 109 50 0)))
4 (setq radius 50)
5 (command "circle"c<Enter>pt radius)
6 (prompt "Hello World!")
7 (princ)
8 )
```

错误之处：程序的第 3 行最后多了一个“)”反括号。

写完程序按下加载按钮，程序会提示：

```
; 错误：输入中含有多余的闭括号
```

根据错误提示，知道是多了一个右括号。遇到这种情况在这个简单例子中排除并不困难。

方法一：将光标放在第一行左括号的左方，双击检查与之匹配的右括号（见图 13-6，匹配区域内的程序会反白显示）。也可以将光标置于右括号的右方，双击检查与之匹配的左括号。

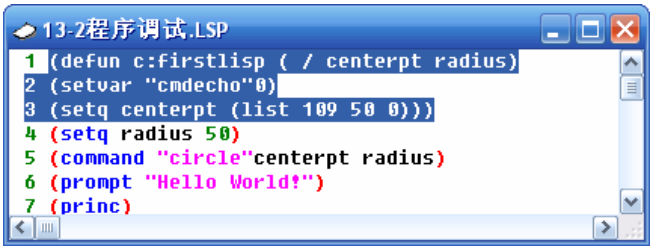

图 13-6

> 技巧：在大程序中可能一次出现多个语法错误，先看一下这些错误之间是否有关联，全部修改完再加载，这样可以节省时间。

将刚才的程序稍微改动，程序错误如下：

```
1 (defun c:firstlisp (/ c<Enter>pt radius)
2 (setvar "cmdecho" 0)
3 (setq c<Enter>pt 100 (list 109 50 0))
4 (setq radius 45)
5 (command "circle" c<Enter>pt radius)
6 (prompt "Hello World!")
```


```
7 (princ)
8 )
```

由该程序画出的圆的半径为 45，因此这个程序能够正常地执行，但是结果是错误的。当知道是半径出错之后，直接将程序中的 45 改回 50 即可。

下面演示应用 VLIDE 提供的检测手段来检查错误，假设不知道程序变量的含义，步骤如下。

操作步骤

（1）将光标放在绘制圆的程序前，为程序设置断点，按下 F9 键，如图 13-7 所示。

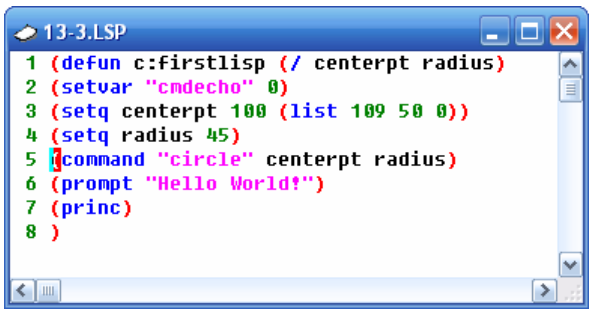

图 13-7

（2）单击加载按钮来加载程序执行。

（3）程序会停止在第 5 行之前，并在控制台窗口提示：

```
; 错误: SETQ 中参数太少: (SETQ C<ENTER>PT 100 (LIST 109 50 0))
```

（4）程序在画圆之前要先检查一下圆的半径，其中语句(command "circle" c Enter pt radius)的最后一个变量 radius 代表半径，双击 radius，然后右键单击，在快捷菜单中选择“添加监视”菜单项，如图 13-8 所示。

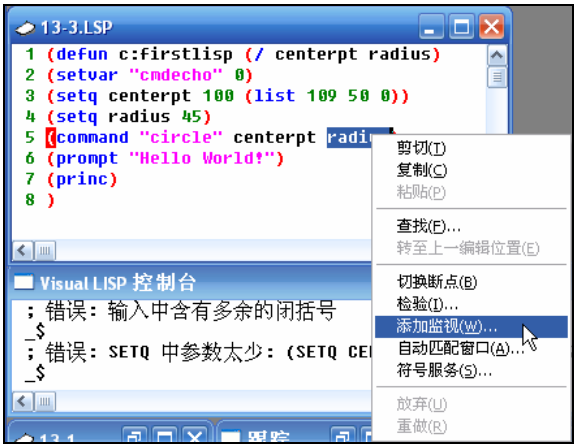

图 13-8

在如图 13-9 所示的监视窗口中可以看到 radius 的值为 45，这与需要的 50 不同，从而知道是对 radius 的赋值出错了。

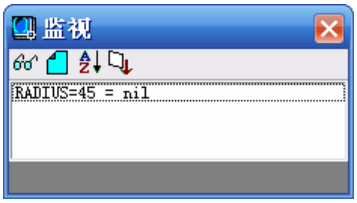

图 13-9

（5）按下 按钮跳出这一次的检测。

（6）再根据经验在程序中可能影响 radius 赋值处设下断点。因为此程序较简单，因此我们将断点设在第 4 行。如果是复杂的程序不容易看出什么地方影响 radius 的赋值则可以将断点前移至较长的语句地方，以保证影响 radius 赋值的代码被包含在其中，如图 13-10 所示。

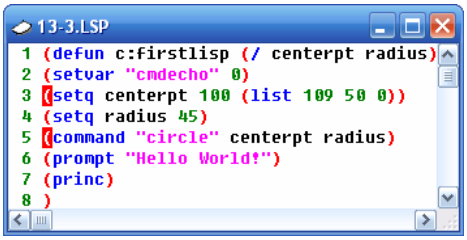

图 13-10

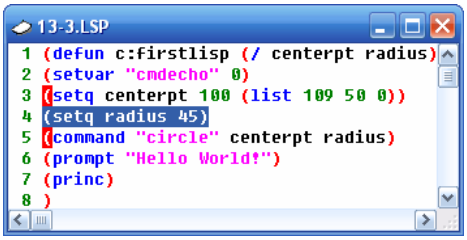

图 13-11

（7）加载并执行程序，程序在第 4 句处停止，如图 13-11 所示。

（8）如上将 radius 加入监视窗口中，radius 的值为 nil，按 F8 键单步执行第 4 行语句，监视窗口

中 radius 的值发生了变化，如图 13-12 所示。

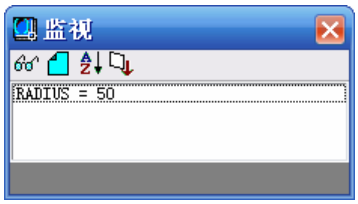

图 13-12

此时可知是第 4 行语句赋值出错，因此应该修改此句。

查找此类错误的需要以下步骤。

- 先发现错误结果的直接起因代码。
- 将错误量化为数值，查找错误的变量赋值。如本例中为圆半径出错，则应找出代表圆半径的变量 radius。
- 找出影响变量赋值的语句进行跟踪。

❖ 复杂的程序中影响变量赋值的语句可能在许多子程序中，这将大大提高查错的难度。不过查错的原理和基本方法（设置断点和跟踪等）不会变。

除了 F8 键的单步跟踪外，“调试”菜单中还有几种执行代码的方法，灵活地运用可以提高复杂程序的调试速度，因为在大程序中我们不可能单步跟踪第一句程序。

A.7 编译 Auto LISP 程序

程序的编译首先要设定文件搜索。只有设定了文件搜索位置，AutoCAD 才能找到 Auto LISP 文件范例、图块文件范例或菜单文件。可以按照下述步骤来执行这些文件搜索路径的设定。

输入 CONFIG 命令后按<Enter>键，出现图 13-13 所示的“选项”对话框。

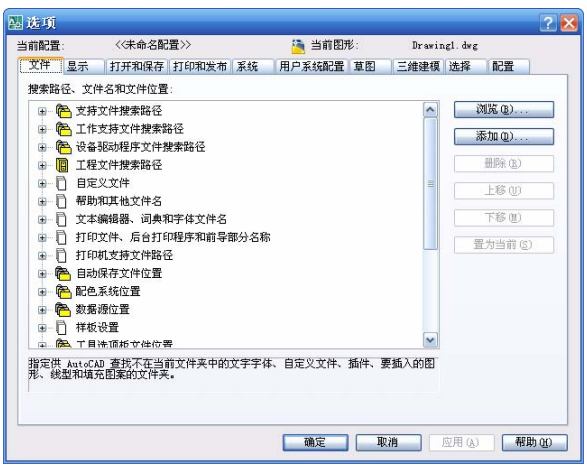

图 13-13

现在要设定的是搜索路径。应单击 支持文件搜索路径 名称前的“＋”符号，开始窗口里只有几个预设的 AutoCAD 路径。要添加新的路径为 E:\AutoCAD2007\Samples\CH13。

操作步骤为：单击窗口右边的 添加(D)... 按钮，然后直接输入路径或者单击“浏览（B）”按钮来选取路径即可，如图 13-14 所示。

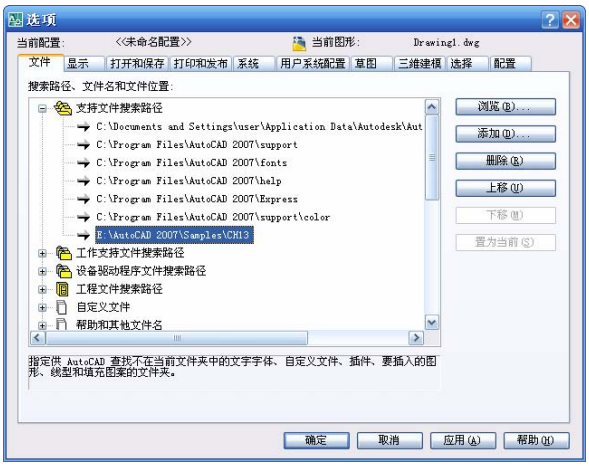

图 13-14

设定完文件的搜索路径，即可对程序进行编译。AutoCAD 的编译功能可以将一个文本文件的 Visual LISP 源文件.lsp，编译成一个文件扩展名为.fas 的二进制编译文件。这样的.fas 的编译文件只能被执行，而不能修改其内容。

可以按照以下的步骤来进行编译 Visual LISP 源文件的操作。

操作步骤

（1）选择“工具”→“Auto LISP”→“Visual LISP 编辑器”菜单命令，将要编译且执行已无问题的 Visual LISP 源文件加载。

（2）选择“工程”→“新建工程”菜单命令，按照如图 13-15 所示的步骤操作。

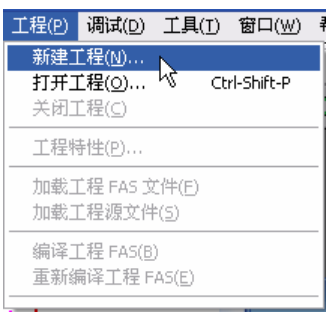

图 13-15

（3）弹出“新建工程”对话框，选择要建立工程的目录，如图 13-16 所示。

（4）输入工程名称。

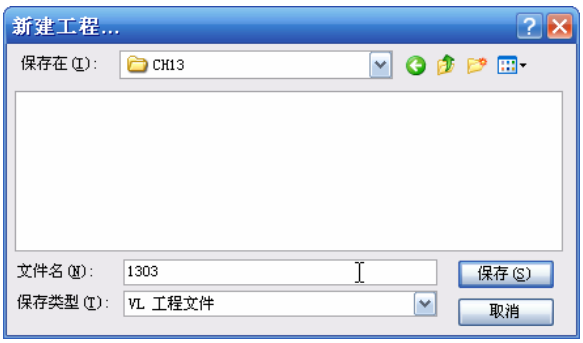

图 13-16

（5）然后单击 保存(S) 按钮。

（6）弹出“工程特性”对话框，选择要编译的源程序文件名，如图 13-17 所示。

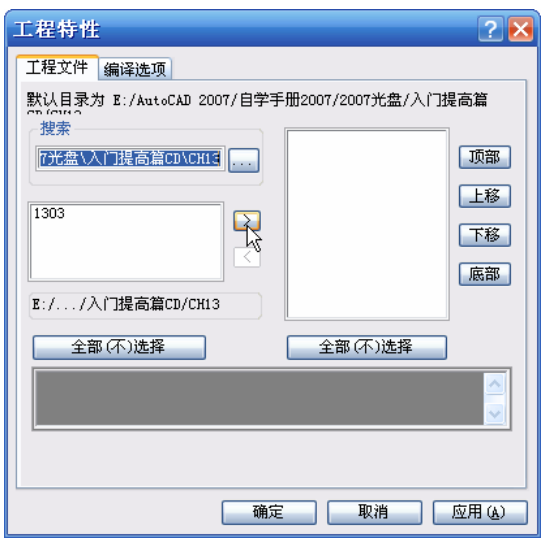

图 13-17

（7）单击 按钮，随即在右边的框中选择“1303”字样。

（8）单击 确定 按钮，弹出以“1303”命名的窗口（见图 13-18）。

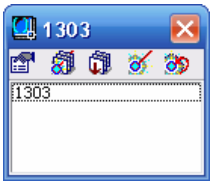

图 13-18

（9）单击（工程特性）按钮返回到“工程特性”对话框。单击 编译选项 选项卡中“编译模式”选项区中的 优化(T) 选项。然后单击“FAS 目录”框中的 ...（浏览目录）按钮来指定要存放编译文件的目录，如图 13-19 所示。

图 13-19

（11）然后单击 确定 按钮，在 1303 窗口中单击（编辑工程 FAS）按钮，系统开始编译，如图 13-20 所示。

图 13-20

（12） 编译完成后，到刚刚指定的目录下，就可以看到编译后的结果。此外，还可以将.LSP 文件另存起来，而仅留下.FAS 文件来加载执行即可。

A.8 专家技能点拨：Auto LISP 基本语法结构

本章介绍了 Auto LISP 语言的基础知识，包括函数、函数控制以及数据类型，并讲解了如何

用 AutoCAD LISP 编写、调试以及运行程序。下面对 Auto LISP 中重要的语法结构作以介绍。

1．以括号组成表达式

左右括号数一定要成双成对、相对称。请用户依次在“命令:”列 （或命令:）后面直接输入以下表达式。

命令: (+ 1 2 3 5 8) 传回加总和 19

命令: (max 20.5 18.9 45) 传回最大值 45

命令: (menucmd "p1=*") 显示拉下 POP1

命令: (getint "<Enter> Real:") 传回<Enter> Real:要求输入一个实数

命令: (setq a (* 14.5 6)) 传回相乘值 87，同时将值设定给变量 a

命令: !a 回应 87 (注意：！号可用来查询变数值)

错误的范例：

命令:(setq a (* 12.5 4)

1> 传回“1>”表示少了一个右括号，此时只要再补上一个) 即可。

2．表达式类型

(表达式 运算函数 运算函数 运算函数 ...)或 (函数(式) 自变量 自变量 自变量 ...)，或 (函数(式) 元素 元素 元素 ...)

运算函数包括“功能函数”和“自定义函数式”，功能函数包括如+、max、menucmd、getint、setq、*、min 等；自定义函数式包括由用户自定义的新函数式或子程序。

自变量或元素包括：

A．整数(Integer)：如 10，-18，300 等，范围在-2^{16}～2^{16}（-32768～32767）之间，若以 float 函数将其转为实数，则范围为-2^{32}～2^{32}（-2147483648～2147483647）之间。

B．实数(Real)：如 8.5，-17.456 等，为带有小数点的数，精度可达 14 位有效数字。

C．字符串(String)：如 "AutoCAD"，"123"和"<Enter> Real:"等，是以双引号内字符为认定，字符数不受限制。

D．串行(List)：如 ("a" "b" "c")，(x y)和("a" 8 3.5)等，是以括号内元素为认定，元素类型比较随意，在 AutoLISP 程序设计中应用频率非常高

E．像素名称代码：如 <entity name: 6000f262>，AutoCAD 会自动赋予像素指向代码,透过此代码可找到像素的数据库记录

F．档案代码：如<file:#12438>，打开文件作读文件、写文件时所产生的代码。如：

(setq ffr (open "test.txt" "r")) 回应 <file:#24138>

(setq ffw (open "abc.txt" "w")) 回应 <file:#34812>

G．选择群集代码：如<Selection set:1>、<Selection set:1>，是一个或数个像素所形成的选择

集

3．表达式中的运算函数

可以是另一表达式或子程序。功能函数使用语法：

(strcat 字符串 1 字符串 2 字符串 3 ....) 字符串结合功能函数

(getstring 提示) 要求输入一个字符串

(rtos 实数) 将实数转换成字符串

如 1：(strcat "abc" "123" "LISP") 传回："abc123LISP"

2：(strcat "abc" (getstring "<Enter> String:") (rtos 123.45))

执行结果：<Enter> String：输入任一字符串，如输入 *HELLO*

回应 "abc*HELLO*123.45"

4．多重的括号表达式

运算的先后顺序是由内而外、由左而右

如将 10.25 + 17 - A.2 / 7 的结果，设定给变量 kk，并转换成 AutoLISP 的表达式

解答技巧：

（1）运用二分法、对函数加括号：(10.25 + 17) - (A.2 / 7)

（2）将函数往前提： kk = (- (10.25 + 17) (A.2 / 7))

（3）将函数再往前提：kk = (- (+ 10.25 17) (/ A.2 7))

（4）设置完成： (setq kk (- (+ 10.25 17) (/ A.2 7)))

注意：以文件存在的 AutoLISP 程序（ASCII 类型），其扩展名必须是.LSP。

5．以 defun 功能函数定义的命令或功能函数式

程序结构：

(defun 函式名称 (自变量/区域变量)

……

程序内容

……

)

技巧：撰写的环境，只要是一般的文本编辑软件，可编辑 ASCII 类型都可以使用。如 PE2、PE3、DW3、汉书、EDIT、记事本等。

6．C:函式名

新定义的功能函数式名称若为“C:函式名”，则此函数式可作为 AutoCAD 新的命令。其中，自变量及区域变量可省略

程序结构

(defun C:KK(自变量/区域变量)

……

程序内容

……

)

则加载此 LISP 程序后,可在命令后直接输入新定义的 KK 命令

7．加载 AutoLISP 程序的做法

在命令后直接输入 (load "LISP 文件名")

其中，load 与字符串以及双引号间的空格，可有可无 (load"LISP 主文件名")

如在 c:\lsptools 目录下有一 LISP 程序 tt.lsp，而目前的工作目录在 c:\dwg 下。若“选项”中“支持文件搜寻路径”指定到 c:\lsptools 目录，则在使用加载 tt.lsp 命令后输入(load "tt") 即可。如果“选项”中“支持文件搜寻路径”没有指定到 c:\lsptools 目录，则输入 (load"tt") 后会出现如下错误信息。

无法开启「tt.lsp」做输入错误: 载入失败(LOAD "tt")*取消*

此时可以在命令行输入(load "c:\\lsptools\\tt") 或(load "c:"/lsptools/tt)，而不能输入(load "c:\lsptools\tt")，因为“\”在 AutoLISP 结构语法中已经用于其他用途了。

8．批注的使用

在 AutoLISP 程序中，在分号后的内容均为批注，程序不作处理，适时的增加批注，将使程序更具可看性和完整性。如 pp.lsp 内容如下：

; 本程序功能可快速绘制螺钉、机械螺旋

; 设计者：刘 伟

; 版权所有，欢迎广大用户传播使用

(defun c:pp()

……

; 以下为绘制螺钉子程序，

……

; 以下为绘制机械螺旋子程序，

……

)

9．查询变量名

如果需要在 AutoCAD 的环境中，查看一变量值，在命令行输入“!变量名”即可。

如：输入(setq aa (+ 100 75)) 传回：175

命令:!aa 传回：175

10．区域变量和整体变量

以(defun C:函式名(自变量 / 变数) .....)程序中，程序中的变量若在“/”右边变量内，则称为“区域变量”，否则为“整体变量”，其中在 AutoLISP 中，未赋予值的变量,其值皆响应 nil。

“区域变量”即在该程序执行完毕后，其值自动消失；“整体变量”即在该程序执行完毕后，其值仍然存在。如 test.lsp 内容如下。

```
(defun c:tt(/ sa sb sc)
(setq sa 100)
(setq sb 20)
(setq sc 10)
(setq sd (+ sa sb sc))
)
```

命令: (load"tt") 传回：c:tt

命令:tt 传回：130

命令:!sa 传回：nil (属区域变量)

命令:!sb 传回：nil (属区域变量)

命令:!sc 传回：nil (属区域变量)

命令:!sd 传回：130 (属整体变量)