主编 舒 飞

本书内容

- Auto LISP 核心代码
- 程序化绘图过程与 Visual LISP
- DCL 代码/菜单代码
- AutoLISP 程序实例
- 应用 Auto LISP 编程实例
- 简单的 DCL 实例
- 复杂的 DCL 实例
- 编程设计三维造型
- 简单的菜单设计
- 图像菜单设计

中文版

AutoCAD 2004

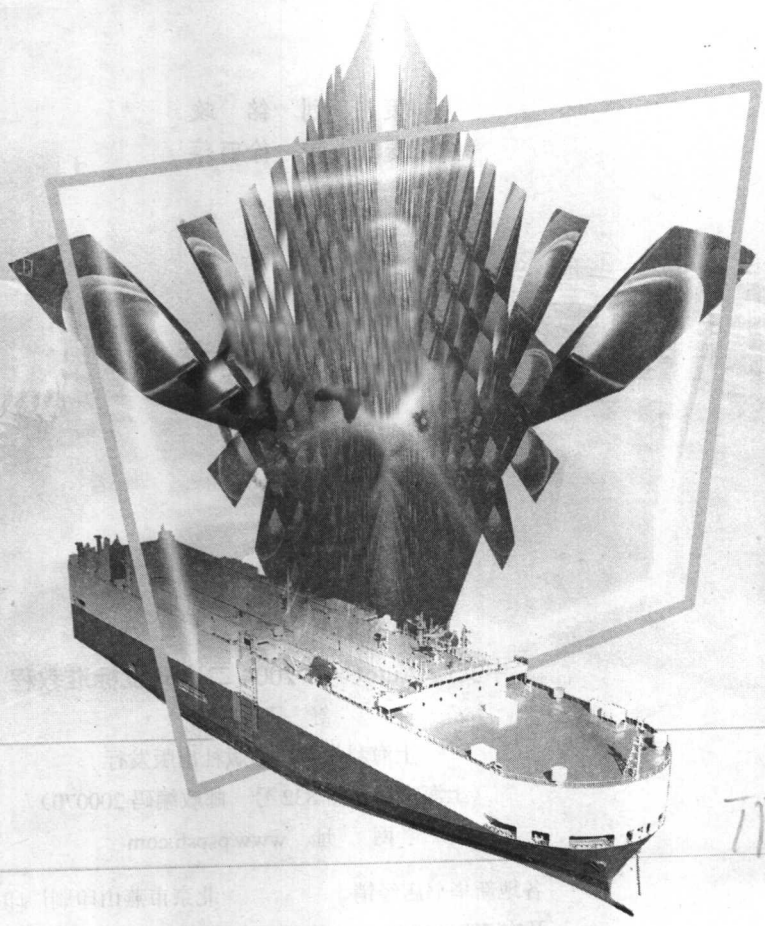
二次开发标准教程

Auto CAD 2004 Er ci kai fa Biao zhun jiao cheng

上海科学普及出版社

中文版
AutoCAD 2004
二次开发标准教程

主 编 舒 飞



Auto CAD 2004
Er Ci Kai Fa Biao Zhun Jiao Cheng

上海科学普及出版社

图书在版编目（CIP）数据

中文版 AutoCAD 2004 二次开发标准教程 / 舒飞主编.
上海：上海科学普及出版社，2004.3
ISBN 7-5427-2732-X

Ⅰ.中… Ⅱ.舒… Ⅲ.计算机辅助设计—应用软件，AutoCAD 2004—教材 Ⅳ.TP319.72

中国版本图书馆 CIP 数据核字（2004）第 001312 号

策　划 铭　政
责任编辑 徐丽萍

中文版 AutoCAD 2004 二次开发标准教程
舒　飞　编著
上海科学普及出版社出版发行
（上海中山北路 832 号　邮政编码 200070）
网　址　www.pspsh.com

各地新华书店经销　　北京市燕山印刷厂印刷
开本 787×1092　1/16　　印张 21　　字数 563000
2004 年 3 月第 1 版　　2004 年 3 月第 1 次印刷

ISBN 7-5427-2732-X / TP·546　　定价：28.00 元

内 容 提 要

长期以来，广大工程人员一直热切期盼有一本易学易用的 AutoLISP 程序开发图书，以便学习、使用 AutoLISP 程序和代码，解决绘图设计问题，更希望能掌握一门实用的程序编制技术，以转行到软件行业中。鉴于这种情况，同时结合广大工程人员的需要和建议，我们精心编写了本书。

与 AutoCAD 2000 和 AutoCAD 2002 相比，中文版 AutoCAD 2004 拥有更为强大的性能，这在进行 AutoLISP 程序开发的时候特别明显。在中文版 AutoCAD 2004 中，即使运行 10 万个图元也游刃有余，因此在中文版 AutoCAD 2004 中进行 AutoCAD 二次开发非常方便。

本书以大量实用、精炼的程序实例为主，对工程设计人员、软件编制人员，尤其是希望从工程行业过渡到软件行业的人员很有参考价值。本书在编排上遵循循序渐进的原则，既有简单的平面、曲面程序，又有三维造型编程、商业编程。全书实例丰富，讲解清楚，循序渐进，非常适合各种计算机辅助设计工作人员和电脑爱好者使用，是一本集知识性和适用性于一体的最佳参考书。

前　言

在连续出版多本计算机辅助设计的图书之后，笔者收到众多计算机辅助设计爱好者的热情信函，在给予笔者着力推广计算机辅助设计技术的工作以热情的支持之外，也提出很多相当有分量的技术问题，例如，螺纹在机械行业的用途十分广泛，而 AutoCAD 软件系列居然没有给用户提供绘制各种螺纹的功能，不能不说是一大遗憾。何况今日的制造业，正在大量使用各种曲线、曲面造型来完成各种各样的特殊功能，不再局限于螺纹这一种了。

AutoCAD 真的不能绘制复杂曲线、曲面、三维造型吗？非也！AutoCAD 之所以不提供这些功能，不是技术上不许可，而是受到两个限制。首先，Autodesk 认为 AutoCAD 是一个提供基本功能和可以进行再次开发的软件平台，它应该适用于包括机械在内的各个行业，诸如建筑、土木、服装、电子、装修等。如果各个行业需要什么特殊功能，由使用者自行开发即可；其次，AutoCAD 软件只是运行在计算机上的一个应用软件而已，运行速度不快，占用计算机资源太多，执行绘制复杂图形的任务效率不高。因此，AutoCAD 不主动提供这些功能，但是提供了开发这些功能的工具。

本书首先讲述如何进行 AutoCAD 二次开发，如何绘制各种各样的曲线、曲面，以满足广大工程设计人员的要求；其次，对那些希望进入软件行业的读者，本书也讲述了若干商业 AutoLISP 程序的开发过程，读者可以仔细研究商业软件和自编自用程序之间的区别，多掌握一门新技艺，对择业是大有好处的。

不同于讲述 AutoCAD 二次开发的其他图书的是，本书着重于那些 AutoCAD 不提供的、非要编程不可的功能，例如，本书决不讲述“用户定制”这种内容，既然 AutoCAD 已经提供了这种功能，何必再讲述呢？而诸如自由曲面、图像菜单等才是本书重点。而且，本书使用大量实例来讲述二次开发所需的知识点，让读者在实际操作中掌握所需知识，而没有专门进行理论讲解。

本书所使用的例子力图短小精悍，非常有利于读者学习参考。而一般编写软件要求的容错性、健壮性等要求，就不在代码中反映了。读者可在实践中积累经验，自行修改这些例子来增强这些能力吧。

本书有的例子运行起来有小小的问题，例如，切割螺纹的程序编写正确，也可以运行，可就是走刀不正确，所切割出来的螺纹不对，这一般是因为启动 AutoCAD 时，系统自动启动了捕捉、正交等功能，执行螺纹切割任务的时候没能按照程序计算出来的点移动刀具，导致程序失败。因此，学习本书时，应遵守以下约定：

- 最好在中文版 AutoCAD 2004 中编写、运行程序代码，其他 AutoCAD 软件也可以，但是它们的稳定性大不如中文版 AutoCAD 2004。
- 应该使用赛扬 1.3G、内存 128MB 以上配置的电脑编制、调试和运行程序代码。
- 编写、运行程序代码之前，必须关闭特征点捕捉、正交等自动功能。
- 如未加特别说明，“单击”是指按一下鼠标左键，“双击”是指连续按两下鼠标左键。
- 凡是绘制平面图，默认的初始绘图平面都是指 X 轴向右、Y 轴向上的 XY 平面。

✸ 凡是绘制三维造型，默认的初始视角均是西南等轴测视角，使用 X 轴向右、Y 轴向上、Z 轴垂直 XY 平面向上的通用坐标系。

本书由舒飞主编，同时参与本书编写和审校的还有崔慧勇、任立功、吴闯、董金波、芦淑珍、李建慧、杨志丽等。由于作者水平所限，书中不足与疏漏之处在所难免，恳请广大读者不吝指正，以便再版时进行改进。联系网址：http：//www.china-ebooks.com。

编 者

2004 年 1 月

目　录

目录

第 1 章 AutoLISP 核心代码

凡是语言，都要拥有两个基本方面：表达单个意义的词语和把词语组织起来的语法，AutoLISP 也不例外。如果要正确使用这种语言，读者应该先学会一些常用的词语和语法，就像学习英语、日语等其他的人类语言一样，可以把 AutoLISP 语言的词语和语法看作为 AutoLISP 语言使用的数据类型，它包括整数、实数、字符串、文件指针、图元名、表、AutoCAD 的选择集、函数和图元群码等类型。学完本章，读者即可编写出若干正确的 AutoLISP 代码组，执行这些代码，可以实现简单的功能。

1.1 数 据 类

AutoLISP 语言使用 7 种类型的数据，它们是整数、实数、字符串、文件指针、图元名、表和 AutoCAD 的选择集，下面将一一介绍它们。

1.1.1 整数

AutoLISP 语言中使用的整数包括日常生活中理解的那种整数，即不带小数点的阿拉伯数字，如 0、1、2、3、…、9 等，不过，也包括如+、-等字符，如果是正整数，可以省略前面的+号，所以这种整数是带符号的。AutoLISP 以 32 位数据保存这种整数，它有最大、最小的范围，即在-2 147 483 648~+2 147 483 647 之间，超过这个范围的整数不能被 AutoLISP 识别。AutoLISP 和 AutoCAD 之间进行数据传递时，所传递的整数只有 16 位，即在-32 768~+32 767 之间，超过这个范围的整数无法传递给 AutoCAD。

如果在 AutoLISP 表达式中直接使用整数，这时它被称为常量。下面是合法的 AutoLISP 整数：

```
3、-4、38900
```

1.1.2 实数

实数就是带小数点的数。AutoLISP 以双精度浮点格式保存实数，精度不低于 14 位有效位数，但是在 AutoCAD 命令行窗口中只显示 8 位有效数字。

实数可以用科学计数法表示，科学计数法的格式中可包括 e 或 E 及指数，例如，按科学计数法表达的实数 3.4e-6 和 0.0000034 是一样的。

如果在 AutoLISP 表达式中直接使用实数，这时它被称为常量。下面是合法的 AutoLISP 实数：

```
2.1、-4.6、7800000.1
```

1.1.3 字符串

字符串也可以被 AutoLISP 语言当作一种数据使用，但是必须用双引号引起来。引号内的内容可以传递给 AutoCAD 当作指令执行，所以在引号包括的字符串中，可以用反斜杠（\）添加控制字符。字符串在 AutoLISP 程序中常用于文件名、标志符及 DCL 中的控件名。

如果在 AutoLISP 表达式中直接使用引号包括的字符串，该字符串的值被称为字符串常量。

字符串中除了分隔符以外的字符的个数称为字符串长度。这个长度可以是任意的，系统通过动态分配存储空间来满足其长度需求。不过，字符串常量的最大长度为 132 个字符，如果超过这个长度，则后面的字符无效。

字符串中可以包括 ASCII 中的任何字符，通常格式为“\nnn”，其中 nnn 是其字符的八进制 ASCII 码，如字符串“5AB7”也可表示为“\065\101\102\067”。

因为系统预先把“\”当作标志字符，所以如果字符串需要包含它，则必须使用“\\”或“\134”来代替“\”。双引号（"）被系统用于字符串定界，所以如果字符串需要包含它时，可以使用“\042”来表示。

AutoLISP 还提供了一些特定的控制字符的简单表示形式：

```
\r      意义是回车 CR
\e      意义是取消 Esc
\t      意义是制表 HT
\n      意义是换行 LF
```

1.1.4 文件指针

当 AutoLISP 函数需要读写文件时，必须引用该文件的标签。文件指针，或者称为文件描述符号，就是分配给该文件的标签，例如，现在要以只读方式打开一个文件“零件统计.doc”：

```
_$(setq file1(open "c:\\ 零件统计.doc" "r"))
#<file"c:\\ 零件统计.doc" >;返回值
```

文件指针保存在变量 file1 中。

除非用户使用 close 函数关闭该文件，例如：

```
_$(close file1)
nil;返回值为空
```

否则该文件将一直保持打开状态。

1.1.5 图元名

AutoCAD 所绘制的图形中，每个图形都有自己的标签，它就是图元名，例如，现在绘制一个圆柱体，可以通过函数 entlast 来获取它的图元名：

```
_$(entlast)
<图元名: 40079e28>
```

图元名其实是一个指向该图形的指针。如果该图形在屏幕上，通过该指针，AutoLISP 可以从图形数据库中找到该对象的数据记录和矢量。图元名可以被 AutoLISP 函数调用，这样

在处理该图形时提供了多种选择该图形的方式。图元名仅仅在当前编辑阶段有效，下次打开该图形时将被分配新的图元名。这一点与句柄不同，句柄是随文件保存的。

1.1.6 表

AutoLISP 存储和处理数据最有效的方式是表。鉴于 AutoCAD 的数据是以链表的方式进行存储的，所以 AutoLISP 使用表这一数据类型。

所有的图形都是由点、线、面组成的，而线、面可以再分解为点和点的组合，所以，点是最基本的元素。AutoLISP 使用包含点的坐标的表即可表达、处理所有的图形，例如，平面上的点可以表示为（x y），空间的点可以表示为（x y z）。

取消坐标的意义，读者可以认为，表就是包含在括号中以空格隔开的一组相关值，例如，（1.2 3.4 5.6）、（"it" "a" "sheep"）、（2 "three"）都是表。可以通过 list 函数创建和处理表。

1.1.7 选择集

一个或者多个图形对象都可以构成对象的集合，称为选择集。可以使用 ssget 函数返回包含所有图形对象的选择集：

```
命令: (ssget "X")
<Selection set: 14>;返回值
```

如果要把最后绘制的图形对象加入名称为 W 的选择集中，可以使用 ssadd 函数：

```
命令: (setq w (ssadd(entlast)))
<Selection set: 12>;返回值
```

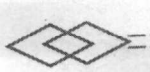
1.2 函数类

函数是许多计算机语言的重要组成部分，在 AutoLISP 语言中，函数也占有非常重要的地位。进行 AutoLISP 程序设计时，通过众多函数才能实现赋值、计算、输入输出以及编写条件语句、循环语句和子程序等。AutoLISP 语言中以表的形式写出所有的函数，每个函数在程序中表现为一条或多条语句，执行特定的功能，最后返回某种 AutoLISP 数据。函数的名称是表中的第一个元素，后续元素是函数必需的参数。

AutoLISP 预定义了 140 多个函数，本书介绍一些基本的和常用的函数，它们可以分为 14 类：赋值函数、计算函数和三角函数；逻辑函数和关系函数；交互性输入数据函数；输出和输入函数；文件操作函数；条件执行函数；循环函数；表处理函数；类型转换函数；字符串处理函数；求值函数；与 AutoCAD 交流的函数；访问 AutoCAD 实体的函数；定义、调用函数。

1.2.1 赋值函数、计算函数和三角函数

在 AutoCAD 2004 编程开发中，要处理大量图形的数据。本小节介绍的三种函数，就是计算数据用的。



赋值函数

赋值函数是基本和常用的函数。其格式为：

```
(setq <符号><值或表达式> [<符号> <值或表达式>]……)
```

它将表达式的值赋给符号，并返回最后一个表达式的值，例如：

```
(setq a 125);把 125 赋给 a，返回 125
(setq b 26 c 4.7);b 值为 26，c 值为 4.7，返回 4.7
(setq s "to");s 为字符串变量，其值为 to，返回 to
(setq ab (2 4));ab 的值为表(2 4)，返回(2 4)
```

表达式(setq x nil)和(setq y nil)则表示释放变量 x 和 y 所占用的内存空间。

计算函数

计算函数的格式为：

```
(运算符  运算数 1  运算数 2……)
```

这里运算符一定要放在最前面，这不同于一般的数学表达式。运算符与运算数之间以及运算数与运算数之间至少要有一个空格。举例如下：

```
求和函数：(+ 1 3);返回 4
差函数：(- 10 2 4);返回 4
乘函数：(* 0.4 3 -2);返回-2.4
除函数：( /  1000 5);返回 200
求余函数：(rem 7 3);返回 1
加 1 函数：(1 + 5);返回 6
减 1 函数：(1 - 5);返回 4
绝对值函数：(ABS -9);返回 9
幂函数：(expt - 2.0 3);返回-8.0
指数函数：(EXP l0);返回 2.718282
自然对数函数：(log 4.5);返回 1.504077
平方根函数：(sqrt 4);返回 2.0
求最大值函数：(max 4.07 -149);返回 4.07
求最小值函数：(min 74 2 84 5);返回 2
```

三角函数

```
正弦函数：(sin 1.0);返回 0.841471（弧度）
余弦函数：(cos 1.0);返回 0.540302
反正切函数：(atan 1.0);返回 0.785398
(atan - 3.0 2.0);返回 0.982793（即返回<数 1> / <数 2>的反正切值）
;反正切函数返回的弧度范围为-π到+π
```

1.2.2 逻辑函数和关系函数

几乎所有的计算机语言都有逻辑函数和关系函数，学习过 C 语言、B 语言的读者，很快就能掌握它们，只是语法不同而已。

逻辑函数

逻辑函数的返回值只有 T（真，即非零）和 nil（假，即零）两种。

逻辑函数有：

(not <表达式>)；返回表达式的逻辑“非”。

(null <表达式>)；当表达式为 nil 时返回 T，否则返回 nil。

(or <表达式 1> <表达式 2>……)；其中如果至少有一个表达式的值为 T，则函数的值为 T；若各表达式的值均为 nil，则函数的值为 nil。

(and <表达式 1> <表达式 2>……)；返回一系列表达式的逻辑“与”，该函数对表达式从左至右求值，如有一个表达式的值为 nil，则停止求值并返回 nil；当对所有表达式求值的结果都不为 nil 时才返回 T。

关系函数

它的一般形式为：

```
(<关系函数符号> <数 1> <数 2>……)
```

关系函数包括相等函数（=）、不相等函数（/=）、小于函数（<）、小于等于函数（<=）、大于函数（>）、大于等于函数（>=）。从左向右排列的数，如果均满足函数规定的关系，则函数的值为 T；若其中一个不满足，则函数的值为 nil。

1.2.3 交互性输入数据函数

交互性输入数据函数指的是从键盘或其他定点装置输入单个数据的一些函数。

（1）输入实型数函数(getreal [<提示>])

这个函数暂停下来等待输入一个实型数，输入后返回该实型数的值。<提示>是做提示用的，即显示在屏幕上的字符串。例如：

```
(getreal "Enter width a")
```

（2）输入整型数函数(getint [<提示>])

该函数等待输入一整型数，输入后返回该整型数的值。

（3）输入角度函数(getangle [<提示>])

该函数等待输入一角度值，以“度”计量，但返回弧度值。

（4）输入点表函数(getpoint [<提示>])

该函数等待输入一个点，并返回该点的坐标（点表）。调用 getpoint 函数，可以采用 AutoCAD 允许的任何点输入方式，如输入点的坐标数值，用光标移动输入或用鼠标来输入点等，这个函数还可有这样的形式：

(getpoint [<点>] [<提示>])

此时当前光标位置和[<点>]之间会有一条橡皮筋线作为辅助观察输入点。

（5）输入距离函数(getdist [<点>] [<提示>])

该函数让用户输入一个距离值，也可输入两个点，系统则把两点间的距离作为输入值。如果有选项[<点>]，则只要再输入一个点即可。此时，这两点之间有一条橡皮筋线，以帮助用户观察所设距离。该函数返回实数型的距离数值。

（6）输入字符串函数(getstring [<cr>] [<提示>])

这个函数暂停下来，等待输入一个字符串，并返回该字符串。如果提供了<cr>且不为 nil，则输入字符串必须以回车键结束，这样可在输入的字符中包含空格；如果没有<cr>或其值为

nil，则空格键和回车键都将结束字符串的输入。例如：

```
(setq s (getstring T "Enter your name"));调用该函数，允许输入的字符串中含有空格
```

（7）输入关键字函数(getkword [<提示>])

用于输入关键字。调用 getkword 前应先调用 initget 函数设置一个有效的关键字表。

getkword 将用户的输入与关键字表进行比较，返回匹配的关键字（字符串）。比较的方法是将关键字表中的大写字符部分与输入字符串的前 N 个字符比较，N 为关键字表中用于比较的大写字符部分的字符个数，这种比较不计大小写。如果相同，则认为匹配，返回该关键字（其中包含关键字中的小写部分），否则比较关键字表中的下一个关键字，直到有一个匹配时终止。如果都不匹配，则要求用户重新输入。如果是空输入且允许空输入，或者没有建立关键字表，都将返回 nil。例如：

```
(initget 1 "Yes NO")
(setq a (getkword "\n Are you sure[Y / N]? "))
```

其中，"Yes NO"为关键字表，1 表示不接受空输入。执行这两个函数，系统将接受用户输入“Y”、“yes”或“NO”等字符串（大小写不限），并把 a 置成相应的字符串。如果用户输入其他字符或字符串，系统都认为不匹配，而要求再输入。

1.2.4 输出和输入函数

下面将介绍输出函数与输入函数的相关知识。

输出函数

输出函数有以下几种：

（1）(prinl <表达式> [<文件描述符>])

该函数在屏幕上打印<表达式>的值，并返回这个值。如果<表达式>是一个含有控制字符的字符串，那么这些控制字符不起作用，而是原样打出。下面是控制字符及其含义：

```
\e        代表 Esc
\n        代表换行
\r        代表回车
\t        代表 Tab
\nnn      代表八进制码为 nnn 的字符
```

例如：

```
(setq a 248)
(prinl a);打印 248，返回 248
(prinl"\t Hello! ");打印“\t Hello!”，返回“\t Hello!”
```

如果该函数指定了<文件描述符>（且为一个为写而打开的文件描述符），则将 <表达式> 的值按照它在屏幕上显示的格式写入那个文件。

（2）(princ <表达式>[<文件描述符>])

函数 princ 与 prinl 基本相同，惟一的差别在于 princ 能够实现<表达式>中控制字符的控制功能。例如：

```
(princ "\t Hello! ");打印"          Hello! "
```

（3）(print <表达式> [<文件描述符>])

除了在打印<表达式>值前先自动换行及在打印<表达式>值后加上一个空格外，该函数和

prinl 完全相同。

（4）(prompt <信息>)

该函数将<信息>显示在用户的屏幕上，返回 nil，<信息>是一个字符串。在双屏幕配置中，prompt 将在两个屏幕上都显示<信息>，而 princ 却不能。

（5）(write-char <数> [<文件描述符>])

该函数将一个字符写到屏幕上或写到由<文件描述符>表示的打开的文件中，其中<数>是所写字符的 ASCII 码，也是函数的返回值。例如：

```
(write-char 67)
```

它将字母 c 写到屏幕上。

（6）(write-line <字符串> [<文件描述符>])

该函数将<字符串>写到屏幕上或写到由<文件描述符>表示的已打开的文件中，它返回用引号引起来的<字符串>，但写到文件中则省略引号，例如，fp 是一个有效的已打开的文件描述符，则：

```
(write-line "DEMO" fp)
```

将字符串 DEMO 写到 fp 描述的打开的文件上。

输入函数

输入函数有以下几种：

（1）(read <字符串>)

该函数返回从<字符串>中取得的第一个表或原子，并返回相应的数据。<字符串>中不能含有空格。

（2）(read-char [<文件描述符>])

该函数从键盘缓冲区中或从<文件描述符>表示的打开的文件中读入一个字符。它返回一个整型数，这个数是代表读入字符的 ASCII 码。若当键盘缓冲区为空时，(read-char) 等待用户输入，如用户输入 ABC 并按回车键，则(read-char)返回字符 A 的 ASCII 码 65。对(read-char)的以后三次调用将分别返回 66、67 和 10（即换行符）。如果再一次调用（read-char），它又将等待输入。

（3）(read-line [<文件描述符>])

从键盘或从<文件描述符>指定的文件中读入一行字符串。如果遇到了文件结束符，将返回 nil，否则它返回所读的字符串。若 fp 是一个打开的有效的文件描述符，则（read-line fp）将返回该文件的下一行，读后自动移动文件指针到下一行的行首。

1.2.5 文件操作函数

文件操作函数主要用于打开或关闭文件。

（1）(open <文件名> <状态>)

该函数打开文件，以便其他 I/O 函数进行存取。当打开成功时，返回文件描述符，否则返回 nil。<状态>为读写标志，必须是一个小写的单个字母，其中，r 为读方式，w 为写方式，a 为添加方式。例如：

```
(setq fp1 (open "exam.dat" "10"))
```

```
(setq fp2 (open "c:\\user\\user.dat" "r"))
或(setq fp2 (open "C:\\user\\user.dat" "r"))
```

<文件名>中可包含路径名，此时必须用“\\”才能在字符串中得到一个反斜杠“\”。

（2）(close <文件描述符>)

该函数关闭指定文件，并返回 nil。文件关闭后，其对应的文件描述符不再有效。

1.2.6 条件执行函数

条件执行函数有两种：

（1）(if <测试式> <表达式 1> [<表达式 2>])

当测试式为 T 时，执行<表达式 1>，否则执行<表达式 2>；当没有<表达式 2>时，如果测试式为 nil 时，该函数返回 nil，否则返回<表达式 1>的值。例如：

```
(if (> a b)(setq c 3)(setq c 4));当 a>b 时，c 值为 3，否则 c 值为 4
```

但 if 函数只能计算一个表达式，这显然不能满足实际需要。为此，AutoLISP 提供了另一函数 progn，称为顺序处理函数，其格式为：

(progn <表达式 1> <表达式 2>……)

该函数依次执行随后的各表达式，并返回最后表达式的求值结果。常用它在只能用一个表达式的地方对多个表达式进行计算，例如：

```
(if   (= a b))
(progn
(setq a (+ a 10))
(setq b (- b 9))
)
```

（2）(cond (<测试式 1> <表达式 1>)(<测试式 2> <表达式 2>)……)

该函数接管任意数目的表作为变元。它依次对各测试式进行计算，一旦该式不为 nil，则执行后面的表达式而不再测试以后的式子，例如：

```
(setq yn (getstring "\nEnteryourselect[Y/N]: "))
    (cond
    ((= yn "Y")(setq test 1))
    ((= yn "y")(setq test 1))
    ((= yn "N")(setq test 0))
    ((= yn "n")(setq test 0))
    (T (prompt "\nInvalid selection must be Y Or N"))
    )
```

1.2.7 循环函数

循环函数分为条件循环函数和重复循环函数两类，下面将分别进行介绍。

（1）条件循环函数(while <测试式> <表达式>……)

该函数先计算<测试式>，如果不为 nil，就计算后面的<表达式>（可有多个），然后再计算<测试式>，这样一直循环到<测试式>为 nil 时停止，并返回最后计算的表达式的值。使用该函数时，注意要有适当条件保证程序不至于死循环。

（2）重复循环函数(repeat <数> <表达式>……)