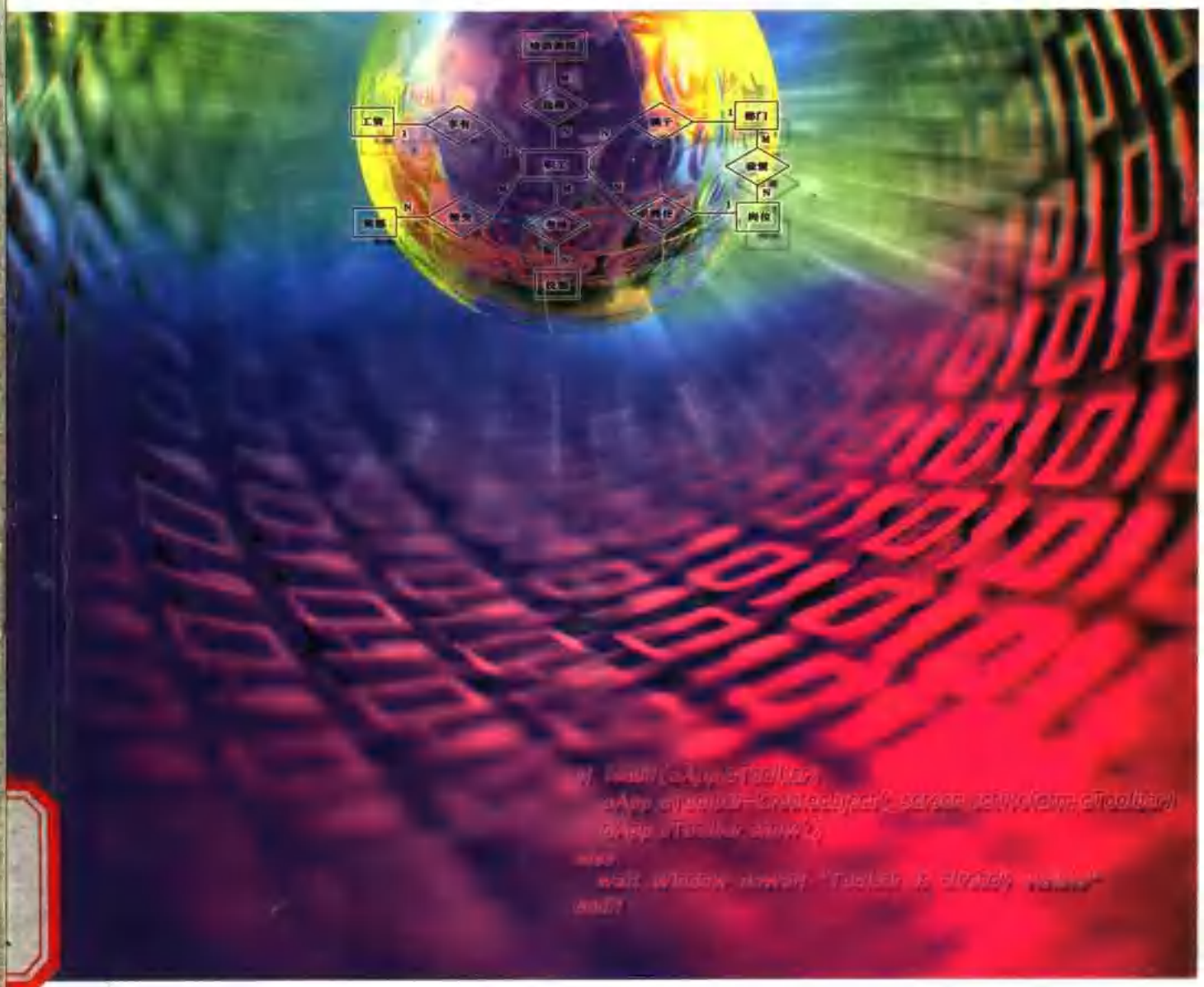
Visual FoxPro 实用教程

—— Visual FoxPro 编程

罗小林 施永香 主编



中国石化出版社

Visual FoxPro 实用教程

——Visual FoxPro 编程

罗小林　施永香　主编

中国石化出版社

内 容 提 要

Visual FoxPro 具有功能强、界面友好、工具便捷、编程机制简单
拥护，广泛应用于经济管理中。最新版本的 Visual FoxPro 6.0，在继
点的基础上，更增加了许多功能，在社会上有着更广泛的应用。

本教程分《Visual FoxPro 基础》与《Visual FoxPro 编程》两册。
数据库的基础及数据库的操作，内容简单、实用，注重可操作性，适用
的读者。《Visual FoxPro 编程》通过具体的实例介绍 Visual FoxPro 的
编写一些实用的应用程序，适用于具有一定 Visual FoxPro 基础的读

本教程可作为数据库应用软件学习及培训教材，也适合作为 Visu
学教材

图书在版编目(CIP)数据

Visual FoxPro 实用教程.Visual FoxPro 编程/
罗小林，施永香主编.
—北京：中国石化出版社，2002
ISBN 7-80164-281-3

Ⅰ.V… Ⅱ.①罗…②施… Ⅲ.关系数据库—数据库管.
系统，Visual FoxPro-教材 Ⅳ.TP311.138

中国版本图书馆 CIP 数据核字(2002)第 074059 号

中国石化出版社出版发行
地址：北京市东城区安定门外大街 58 号
邮编：100011 电话：(010) 84271850
http://www.sinopec-press.com
E-mail：press@sinopec.com.cn
北京精美实华图文制作中心排版
河北省徐水县印刷厂印刷
新华书店北京发行所经销
*
787×1092 毫米 16 开本 12.5 印张 319 千字 印 1—4000
2002 年 9 月第 1 版 2002 年 9 月第 1 次印刷
定价：25.00 元

前　言

Visual FoxPro 是一种易学习、功能强、效率高的数据管理和开发工具。无论是在组织信息、运行查询、创建集成的关系数据库管理系统，还是为最终用户编写功能全面的数据库管理应用程序，它都能非常胜任。

Visual FoxPro 具有功能强、速度快、界面友好等特点。它提供了众多的向导、生成器和设计器，用于帮助用户快速开发应用程序；与过去的表相比，Visual FoxPro 中的数据库是多表及由表生成的查询、视图、连接、关联、有效性规则和说明、缺省值、触发器等的综合管理者，而不再是单一的一张表；数据库中表与表的关系是以图形方式表示的，所以数据的关系特别直观明了；它的对象与事件模型使得用户可以快速建立应用程序；它不仅支持传统的过程式编程，而且支持面向对象编程，可以充分利用面向对象编程的所有特点。最新版本的 Visual FoxPro 6.0 对过去版本的功能做了更好地完善和改进。

经济管理工作中存在着大量的数据管理，而 Visual FoxPro 的丰富功能可以为我们的数据管理工作提供强大的技术支持和帮助。另外，由于 Visual FoxPro 易学好用，为我们在经济管理中的运用提供了良好的基础和可能。

基于此我们编写了这套《Visual FoxPro 实用教程》。本教程分两册，分别为《Visual FoxPro 基础》、《Visual FoxPro 编程》。《Visual FoxPro 基础》是 Visual FoxPro 的入门教程，适用于刚接触 Visual FoxPro 的读者。编写的主导思想是力求简单、实用，尤其注重可操作性　绝大部分的操作都是窗口下的界面方式，无须记忆大量的命令，因此大部分内容通过自学即可掌握。《Visual FoxPro 编程》适用于具有一定 Visual FoxPro 基础的读者，通过具体的实例学习可以基本掌握 Visual FoxPro 的编程规则，并能初步独立编写一些实用的应用程序　为帮助读者充分掌握运用，在每章后都提供了实用的练习。

学习 Visual FoxPro，我们认为首要的是扎实掌握表（数据库表）的建立和运用。因为 Visual FoxPro 的查询、视图、表单、报表、编程等都是建立在其基础之上的，不能全面理解与掌握表的建立和操作，将无法充分运用 Visual FoxPro 提供的强大功能。因此我们将表作为重要内容单独列为一章放在上册最前面的位置。

本教程由罗小林、施永香主编，孙卫、黄作明、丛秋实副主编。《Visual FoxPro 基础》第 1 章由施永香编写，第 2 章、第 3 章、第 4 章由孙卫编写，第 5 章、第 6 章、第 7 章、第 9 章由黄作明、丛秋实编写，第 8 章由张艳编写。《Visual FoxPro 编程》第 1 章、第 2 章、第 3 章由施永香编写，第 4 章由孙卫编写，第 5 章、第 6 章由黄作明、丛秋实编写，实例由荆霞编写。全书由罗小林、施永香总纂。本书在编写过程中得到了钟莹、李志敏同志的大力支持，在此深表谢意。

由于作者水平有限，书中难免错误和缺点，敬请广大读者批评指正。

编　者

目　录

第一章 Visual FoxPro 语言概述

VFP 虽然不是高级语言，但也包含了许多高级语言要素，例如，数据类型、常量、变量、函数等。

1.1 数据类型

数据类型是数据的基本属性。对数据进行操作的时候，只有同类型的数据才能进行操作，若对不同类型的数据进行操作，将被系统作为出错处理。

Visual FoxPro 提供了多种数据类型，它们是：

（1）字符型（Character 用代号 C 表示）：由字母、汉字、数字、空格、符号和标点等组成，通常用来存储姓名、单位名称、地址等类信息，宽度由用户定义，但不超过 254 个字符。当信息量较大时宜放到备注型字段中。

字符型常量是用定界符括起来的字符串。定界符有三种，即单引号、双引号和方括号，如："A0001"、'男'、[信息管理系]都是合法的字符型常量表示形式。

注意：定界符是西文方式下的符号。

（2）货币型（Currency 用代号 Y 表示）：使用货币值时可使用货币数据类型以代替数值类型。货币型常量的表示是在数字的前面加上美元符号（$）。如：PRICE=$100

（3）日期型（Date 用代号 D 表示）：存储日期数据，其存储格式为“YYYYMMDD”，占 8 个字节。显示格式有多种，取决于当前系统的状态，可通过“工具”菜单下的“选项”命令设置或使用 SET DATE TO 命令设置，常用的为“MM/DD/YY”(月/日/年)格式。在年份表示时，是否显示世纪位可由 SET CENTURY ON/OFF 命令设定。若 SET CENTURY ON，则显示世纪位，即年份用四位表示。

日期型常量的表示方法是在一个日期型值的两端加上花括号，例如{^07-01-2002}。

格式中的符号^表明该日期格式是严格的，并按照年月日的格式来解释日期。其中的-可用/代替，例如{^07/01/2002}。

VFP 默认使用严格的日期格式，如果要使用其他的日期格式，可以先执行如下命令：

SET STRICTDATE TO 0

若要执行严格日期格式执行，可执行命令：

SET STRICTDATE TO 1

（4）日期时间型（DateTime 用代号 T 表示）：用于存储日期时间值，存储格式为“YYYYMMDD HHMMSS”。使用方法类似日期型。

（5）数值型（Numeric 用代号 N 表示）：用于表示数量，它由数字 0~9、正负号（+或

）和一个小数点（.）组成，长度 1~20 个字节。小数点的位置和字段的宽度由用户创建该字段时指定，宽度包含小数点和小数位数。

数值型常量是可以带正负号的整数或小数（正号可省略），如 123.4，还可以用科学记数法表示常量，如 25000 可表示成 2.5E+4。

（6）逻辑型（Logical 用代号 L 表示）：用于存储只有两个值即真(.T.)和假(.F.)的数据，其存储宽度固定为 1 个字节。

（7）浮点型（Float 用代号 F 表示）：以浮点数的形式存储数值数据，用于存放数据处理中的高精度数据，常用于科学计算，宽度的规定和数值型相同。

（8）双精度型（Double 用代号 B 表示）：用于取代一般的数值类型，以便能提供更高的数值精度。

（9）整型（Integer 用代号 I 表示）：用于无小数部分数值的存取，只用于表中字段的定义，具有固定的存储宽度 4 个字节。

（10）备注型（Memo 用代号 M 表示）：用于数据块的存储，当字符型数据的长度超过 254 个字符，或者字符型数据的长度长短不一，差别很大时，用备注型存储，例如，一个人的简历、一般表格的备注栏等等。存储备注型数据的宽度固定为 4 个字节，但这 4 个字节不是它真正的内容，其实际内容存放在一个主文件名与表文件名相同，扩展名为.FPT 的文件中。这 4 个字节用于存放指向备注文件中相应内容的指针。

（11）通用型（General 用代号 G 表示）：VFP 可以处理由其他应用程序生成的数据（OLE 对象），如图像、文档、表格、视频等，这些外部数据可以链接或嵌入到表的通用型字段中。如一个学生的照片、一段音乐等。一个 OLE 对象的具体内容可以是一张 EXCEL 的电子表格、或一个 WORD 的文档、图片、声音等。存储通用型字段的长度固定为 4 个字节，与备注型字段类似，该字段的真正内容存放在备注文件中。

还有两种是二进制字符型和二进制备注型，适用场合同对应的字符型和备注型。只是当代码页改变时，其值不会随之改变。所谓代码页是指各国因语系不同，必须使用不同的代码。当代码页改变时，VFP 会自动转换成相应的语系。而二进制字符型和二进制备注型的数据并不随着代码页的转换而改变。

1.2 数 据 存 储

在 VFP 中，存储数据的容器有常量、变量、数组和字段、对象等。

1.2.1 常量

常量是指在程序运行过程中其值不发生变化的量。VFP 支持字符、数值、日期、日期时间、货币和逻辑 6 种类型的常量。

数值型常量，如 123，3.14159，2000.00

字符型常量，如"信息管理系"，"男"，"960201"

日期型常量，如{^08-01-2002}

逻辑型常量，如.T.和.F.。

VFP 还可以用 DEFINE 预处理命令创建编译时常量，例如：

```
#DEFINE  PI 3.14159
```

这样在程序中，凡是要用到 3.14159 的地方，都可以用 PI 代替。如果要把程序中多个地方出现的 3.14159 改为 3.1415926，则只需修改预编译命令一次即可。

需要说明的是，在程序中的编译时常量不能再作为变量使用。

1.2.2 变量

变量是在程序运行过程中其值可能发生变化的量。在 VFP 中，变量分为字段变量、内存变量和数组。

1）字段变量：每个字段名就是一个字段变量。这是一种多值变量，表中每一条记录对应每一字段都有一个取值。当用字段名作变量时，它的当前值就是表的当前记录的值。

2）内存变量：是用户通过命令或程序临时定义的变量，表示一块工作单元。变量名由字母、汉字、数字或下划线组成，最多为 10 个字符。

内存变量的定义有两种方式：一种是使用 STORE 赋值命令，它可以一次给多个内存变量赋值，如：

```
STORE  0  TO  N，S
STORE  "990101"  TO  XH1
```

另一种是使用赋值语句（等号“=”）操作，如：

```
N=0
S=0
XH1="990101"
```

上面两种方式创建变量，都是在定义的同时并给它们赋值。

如果只创建它们，可以使用下面的这些语句：

```
PUBLIC A，B，C                &&创建全局变量
PRIVATE S，Y，Z                &&创建私有变量
LOCAL BOY，MAKES，GOOD        &&创建局部变量
```

全局变量（又称公共变量）可以在应用程序的任何位置使用和修改；局部变量只在创建的过程或函数内有效，而不能在上一级或下一级的过程或函数内使用或修改；私有变量不能在上一级但可以在下一级的过程或函数内使用或修改。

几乎不像任何其他的程序语言，VFP 里的变量能改变类型。举例来说，下列内容完全合法：

```
Name = 3              &&数值型
Name = "王一平"       &&字符型
Name = .F.            &&逻辑型
```

当内存变量和字段名变量同名时，字段名优先。若非要访问内存变量，可在变量名前面加 m.或 m->引用。

例如：USE JS

```
?XM          &&显示教师表中第一个记录的姓名“王一平”
```

```
XM="李红"
? XM          &&还是显示教师表中第一个人的姓名“王一平”
? M->XM       &&显示“李红”
```

1.2.3 数组

数组是一组有序数据的集合。这里的有序数据可以是不同类型的数据，数组中的每一个数据称为数组元素。

在 VFP 中，最多可以定义二维数组，下标从 1 开始计数。数组在使用之前，必须先定义后使用。

1）数组的创建或声明

DIMENSION/DECLARE　数组名 1[下标 1[,下标 2]][,数组名 2[下标 1[,下标 2]]...]

例如：DIMENSION　ARRAY[3]

则定义了一维数组 ARRAY，数组下标为 3，有三个数组元素，分别为 ARRAY[1]，ARRAY[2]，ARRAY[3]。

例如：DECLARE　BB[2,3]

则定义了二维数组 BB，数组下标有两个，分别为 2 和 3，有 2*3 共 6 个数组元素，按先行后列排列，分别为 BB[1,1]，BB[1,2]，BB[1,3]，BB[2,1]，BB[2,2]，BB[2,3]。

二维数组各元素在内存中按行的顺序存储，而且也可以按一维数组来表示其数组元素。例如上述二维数组 BB 中的元素 BB[2][2]数组声明后，位于第 5 位，所以该元素也可用 BB[5]表示。

当数组被声明后，每个数组元素被默认地赋予逻辑型，值为.F.。

2)数组的赋值

可以给数组的单个数组元素赋值，也可以给整个数组（数组名）赋值。

例如：

```
DIMENSION　A[3]
A[1]=123
A[2]=456
? A[1],A[2],A[3]
```

输出结果是：123　456　.F.

例如：

```
DIMENSION BB[3,4]
BB=123
? BB[1,1], BB[1,2]
```

输出结果为：123　123

1.3 运 算 符

1. 字符运算符

+运算符：用于连接两个字符串。

-运算符：用于连接两个字符串，并将第一个字符串尾部的空格移到结果字符串的尾部。

例如：

```
? " FoX" +" Fro"                &&结果为" FoxPro"
? " Fox    " —" Pro"            &&结果为" FoxPro    "
```

用于字符操作的还有一种操作符是包含（$），其含义是判断$ 前的字符串是否包含在后一个字符串中，结果是逻辑型。例如，

```
? " Fox" $" FoxPro"          &&结果为.T.
? " abcd" $" efgh"           &&结果为.F.
```

2. 日期和日期时间操作符

两个日期型数据不能相加，一个日期型数据只能加一个数值型的量（代表“日”）；两个日期型数据可以相减，结果为数值型。同样，两个日期时间型数据不能相加，一个日期时间型数据只能加一个数值型的量（代表“秒”）；两个日期时间型数据可以相减，结果为数值型。

例如：{^08-20-2001}+5={^08-25-2001}

{^08-20-2001}-{^08-16-2001}=4

3. 关系操作符

关系操作符主要用于同类型之间的比较。有>，>=，<，<=，=，!=或<>或#，==共七种运算符。关系运算的结果为逻辑型。

例如：?4>3

```
.T.
STORE 5 TO   X
Y=6
Z=7
?X>Y
.F.
?Y<Z
.T.
? " 讲师" >" 教授"          &&按拼音字母排序
  .T.
```

4. 逻辑运算符

逻辑运算符有.NOT.（!），.AND.，.OR.共四种，主要用于逻辑数据类型，逻辑运算的结果为逻辑型。

逻辑与（AND）操作：X .AND.Y　当 X，Y 都为.T.时结果即为.T.。

逻辑或（OR）操作：X .OR.Y　当 X，Y 中有一个为.T.时，结果即为.T.。

例如：STORE 5 TO   X

```
Y=6
Z=7
?X>Y .AND. Y<Z
.F.
```

```
?X<Y .AND. Y<Z
 .T.
?X>Y .OR. Y<Z
 .T.
?.NOT.X>Y
.T.
```

5. 数值操作符

也称算术运算符，如（）、/（除），*（乘），^或**（乘方）和%（取模）、+、-等，与它们在计算器上的作用一样。

例如：？9%2　　&&结果为 1

1.4 函　　数

函数是由函数名、函数参数、函数的返回值三要素组成。函数可返回特定类型的数据。例如，　STR(　)　函数和　VAL(　)　函数分别返回字符型和数值型数据。可以用多种方法调用　VFP　函数，例如：将函数的返回值赋给某个变量。

```
dToday = DATE(  )          &&用变量 dToday 保存当前系统日期。
```

在活动输出窗口中输出返回值。

```
? TIME(  )                 && 在Visual FoxPro 主窗口中输出当前系统时间。
```

函数嵌套。

```
? DOW(DATE(  ))            &&输出今天是星期几。
```

下面介绍系统常用的几类函数

1.4.1　数值运算函数

1）取整函数 INT

语法：INT（〈expN〉）

功能：返回〈expN〉的整数部分

返回值类型：　N 型

实例：？INT（12.58）

结果：12

2）求平方根函数 SQRT

语法：SQRT（〈expN〉）

功能：求〈expN〉的平方根

返回值类型：　N 型

实例：？SQRT（9）

结果：3.00

3）取模函数 MOD

语法：MOD（〈expN1〉，〈expN2〉）

功能：求以〈expN1〉为被除数，〈expN2〉为除数的模

返回值类型： N 型

实例 1：？MOD（100，3）

结果：1

实例 2：？MOD（100，-3）

结果：-2

实例 3：？MOD（-100，-3）

结果：-1

注意：MOD（ ）函数，若两参数的值大于 0，则模为余数；若两参数一正一负，则模为余数与〈expN2>之和；若两参数均为负，则模为余数取反。最终模的符号和〈expN2>相同。

4）求最大值函数 MAX

语法：MAX（〈expN1〉，〈expN2〉）

功能：求〈expN1〉和〈expN2〉的较大者

返回值类型： 取决于<expN1〉和〈expN2>的类型，可以是 N 型、C 型、D 型。

实例：？MAX（13，8）

结果：13

5）求最小值函数 MIN

语法：MIN（〈expN1〉，〈expN2〉）

功能：求〈expN1〉和〈expN2〉的较小者

返回值类型： 取决于<expN1〉和〈expN2>的类型，可以是 N 型、C 型、D 型。

实例：？MIN（13，8）

结果：8

6）符号函数 SIGN

语法：SIGN（〈expN〉）

功能：返回〈expN〉符号，若为正返回 1；为负返回-1；为零，返回 0

返回值类型： N 型

实例：？SIGN（456.23）

结果：1

7）截尾函数 ROUND（ ）

语法：ROUND（<expn1>,<expn2>）

功能：<expn1>四舍五入,指定对第几位四舍五入。

返回值类型：数值型。

说明：<expn2>的取值，从小数点位置开始记数，向后分别为 1，2，3，…，向前分别为 0,-1，-2，-3，…。

实例 1：?ROUND(456.789,2)

结果是：456.79

实例 2：?ROUND(456.789,-2)

结果是：500

实例 3：?ROUND(456.789,0)

结果是：457

8）求随机数函数 RAND（　）

语法：RAND([nSeedValue])

功能：返回一个 0 到 1 之间的随机数。

返回值类型：数值型。

说明：[nSeedValue]：指定的种子数值，它决定 RAND（　）函数返回的数值序列。在第一次发出 RAND（　）函数时使用一个种子数值 nSeedValue 后，以后再使用 RAND（　）函数时不带 nSeedValue，将得到一个随机数序列。如果第二次发出 RAND（　）函数时使用同样的种子数值 nSeedValue，那么 RAND(　)返回同样的随机数序列。如果第一次发出 RAND（　）时使用的 nSeedValue 参数是负数，那么将使用来自系统时钟的种子值。若要获得随机程度最大的数字序列，可以最初用一个负的参数发出 RAND（　）函数，然后再不带参数发出 RAND（　）函数。如果省略种子数值，RAND（　）函数将采用默认值 100,001。

实例：RAND（　），RAND（-10）

1.4.2　字符处理函数

1）空格构造函数 SPACE

语法：SPACE（〈expN〉）

功能：返回长度为〈expN〉的空格串

返回值类型：　C 型

实例：？SPACE（3）

结果：″　　″

2）求字符串长度函数 LEN

语法：LEN（〈expC〉）

功能：返回〈expC〉的长度

返回值类型：　N 型

说明：一个西文占一个字符，一个汉字占两个字符长度。

实例 1：？LEN（″英语 ABC″）

结果：7

实例 2：？LEN（″fox″+SPACE(2)+″pro″）

结果：8

3）取子串位置函数 AT

语法：AT（〈expC1〉，〈expC2〉[，〈n〉]）

功能：搜索〈expC1〉在〈expC2〉中的位置

返回值类型：　N 型

实例 1：？AT（″A″,″CBADEFAG″）

结果：3

实例 2：？AT（" A"，" CBADEFAG"，2）

结果：7

说明：[，〈n〉]为可选项，表示〈expC1〉在〈expC2〉中第 n 次出现的位置。AT（ ）函数区分大小写，ATC（ ）函数不区分大小写。

4）求子串函数 SUBSTR

语法：SUBSTR（〈expC〉，〈expN1〉[，〈expN2〉]）

功能：从〈expC〉中第〈expN1〉位置开始取长度为〈expN2〉的字符串。

返回值类型： C 型

说明：当〈expN2〉缺省时，从〈expN1〉位置开始一直取到〈expC〉的尾部。

实例 1：？SUBSTR（" 计算机中心"，7，2）

结果：中

实例 2：？SUBSTR（" ABCDEFGH"，3）

结果：" CDEFGH"

5）去除首尾空格函数 ALLTRIM

语法：ALLTRIM（〈expC〉）

功能：删除一个字符串的前后空格

返回值类型： C 型

实例：？ALLTRIM（"　　ABCDEFGH　　"）

结果：" ABCDEFGH"

6）去除字符串尾部的空格 TRIM（ ）函数

语法：TRIM（〈expC〉）

功能：删除一个字符串的右边的（尾部）空格。

返回值类型： C 型

实例：？TRIM（"　　ABCDEFGH　　"）

结果："　　ABCDEFGH"

7）求左函数 LEFT（ ）

语法：LEFT(expC, expN)

功能：从字符表达式 expC 最左边字符开始，返回指定数目 expN 的字符。

返回值类型：C 型。

说明：expC 指定的字符表达式，LEFT（ ）函数从中返回字符。expN 指定从字符表达式中返回的字符个数。若 expN 的值大于 expC 的长度，则返回字符表达式的全部字符。LEFT（ ）函数与起始位置为 1 的 SUBSTR（ ）函数是等价的。

实例：？LEFT(" FOXPRO", 3)

FOX

8）求右函数 RIGHT（ ）

语法：RIGHT(expC, expN)

功能：从字符串 expC 的右边开始返回指定数目 expN 的字符。

返回值类型：C 型。

说明：expC 指定的字符表达式，RIGHT( ) 函数从中返回最右边的若干字符。