目录

使用表达式和变量 1
使用表达式和变量：概述 1
使用测量例程中的表达式 1
查看表达式值 1
仅保持表达式值 2
在分支时使用表达式 2
在文件输入/输出时使用表达式 2
使用表达式构造器创建表达式 2
在表达式中使用变量 7
了解表达式的组成 8
文本 8
引用 9
变量 14
结构 17
指针 18
数组 19
表达式运算符 26
优先级 27
函数 28
函数列表 28

通过扫描段构造直线示例 57

操作数强制.....58

ID 表达式 60

进入报告对象属性.....62

从构造最小扫描圆中获取信息.....64

使用表达式和变量

使用表达式和变量：概述

表达式是与 PC-DMIS 流程控制命令一起使用的用户定义条件。使用流程控制语句可以在测量例程中测试这些条件，并根据条件是否符合，确定 PC-DMIS 采取的操作。

表达式是 PC-DMIS 完成指定任务的重要组成部分。将表达式和流程控制命令组合使用，可以发挥出 PC-DMIS 更强大的功能。

本章介绍如何在 PC-DMIS 的“编辑”窗口中创建和使用表达式。要处理表达式，应将 PC-DMIS 的编辑窗口置于命令模式。这样可以直接查看“编辑”窗口的代码。

本章包含以下主要主题：

- 使用测量例程中的表达式
- 使用表达式生成器创建表达式
- 在表达式中使用变量
- 了解表达式的组成
- 进入报告对象属性
- 访问扫描构造最小圆的信息

有关报告表达式的相关信息，请参阅“报告测量结果”一章中的“关于报告表达式”。

使用测量例程中的表达式

PC-DMIS编辑窗口的大多数可编辑字段中均允许使用表达式。可编辑字段通常是在编辑窗口处于命令模式按 TAB 键时，以黄色突出显示的字段。更改特征类型的字段不允许使用表达式。

示例：指定自动特征类型的自动特征框（如曲面点、自动圆、自动圆槽等）不允许使用表达式。

本主题的子主题提供可用表达式的完整参考。

查看表达式值

要查看表达式的值，将鼠标光标置于该表达式上，保留在该位置至少一秒。将对表达式求值，并在鼠标光标下方弹出一个黄色的小窗口，显示表达式及其当前值。

仅保持表达式值

在编辑窗口中立即对表达式进行计算，只保留值：

1. 在编辑窗口中选择表达式文本。
2. 在表达式文本前加上 ` 重音字符。

示例：假定在数值字段中键入了表达式 "**`1/7**"。表达式的值将立即计算出来，且该数值字段中仅有值 (**0.143**)。

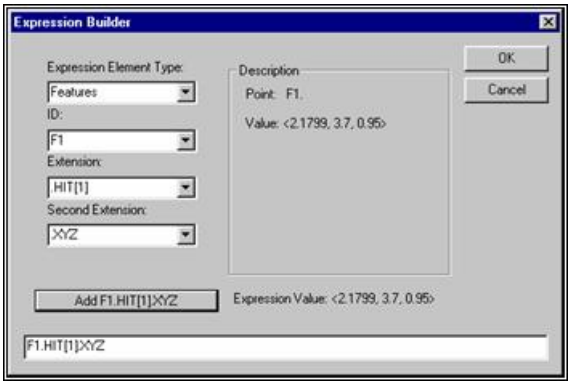
在分支时使用表达式

流程控制命令使用表达式决定例程执行的流程。有关何时可以或不可以出现分支的信息，请参见“使用流程控制进行分支”一章。

在文件输入/输出时使用表达式

将数据写入外部数据文件或从外部数据文件读取数据时，通常会使用变量和其它表达式，以便有效地管理和存储或显示数据。详细信息，请参阅“使用文件输入/输出”一章。

使用表达式构造器创建表达式



表达式构造器对话框

PC-DMIS
允许您通过直接键入，或使用**表达式生成器**的友好界面创建表达式，并将表达式添加到“编辑”窗口。**编辑 | 表达式**菜单项显示**表达式生成器**对话框。

使用这个对话框，可创建表达式，并将表达式插入到可编辑的字段。当光标处于允许输入表达式的字段时按 F2 键，将显示**表达式生成器**对话框。

表达式生成器对话框列出表达式可用的所有类型的运算符与函数。

通过键入创建表达式

通过键入直接在编辑窗口中创建表达式：

1. 打开“编辑”窗口（**视图 | 编辑窗口**）。
2. 将编辑窗口切换到命令模式。
3. 按TAB键将光标移动到要插入表达式的可编辑字段。字段用黄色高亮显示表示"可编辑"。
4. 键入表达式。

使用表达式构造器创建表达式

注意：对于需启用的表达式选项，必须处于命令模式。

若要使用表达式生成器对话框输入表达式（**编辑 | 表达式**）：

1. 打开“编辑”窗口（**视图 | 编辑窗口**）。
2. 将“编辑”窗口置于命令”模式（**浏览 | 命令模式**）。
3. 将光标移动到要插入表达式的可编辑字段。
4. 当光标处于允许输入表达式的字段时按 F2 键。屏幕上将显示**表达式生成器**对话框。**表达式生成器**对话框列出了所有类型的运算符、操作数及函数。通过这个对话框，可参照以下各项：
   - 可用的表达式类型
   - 变量
   - 特征
   - 尺寸
   - 坐标系
   - 注释
5. 从第一个下拉列表中选择表达式元素。根据您的选择，将出现其它下拉框。
6. 从 **ID** 下拉列表选择所需 ID。
7. 从**扩展名**下拉列表中选择扩展名。
8. 从**第二扩展名**下拉列表中选择另一个扩展名。如果表达式是可以使用的，**添加**按钮将可以使用。
9. 单击**新建**按钮。该表达式将出现在编辑框中。
10. 单击**确定**按钮。表达式将出现在“编辑”窗口中光标所在的位置。

注：您也可从其他对话框打开**表达式生成器**对话框：

- **If 表达式**对话框 - 选择**插入 | 流控制命令 | If Goto**。单击**表达式**按钮。
- **赋值**对话框-选择**插入|赋值**。单击**赋给**或**赋自**按钮。

一旦创建了表达式，PC-DMIS自动在下一个允许的位置插入表达式。

检查表达式正确性

当光标离开添加的表达式字段时，PC-DMIS会检查表达式的正确性。如果表达式有问题，会有错误信息指出可能是出现了无效的数字，或表达式文本变为红色。同样，表达式中涉及的不存在的对象也会显示为红色文本。

由于对于表达式正确性的测试发生在离开字段后，因此指向不存在对象而变为红色的字段（如CIRCLE1.X）会保持红色，即使添加了新对象（ex. CIRCLE1）。字段会保持红色直至表达式重新进行正确性测试。

要重新测试表达式的正确性，请执行以下步骤：

1. 将光标移动到表达式字段。
2. 按 F2 键。再次打开**表达式构造器**对话框。对表达式所做的任何更改均会显示在编辑框中。
3. 按ENTER键关闭该对话框。

表达式元素类型

表达式生成器对话框（**编辑 | 表达式**）中的表达式元素类型下拉列表列出了表达式可用的各种元素类型。这些选项包括：

- 功能
- 运算符
- 坐标系
- 注释
- 尺寸
- 特征
- 变量

标识

表达式生成器对话框（**编辑 | 表达式**）中的 ID 下拉列表依据在表达式元素类型下拉列表中选择得表达式元素类型，列出可用的项目集合。

例如：

- 如果从**表达式元素类型**下拉列表中选择**函数和运算符**，**ID** 下拉列表将包含可用函数和运算符的列表。
- 如果从**达式元素类型**下拉列表中选择“特征”，**ID** 下拉列表将显示测量例程中所有特征的 ID。

扩展名

如果在 **ID**
下拉列表中选择的项需要添加**扩展名**才能生成有效的表达式元素，表达式生成器对话框（**编辑 | 表达式**）中的扩展名下拉列表将启用。**扩展名**下拉列表根据在 ID
下拉列表中选择的项显示可用的扩展名。

示例：假定您从 **ID**
下拉列表选择了一项特征。之后在**扩展名**下拉列表中，会列出可用于参照该特征的数据的可能的扩展名列表（如 "X"、"Y"、"Z"、"Diam"、"Length" 等）。

可用的扩展名包含以下数据类型的实测值或理论值：

- 全部 -
  特征的所有值被赋给变量。请参考以下示例。
- X – 测点的 X 测量值
- Y – 测点的 Y 测量值
- Z – 测点的 Z 测量值
- XYZ – 测点的 XYZ 测量值
- I – 测点的 I 测量值
- J – 测点的 J 测量值
- K – 测点的 K 测量值
- IJK – 测点的 IJK 测量值

- TX – 测点的 X 理论值
- TY – 测点的 Y 理论值
- TZ – 测点的 Z 理论值
- TXYZ – 测点的 XYZ 理论值
- TI – 测点的 I 理论值
- TJ – 测点的 J 理论值
- TK – 测点的 K 理论值
- TIJK – 测点的 IJK 理论值

示例：

```
F1=GENERIC/POINT,DEPENDENT,CARTESIAN,$
        理论值/XYZ,<8,9,10>,$
        测定值/XYZ,<7.98,8.98,9.98>,$
        理论值/IJK,<1,0,0>,$
```

```
测定值/IJK, <1,0,0>
ASSIGN/MYFEATURE = F1.ALL
ASSIGN/V1 = MYFEATURE.X
ASSIGN/V2 = MYFEATURE.TX
```

V1 包含值 7.98

V2包含值8

第二扩展名

只有在**扩展名**下拉列表中选择的项需要添加第二扩展名才能生成有效的表达式元素，**第二扩展名**下拉列表才会启用。

示例：假定引用名称为 "D1" 的尺寸的 X 位置轴标称值。

您将：

1. 从 **ID** 下拉列表中选择 **D1**。
2. 从**扩展名**下拉列表选择 **X**。
3. 从**第二扩展名**下拉列表中选择 **Nom**。

添加按钮

使用下拉列表选择可用的或完整的表达式元素时，**添加**按钮将可供选择。此按钮显示要添加至表达式的文本。

例如，如果选择以下项：

- 从**表达式元素类型**列表中选择“尺寸”
- 从 **ID** 列表中选择 D1
- 从**扩展名**列表中选择 X
- 从**第二扩展名**列表中选择 Nom

则**添加**按钮将被启用，并具有以下文本：**添加 D1.X.NOM**。

单击**添加**按钮后，文本将出现在对话框底部的编辑框中。

注：单击**确定**按钮后，编辑框的文本将添加到“编辑”窗口中光标所在的表达式字段。如果从“编辑”窗口的表达式字段选择了项目，并且要添加的文本有圆括号，则选择的项目将插入到添加的文本的圆括号中。

编辑框

在**表达式生成器**对话框（**编辑 | 表达式**）底部，有一个显示当前表达式的编辑框。表达式可以在这个框里直接键入，或你可以使用**添加**按钮。

说明区域

表达式生成器对话框（**编辑 | 表达式**）还包含有**说明**区域，该区域提供了从下拉列表选择的项目的信息。**添加**按钮旁边的字段显示表达式的当前值。

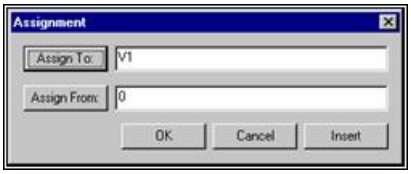
注：无效表达式的值为 0。

在表达式中使用变量

变量是用于存储值的对象。变量引用整数、实数、字符串或点操作数。使用表达式时需要变量。变量应具有名称和取值。名称用于变量的访问。名称为常量，值可以改变。使用 ASSIGN/ 命令可以为变量赋值。

例如，语句“赋值/V1 = 2”创建名为 V1，值为 2 的变量。“赋值/V2 = V1 + 2”访问 V1 的值。如果V1 的值仍为2,语句执行时 V2值将为4。

有关变量的详细信息，请参见变量。



赋值对话框

插入 | 赋值菜单选项显示**赋值**对话框。使用此对话框，可将数值指派给测量例程功能、尺寸或坐标系的变量或数据元素。使用赋值命令需要理解基本的 PC-DMIS 表达式。

赋给按钮 -
该按钮允许指定接收**赋自**框中计算结果的变量。通过**赋给**按钮选择的信息将会置于**赋给**框中。该值可以是变量的名称，也可以是对特征，尺寸或坐标系的数据元素的引用。

数学表达式的数值结果由术语“求值”来定义。

赋自按钮 -
该按钮允许将所赋的值放入**赋自**来源框中。如果此框中包含表达式，则将在执行时对表达式求值，计算结果将赋值到在**赋给**框中指定的对象。

插入按钮 -
该按钮可在开启**赋值**对话框时将赋值命令插入测量例程。可在不关闭对话框的情况下插入一系列赋值命令。

了解表达式的组成

表达式有以下类型的操作数：

- 整数
- 实数
- 字符串
- 点
- 特征指针
- 数组
- 功能

这些操作数在下面详细说明。

操作数类型

操作数存在的形式如下：

- 文本
- 引用
- 变量
- 结构
- 指针
- 数组

文本

***整数**：1、-6、209

实数：1、-6、2.4、-0.1、345.6789

字符串："Hello World"、"47"、"CIRCLE 1"

点：点不能用文字来表示。不过，运用 MPOINT 函数（如 `MPOINT(0,0,1)`、`MPOINT(2.2, 3.1, 4.0)`）可从其他字面值得出点。

指针：特征名加上大括号：`{CIR1}`、`{LIN2}`、`{F3}`

数组：数组不能使用文字表示。但是，可使用 ARRAY 函数（如 `ARRAY(3, 5, 6)`、`ARRAY("Hello", 2.3, 9)`）从其他字面值创建数组。这些函数创建 3 个元素的数组，第一个例子中为整数元素 3、5 和 6，第二个例子中为字符串元素 "Hello"，双精度元素 2.3，以及整数元素 9。

函数：函数不能使用文字表示。函数使用 `FUNCTION` 关键字定义，并通过变量 ID 访问。例如，`ASSIGN/Add2 = FUNCTION((X), X+2)` 定义的函数有一个自变量，并将该函数与 2 相加。变量 Add2 被分配给了这个函数。运用变量 Add2，可按如下调用函数。`ASSIGN/Result = Add2(5)`。分配给结果的值为 7。

注：数值型字面值解释为实数，除非运算符或函数暗指使用的是整数。例如，表达式10/8得到1.25，而不是1。还要注意，通过操作数强制型转运算符，还可以执行离散除法。表达式 INT(10) / INT(8) 求值结果为 1。

引用

参考就是取用测量例程中其他对象的数据成员。参考使用测量例程中对象的 ID，其后跟之以一个圆点及该对象的数据成员的扩充。

示例：若 CIRCLE1 为测量例程中某个测量圆的名称，则 `CIRCLE1.X` 表示 CIRCLE1 的 X 分量的测量值。所有参考均相对于当前坐标系在零件坐标中求值。

双精度类型的引用

可以使用以下引用表达式：

特征有效的扩展名进行双精度类型引用示例

格式：<特征 ID>.<扩展名> -> `CIRCLE1.X`

CIRCLE1.X CIRCLE1 的 X 测量值
CIRCLE1.Y CIRCLE1 的 Y 测量值
CIRCLE1.Z CIRCLE1 的 Z 测量值

CIRCLE1.TX CIRCLE1 的 X 理论（标称）值
CIRCLE1.TY CIRCLE1 的 Y 理论（标称）值
CIRCLE1.TZ CIRCLE1 的 Z 理论（标称）值

LINE1.SX LINE1 起点的 X 测量值
LINE1.SY
LINE1.SZ

LINE1.TSX LINE1 起点的 X 理论值
LINE1.TSY
LINE1.TSZ

LINE1.EX LINE1 终点的 X 测量值
LINE1.EY
LINE1.EZ

LINE1.TEX LINE1 终点的 X 理论值
LINE1.TEY
LINE1.TEZ

POINT.I POINT 矢量的 I 分量测量值
POINT.J
POINT.K

POINT.TI POINT 矢量的 I 分量理论值
POINT.TJ
POINT.TK

FEAT1.TYP 特征类型（即圆、槽、锥）。可用于更改一般特征的类型 (Assign/ Gen1.TYP = Feat1.TYP)。

FEAT1.ALL 表示特征的所有元素。这对于复制信息到一般特征是很有用的。(Assign/ Gen1.ALL = Feat1.ALL)

曲面矢量

EDGE.SURFI

EDGE.SURFJ

EDGE.SURFK

EDGE.TSURFI

EDGE.TSURFJ

EDGE.TSURFK

角度矢量

CIR.ANGI

CIR.ANGJ

CIR.ANGK

CIR.TANGI

CIR.TANGJ

CIR.TANGK

半径

CIRCLE1.R

CIRCLE1.TR

CIRCLE1.RAD

CIRCLE1.TRAD

CIRCLE1.RADIUS

CIRCLE1.PR – 极坐标半径

CIRCLE1.TPR – 理论极半径

CIRCLE1.TRADIUS（仅第一个字符有效）

直径

CIRCLE1.D

CIRCLE1.TD

CIRCLE1.DIAM

CIRCLE1.TDIAM

CIRCLE1.DIAMETER

CIRCLE1.TDIAMETER（仅第一个字符有效）

角度

CONE.A

CONE.TA

CONE.ANG

CONE.TANG

CONE.ANGLE

CONE.TANGLE

CONE.PA – 极角

CONE.TPA – 理论极角（仅第一个字符有效）

长度

LINE.L

LINE.TL

LINE.LEN

LINE.TLEN

LINE.LENGTH

LINE.TLENGTH（仅第一个字符有效）

高度

CYLINDER.PH – 极高

CYLINDER.TPH – 理论极高

半径、角度、高度

POINT.RAH – 具有半径、角度及高度的测量值的点

POINT.TRAH – 具有半径、角度及高度的理论值的点

区域

BLOB1.AREA - 返回 Blob 特征的区域测量值。

BLOB1.TAREA - 返回 Blob 特征的区域理论值。

例如，`ASSIGN/V1=BLOB1.AREA` 返回 BLOB1 特征的区域测量值并将其赋值给 V1 变量。

当前，仅 Blob 特征适用于这些区域扩展名。有关 Blob 特征的信息，请参见 "PC-DMIS Vision" 文档中的 "Vision Blob" 主题。

尺寸有效的扩展名进行双精度类型引用示例

格式：<尺寸 ID>.<AXIS>.<尺寸元素> -> `DIM1.X.NOM`

DIM1.X.NOM	DIM1 X 轴位置的标称值
DIM1.X.MEAS	DIM1 X 轴位置的测量值
DIM1.X.MAX	DIM1的X轴位置的最大偏差
DIM1.X.MIN	DIM1的X轴位置的最小偏差
DIM1.X.PTOL	DIM1的X轴位置的正公差值
DIM1.X.MTOL	DIM1的X轴位置的负公差值
DIM1.X.DEV	DIM1的X轴位置的偏差
DIM1.X.OUTTOL	DIM1的X轴位置的超差值
DIM1.Y.NOM	DIM1的Y轴位置的标称值
DIM1.Z.DEV	DIM1的Z轴位置的偏差
DIM3.PA.MEAS	DIM3的极角位置的测定值

DIM4.M.PTOL	DIM4 的 M 轴的正公差值
*DIM4.PTOL	DIM4的M轴的正公差值（参见下面“有效轴”的备注）。
DIM5.BTOL	DIM5为位置的公差补偿值。

有效轴：

X, Y, Z, D, R, A, T, V, L, PR, PA, M, PD, RS, RT, S, H, DD, DF, TP

注：
*只定义了一条轴的尺寸（如圆度、同心度等）可忽略轴的限定词。若使用了轴的限定词，注意，所有类型的尺寸（仅有一条轴）均使用 M 轴限定词，但 2D 和 3D
角度尺寸除外，这两种尺寸使用的是 A 轴限定词。

坐标系有效的扩展名进行双精度引用的示例

格式：<坐标系 ID>.<坐标轴或原点>.<坐标轴或原点分量> -> A1.ORIGIN.X

A1.ORIGIN.X	坐标系A1原点的X实测值
A2.ORIGIN.Y	坐标系A2原点的Y实测值
A1.ORIGIN.Z	坐标系A1原点的Z实测值
A1.XAXIS.I	坐标系A1的X轴实测矢量I
A1.YAXIS.J	坐标系A1的Y轴的实测矢量J
A1.ZAXIS.K	坐标系A1的Z轴的实测矢量K
A1.CORIGIN.X	坐标系A1的原点的X分量（基于理论数据，C代表CAD）
A1.CXAXIS.J	坐标系A1的X轴的J分量（基于理论数据，C代表CAD）

点类型的引用

可以使用以下引用表达式：

特征有效扩展名进行点类型引用的示例

格式：<特征 ID>.<扩展名> -> CIRCLE1.XYZ

CIRCLE1.XYZ	CIRCLE1 的测量质心
CIRCLE1.TXYZ	CIRCLE1 的理论质心
LINE1.SXYZ	LINE1 的测量起点
LINE1.TSXYZ	LINE1 的理论起点
LINE1.EXYZ	Measured end point of LINE1
LINE1.TEXYZ	LINE1 的理论终点
CIRCLE1.IJK	CIRCLE1 的测量矢量
CIRCLE1.TIJK	CIRCLE1 的理论矢量
EDGE.SURFIJK	EDGE 的测量曲面矢量

EDGE.TSURFIJK	EDGE 的理论曲面矢量
AUTOCIR1.ANGIJK	AUTOCIR1 的测量角度矢量
AUTOCIR1.TANGIJK	AUTOCIR1 的理论角度矢量

坐标系有效的扩展名进行点类型引用的示例

格式：<坐标系 ID>.<坐标轴或原点> -> A1.XAXIS

A1.ORIGIN	坐标系 A1 的测量原点
A1.XAXIS	坐标系 A1 的测量 X 轴
A1.YAXIS	坐标系 A1 的测量 Y 轴
A1.ZAXIS	坐标系 A1 的测量 Z 轴
A1.CORIGIN	坐标系 A1 的理论原点
A1.CXAXIS	坐标系 A1 的理论 X 轴
A1.CYAXIS	坐标系 A1 的理论 Y 轴
A1.CZAXIS	坐标系 A1 的理论 Z 轴

字符串类型的引用

注释引用是唯一属于字符串类型的对象类型。只有INPUT注释或YES/NO注释可以引用。这些注释类型拥有可以用来标识注释的ID。

格式：<注释 ID>.INPUT -> C1.INPUT

C1.INPUT - 注释 C1 的（操作员）的输入值

YES/NO注释类型根据PC-DMIS当前的语言将输入设置为相应的是或否字符串。在英文版的PC-DMIS中，如果操作者按yes按钮，字符串将设置为“YES”，如果操作者按no按钮，字符串将设置为“NO”。在比较字符串以测试“YES”或“NO”时区分大小写。因此，即使YES/NO注释输入设置为“YES”或“NO”，比较“yes”或“no”也会不匹配。

变量

变量可以有七个操作数类型：整数、实数、字符串、点、特征指针、数组或函数。变量通过 ASSIGN 语句存在和接收到它们的值和类型。变量 ID
可以是任何不以数字开头的数字字母字符串。变量 ID
中还可以使用下划线，只要下划线不是第一个字符。

变量值在执行之间保存。这意味着如果例程停止后重新启动，执行停止时的变量值与重新开始执行时的变量值相同。

注：如果“编辑”窗口为活动窗口，当光标位于字段中时，PC-DMIS 将指出变量的当前值。在执行过程中，变量值将依据执行的程序流而更改。将鼠标指针放在所需变量上，查看其当前值。

```
ASSIGN/ V1 = 2.2+2
```
变量 V1 为一个值为 4.2 的实数。

```
ASSIGN/ VAR1 = CIRCLE1.X
```
变量 VAR1 为一个实数，其值等于赋值时 CIRCLE1.X 的测量值。

```
ASSIGN/ MYVAR = LINE1.XYZ
```
变量 MYVAR 为一个点，其值等于赋值时 LINE1 的测量质心。

```
ASSIGN/ SVAR = "Hello World"
```
变量 SVAR 为字符串，值为 "Hello World"

在这些例子中，变量已经赋值。一旦为变量赋值，变量可以作为任何表达式字段的操作数使用。

在数值字段中使用 V1 的示例：

```
ASSIGN/V1 = 1/3
PREHIT / V1
```

此处，V1 作为逼近距离命令的逼近距离值。

注：由于表达式可以在大多数可编辑字段中使用，因此以下表达式也是有效的，并且具有相同的作用：PREHIT / 1/3。

点类型的变量的分量可以分别使用用于引用的点扩展名表示法进行引用。

```
ASSIGN/ V1 = MPOINT(3, 4, 5)
```
V1 是值为 3, 4, 5 的点型

```
ASSIGN/ XVAR = V1.X
```
XVAR 是值为 3 的双精度型

```
ASSIGN/ YVAR = V1.Y
```
YVAR 是值为 4 的双精度型

```
ASSIGN/ IVAR = V1.I
```
IVAR 是值为 3 的双精度型

```
ASSIGN/ REDUNVAR = V1.XYZ
```
REDUNVAR 是值为 3, 4, 5 的点型

以下扩展名是等价的。提供两种扩展名是为了解释测量例程中表达式的含义。

如果 V1 属于点类型。

V1.X 与 V1.I
相同 V1.Y 与 V1.J
相同 V1.Z 与 V1.K 相同

V1.XYZ 与 V1.IJK 和没有扩展名的 V1 相同。

如果字符串类型的变量的字符串值等于特征 ID 的名称，该变量可以作为引用对象使用。

示例：

```
ASSIGN/ V1 = "CIRCLE1"
```

如果存在名为 CIRCLE1 的特征，以下操作数可以是有效的。

V1.X - CIRCLE1 的 X 测量值
V1.TX - CIRCLE1 的 X 理论值
V1.Diameter - CIRCLE1 的测量直径
V1.Radius - CIRCLE1 的测量半径

字符串变量可以使用的间接引用仅限于同一级别的间接引用。以下引用无效。

```
ASSIGN/ V1 = "CIRCLE1"
ASSIGN/ V2 = "V1"
```

V2.X - 求值结果为 0 而非 CIRCLE1.X 的当前测量值。

注：参考 V2.X
不会以红色文本标记为错误，尽管它上面的一个表达式将其设置为字符串型。不标记为错误的原因就是，执行前测量例程的执行流未知。

不过，如果使用大括号，以下代码有效：

```
ASSIGN/ V1 = {CIRCLE1}
ASSIGN/ V2 = {V1}
```

V2.X - 将获得 CIRCLE1.X 的值。

参见下例：

```
ASSIGN/ V1 = "CIRCLE1"
ASSIGN/ V2 = "V1"
IF / CIRCLE1.X > CIRCLE1.TX, GOTO, L2
L1 = LABEL /
ASSIGN/ V3 = V2.X
GOTO / LABEL, L3
L2 = LABEL /
ASSIGN/ V2 = MPOINT( 2, 5, 7)
GOTO / LABEL, L1
L3 = LABEL /
```

如果例程执行过程中，CIRCLE1.X 值大于 CIRCLE1.TX 值，则表达式 V2.X 将有效且等于 2。否则，表达式 V2.X 等于 0，因为 V2 值在赋值给 V3 时为字符串 "V1"。在这些情况下，应由零件程序员来确保表达式将按预期操作。

赋值语句的其他说明：几乎所有特征参照均可在赋值语句的左侧使用，以为特征的测量数据或理论数据赋值。但矢量的 I、J、K
分量除外。要为矢量赋值，必须采用结果为点的表达式，为整个矢量赋值。矢量数据在输入特征的矢量数据成员时执行规范化。

示例：

```
ASSIGN/ CIRCLE1.I = 2 - 无效
ASSIGN/ CIRCLE1.IJK = MPOINT(2, 0, 0)-有效（矢量规范化为 1, 0, 0）
```

有关在尺寸中使用变量的信息，请参见“使用传统尺寸”一章中的“尺寸变量”主题。

结构

您可使用名为*结构*的变量类型，对变量进行扩展，从而识别该变量的子元素。请参考下例：

示例：

```
ASSIGN/V1.HEIGHT = 6
ASSIGN/V1.WIDTH = 4.3
ASSIGN/V1.MODE = "CIRCULAR"
ASSIGN/V1.POINT = MPOINT(100.3, 37.5, 63.1)
```

在该示例中：

- V1 为结构。
- HEIGHT、WIDTH、MODE 和 POINT 为该结构的子元素。

结构的规则

- 与变量类似，结构也不需要声明。
- 结构的子元素可以是以下任意变量类型：
  - 整数
  - 双精度
  - 点
  - 特征指针
  - 功能
  - 数组
  - 结构

例如，包含的结构元素可以是数组，包含的数组元素也可以是结构。因此，以下表达式示例有效：

示例

```
ASSIGN/CAR.LEFTSIDE.DOOR[2].QUADRANT[3].JOINT[5].HIT [4] = MPOINT(558.89, 910.12, 42.45)

COMMENT/OPER,“当前 Z 位置：”+ CAR.LEFTSIDE.DOOR[2].QUADRANT[3].JOINT[5].HIT[4].Z

ASSIGN/CURRENTJOINT = LEFTSIDE.DOOR[2].QUADRANT[3].JOINT[5]

COMMENT/OPER,“下一测点：”+ CURRENTJOINT.HIT[4]
```

变量类型为点的结构

如果变量为点型，用户仍可使用 .X、.Y、.Z、.I、.J 及 .K
扩展名来获取点的各项。用户也可在结构中使用本例的任何一种扩展名，而不必强制将它们作为点元素使用。

示例：

```
ASSIGN/V1.X = "Some string"
ASSIGN/V1.Y = ARRAY(1,3,5,9,7)
ASSIGN/V1.Z = MPOINT(3,5,7)
```

COMMENT/REPT,V1.X	输出为“Some string”
COMMENT/REPT,V1.Y[2]	输出为3，数组的第二个元素。
COMMENT/REPT,V1.Z.Y	输出为 5，即 MPOINT 的 Y 值。

通过将结构与 PC-DMIS 表达式语言的函数功能组合使用，可以获得动态结构引用，如下所示：

（示例续）

```
ASSIGN/DYNAMICSTRUCT = FUNCTION((X,Y), X.Y)
C1 =COMMENT/INPUT,请输入项目
ASSIGN/TESTSTR = C1.INPUT
ASSIGN/FRONT = LEFT(TESTSTR, INDEX(TESTSTR, ".")-1)
ASSIGN/BACK = MID(TESTSTR, INDEX(TESTSTR, "."))
ASSIGN/RESULT = DYNAMICSTRUCT(FRONT, BACK)
```

示例的这一部分要求您输入一个变量参考，并在第一个 "." 处分解参考，然后运用函数 DYNAMICSTRUCT 将 RESULT 赋值为与该参照相等。

因此，若为变量 C1.INPUT 键入 V1.Y[4]，RESULT 将为 9（数组的第四个元素被赋值给 V1.Y）。

表达式的学习时间评估已增强，可以准确显示结构或数组的所有元素。

指针

指针就是大家知道的"特征指针"。详细信息请参见术语表中的“特征指针”。

指针提供了一种使用呼叫子命令通过变量参考特征或传递对象的途径。指针类似于通过字符串名称的间接物。然而，使用指针时的益处就在于子程序。指针与字符串不同，在作为子程序的参数进行传入时，允许间接修改子程序指向的对象。复杂的表达式中不能使用指标。如在复杂的表达式中使用了指标，则指标得到的值为零。

参见下例。

指针使用示例：

- ASSIGN/ V1 = {CIR1}

  V1 现在是指向 CIR1 的指针。

- `ASSIGN/ DIST = DOUBLE(V1.XYZ)`

  Dist = CIR1 到原点的距离。

你也可以将表达式放在大括号之间得到一个特征指针。下面的例子都可以正确的得到特征CIR1的指针：

- `ASSIGN/FEATCOUNT = 1`

  `ASSIGN/V1={"CIR" + FEATCOUNT}`

  将表达式“CIR1”赋给 V1。

- `ASSIGN/V2="CIR1"`

  `ASSIGN/V3={V2}`

  将变量 V2 的表达式 "CIR1" 分配给变量 V3

- `C1=COMMENT/INPUT，请键入特征名称。`

  `ASSIGN/V4={C1.INPUT}`

  取 C1.INPUT 的特征名称并将其放入变量 V4 中。

子例程示例：

在调用例程中：

```
CS1 = 调用子程序/SUB.PRG,CHANGEX, {CIR1}
```

在例程中：

```
GEN1 = GENERIC/FEATURE
SUBROUTINE/CHANGEX, ARG1 = {GEN1}
```

（当CIR1传入时，将取代 GEN1）

```
ARG1.X = 5
```

（将CIR1的X测定值设置为5）

```
终止/子程序
```

复合表达式示例：

`赋值/V1 = {CIR1} + 2`

{CIR1}的值为0，所以整个表达式的结果为2。

数组

有三种可用的数组：特征数组、触测数组和变量数组。

重要：尽管多尺寸数组在软件中显示为多尺寸，但在将数组置于 ARRAY INDICES 命令前，仍可仅将其作为单尺寸数组使用（参见“数组索引对象：”主题）。

特征数组

在例程执行过程中多次测量某项特征时，会自动创建一个特征数组。特征数组的元素数与执行特征的次数相等。

示例：如果测量的圆对象处于 while 循环中，该循环执行了五次，则生成的数组会有五个测量的圆。如果测量的圆的 ID 为 "Circle1"，则可使用数组表达式访问测量的圆对象的各个实例。可使用方括号指示所需的实例。

`Assign/V1 = Circle1[3].X`
V1 被分配了 Circle1 第三个实例的 X 测量值。

注：尽管给定的特征有特征数组，但在对该特征的引用中却并未使用数组标志，使用的是最近实例。在上例中，`Circle1.X` 与 `Circle1[5].X` 相同，因为第五个实例即该对象最近的实例。

在数组表达式的方括号中，可使用表达式。因此 `Circle1[3].X` 与 `Circle1[2+1].X` 相等。下例使用了一个循环来打印输出上例中测量的五个圆的质心。

示例：

```
Assign/V1 = 1
While/V1 < 6
Comment/Rept, "Centroid of instance #" + V1 + " of Circle1: " + Circle1[V1].XYZ
Assign/V1 = V1 + 1
End/While
```

上述示例可能的输出如下所示：

Circle1的1#实例的质心：3.4,2.6,1.43

Circle1的2#实例的质心：4.4,3.6,2.43

圆1的实例#3的质心： 5.4, 4.6, 3.43

Circle1的4#实例的质心：6.4,5.6,4.43

Circle1的5#实例的质心：7.5,6.6,5.43

在给定的执行运行中执行了多次的尺寸与坐标系，同样也有数组。因此尺寸 "Dim1" 至少执行了两次，坐标系 "Align1" 至少执行了四次，则 `Dim1[2].Nom` 与 `Align1[4].Origin` 将可用。

若某特征数组引用超出界限（如用户请求 `Circle1[2.5]` 或 `> Circle1["Hello, World"]`），将返回其上限或下限项目。若 Circle1 有 3 个实例，则 `Circle1[4]` 及以上将返回 `Circle1[3]` 的值，而 `Circle[0]` 及以下将返回 `Circle1[1]` 的值。介于方括号之间的所有表达式将强制为整数，因此 2.5 将变为 2，而 "Hello World" 则变为 0。

数组指数对象

默认情况下，特征数组总是一维数组。通过使用数组指数对象可以方便的用多维数组表示特征数组。

数组指数对象允许指定多维数组的上限和下限。

- 通过为第一个维设置上限和下限，将创建一个二维数组，第一个维有限制，第二个维无限制。
- 通过为数组的前两个维设置上限和下限，将创建一个三维数组。最后一个维总是无限制。

示例：
特征 F1 位于 WHILE 嵌套循环中。内部的 WHILE 循环执行五次，外部的 WHILE 循环执行三次。执行完成时，F1 已执行了 15 次，所以 F15 存在 15 个实例。

请看以下测量例程段示例：

```
ARRAY_INDICES/1..5,..
ASSIGN/V1 = 1
WHILE/V1<=3
ASSIGN/V2 = 1
WHILE/V2<=5
F1=FEAT/POINT,RECT
THEO/V2,V1,0,0,0,1
ACTL/1,1,0,0,0,1
MEAS/POINT,1
HIT/BASIC,V2,V1,0,0,0,1,1,1,0
ENDMEAS/
ASSIGN/V2 = V2+1
COMMENT/REPT,"F1 的位置[" + V2 + "," + V1 + "] :" + F1[V2,V1].XYZ
END_WHILE/
ASSIGN/V1 = V1+1
END_WHILE/
```

该代码段创建一个 3 X 5 网格的 15 个测量点。

数组命令中特征的第一维序列是有限的，在1和5之间。因而在检验报告上显示的不是 F1[1] – F1[15]，而是 F1[1, 1] – F1[5, 3]，更符合特征的布局。注意，注释使用两维数组语法，也参考了此特征数组。

要在测量例程中插入 array_indices 对象：

1. 使用键盘，在“编辑”窗口的空行上键入“**数组**”。
2. 按键盘上的 Tab 键。

注：如果**显示特征数组的括号**复选框没有选中，则显示特征时不会带有用括号括住的名称。请参考“设置首选项”一章中的“显示特征数组的括号”。

测点数组

给定特征的测点可以作为数组，并且使用 form <FeatID>.Hit[<Array Expression>].<Extension> 或 form <FeatID>.RawHit[<Array Expression>].<Extension>

的数组语法可以访问数组。打开测头补偿时，测点会返回测头补偿的数据。RawHit 始终返回无补偿的数据。有效的扩展名有
X、Y、Z、I、J、K、TX、TY、TZ、TI、TJ、TK、XYZ、TXYZ、IJK 和 TIJK

```
Circle1.Hit[1].XYZ
```
"Circle1" 的测点 1 的测量的质心（测头补偿）。

```
Circle1.Hit[2].IJK
```
"Circle1" 的测点 2 的测量矢量

无论实际触测点是否在编辑窗口中显示，所有含触测点的对象的触测点数据都可以被引用。因此，可以获得扫描和自动特征的触测点。

以下主题描述用于查找扫描最小值或最大值的其他数组函数：

将某个范围的测点分配给数组

你也可以使用以下语法将触测点赋值到数组：

<Feature Id>.<Hittype>[<Startnum>..<Endnum>].<Extension>

其中：

<Feature Id>是特征名

<Hittype> 可以是补偿的数据的 "HIT" 一词，也可以是无补偿的数据的 "RAWHIT" 一词。如测头补偿被关闭，则返回的将是中是无补偿的值。

<Startnum>是确定触测点范围的第一个序数的表达式。

<Endnum> 是确定触测点范围的第二个序数的表达式。

<Extension>确定数据的类型。可用的扩展名包含以下数据类型的实测值或理论值：

- X – 测点的 X 测量值
- Y – 测点的 Y 测量值
- Z – 测点的 Z 测量值
- XYZ – 测点的 XYZ 测量值
- I – 测点的 I 测量值
- J – 测点的 J 测量值
- K – 测点的 K 测量值
- IJK – 测点的 IJK 测量值

- TX – 测点的 X 理论值
- TY – 测点的 Y 理论值

- TZ – 测点的 Z 理论值
- TXYZ – 测点的 XYZ 理论值
- TI – 测点的 I 理论值
- TJ – 测点的 J 理论值
- TK – 测点的 K 理论值
- TIJK – 测点的 IJK 理论值

例如：

```
ASSIGN/V1 = SCAN1.HIT[1..10].X
```
V1 被分配给一个有 10 个值的数组，这 10 个值是 SCAN1 的前 10 个测点的的 X 测量值。

```
ASSIGN/V2 = SCAN1.HIT[1..SCAN1.NUMHITS].XYZ
```
V2 被分配给一个扫描的测点质心点的数组。

数组排序

PC-DMIS允许你升序或降序进行数组排序。下面的两个表达式获取一个数组并返回一个经过排序的数组。

若要按*升*序排列，使用：
```
SORTUP(<数组>)
```

若要按*降*序排列，使用：
```
SORTDOWN(<数组>)
```

例如:

```
ASSIGN/V1 = ARRAY(5,8,3,9,2,6,1,7)
```
V1 被分配数组 "5,8,3,9,2,6,1,7"

```
ASSIGN/V2 = SORTUP(V1)
```
V2 按升序排序的数组值："1,2,3,5,6,7,8,9"

```
ASSIGN/V3 = SORTDOWN(V1)
```
V3 按降序排序的数组值："9,8,7,6,5,3,2,1"

返回最大或最小的数组索引值：

使用以下函数可以输入一个数组到函数中返回元素的最大值或最小值的索引值：

要返回具有*最大*值的元素的索引值，使用：

```
MAXINDEX(<数组>)
```

要返回具有*最小*值的元素的索引值，使用：

```
MININDEX(<数组>)
```

例如:

```
ASSIGN/V1 = ARRAY(5,8,3,9,2,6,1,7)
```
V1 被分配数组 "5,8,3,9,2,6,1,7"

```
ASSIGN/V2 = MAXINDEX(V1)
```
V2 将保留数组的索引值 4。该数组元素的实际值为 9

```
ASSIGN/V3 = MININDEX(V1)
```
V3 将保留数组的索引值 7。该数组元素的实际值为 1

可以使用返回的索引值得到实际的数组元素值。

返回排序后的数组索引值

使用以下函数可以输入一个数组到一个函数按升序或降序对数组的值进行排序后再返回索引值：

要按照数组值从大到小排列的顺序返回数组的索引值，使用：
```
MAXINDICES(<数组>)
```

要按照数组值从小到大排列的顺序返回数组的索引值，使用：
```
MININDICES(<数组>)
```

例如:

```
ASSIGN/V1 = ARRAY(4,8,2,9,5,7)
```
V1 被分配数组 "4,8,2,9,5,7"

```
ASSIGN/V2 = MAXINDICES(V1)
```
V2 将保留值为 "4,2,6,5,1,3" 的数组

```
ASSIGN/V3 = MININDICES(V1)
```
V3 将保留值为 "3,1,5,6,2,4" 的数组

使用数组函数查找扫描的最小和最大点的示例

以上讨论的数组函数示例用于便捷查找扫描中的最大和最小点。

要确定 SCAN1 的具有最大 X 测量值的点的尺寸，可使用下面的表达式：

```
ASSIGN/MAXPTINDEX = MAXINDEX(SCAN1.HIT[1..SCAN1.NUMHITS].X)
D1 = LOCATION OF FEATURE SCAN1.HIT[MAXPTINDEX]
```

查找SCN2中Z轴最高的三个点使用以下表达式：

```
ASSIGN/MI = MAXINDICES(SCAN2.HIT[1..SCAN2.NUMHITS].Z)

ASSIGN/THREEPOINTS = ARRAY(SCAN2.HIT[MI[1]].XYZ,
SCAN2.HIT(MI[2]].XYZ, SCAN2.HIT[MI[3]].XYZ)
```

变量数组

变量数组不需要声明。当赋值语句右侧的表达式的值为数组或赋值语句的左侧引用变量数组中的元素时，变量数组通过赋值语句出现。

`Assign/V1 = Array(3, 4, 5, 6, 7)`
创建一个 5 元素的数组，并将其分配给 V1

`Assign/V2 = V1[3]`
为 V2 分配数组 V1 中的第三个元素：5

`Assign/V1[4] = 23`
将数组 V1 的第 4 个元素分配值 23

数组可以动态创建和分配。因此，可以在赋值语句的左侧使用数组引用创建数组。

`Assign/V3[5] = 8`
动态创建数组，数组的第 5 个元素等于 8

在引用从未接收过值的数组元素时，数组表达式的值将为 0。

`Assign/V3[5] = 8`

`Assign/V4 = V3[5]`
V4 设置为等于值 8

`Assign/V5 = V3[6]`
如果从未设置过 V3 的第六个元素，则 V5 设置为等于 0。

与其它数组类型类似，表达式可以在方括号中使用。

`Assign/V3[5] = 8`

`Assign/V4 = V3[2+3]`
V4 设置为等于值 8

变量数组可以有多个维。

`Assign/V6 = Array( Array(4, 7, 2), Array(9, 2, 6))`
V6 被设为两个三维数组，其中 V6[1, 1] 等于 4，V6[1, 2] 等于 7，V6[1, 3] 等于 2，V6[2, 1] 等于 9，V6[2,2] 等于 2，V6[2,3] 等于 6。

`Assign/V7 = V6[2,1]`
V7 设置为等于值 9

变量数组的指数可以是负数：

```
Assign/V8[-3] = 5
```
数组 V8 的第 –3 个索引被设为 5。

数组赋值将替换以前的值：

```
Assign/V8 = "Hello"
```
变量 V8 等于字符串 "Hello"

```
Assign/V8[2] = 5
```
V8 不再是字符串型，而是数组型，该数组的第二个元素的值为 5。

```
Assign/V8 = 9
```
V8 不再是数组型，而是整数型的数值 9。

数组可以由多种类型组成：

```
Assign/V9 = Array("Hello", 3, 2.9, {FEAT1})
```
创建拥有 4 个元素的数组
V9。第一个元素为字符串，第二个元素为整数，第三个元素为实数，第四个元素为 FEAT1 的指针。

数组的尺寸可以增加，包含更多数组：

```
ASSIGN/V10=ARRAY(3,1,5)
ASSIGN/V10[LEN(V1)+1]=7
```
第一个语句创建了一个含有 3 个元素（3、1 和 5）的初始数组 V10。第二个语句然后将 V10 中的数组增加了一个元素，并为最后一个元素提供了一个值 7。

表达式运算符

以下是在PC-DMIS里可用到的基本运算符。

+ 加：*<Expression>* + *<Expression>*
将两个表达式相加。如为字符串，则把字符串连在一起。

- 减：*<Expression>* - *<Expression>*
第一个表达式减去第二个表达式。

***** 乘：*<Expression>* * *<Expression>*
将两个表达式相乘。

/ 除：*<Expression>* / *<Expression>*
第一个表达式除以第二个表达式。

^ 求幂：*<Expression>* ^ *<Expression>*
令第一个表达式成为第二个表达式的幂。

% 求模：*<Expression>* % <Expression>
返回一个表达式与另一表达式相除的余数。

- 加法逆元 *-<Expression>*
返回表达式的加法逆元数。

! 逻辑非：*!<Expression>*
返回表达式的逻辑非的值。

== 等于:*<Expression>* == *<Expression>*
如果表达式相等，则结果为 1。否则为 0。（两个等号用于区分赋值语句中的赋值运算符 =）。

<> 不等于：*<Expression>* <> *<Expression>*
如果表达式不等，结果为 1。否则为 0。

> 大于：*<Expression>* > *<Expression>*
如果第一个表达式大于第二个表达式，结果为 1。否则为 0。

>= 大于等于：*<Expression>* >= *<Expression>*
如果第一个表达式大于或等于第二个表达式，结果为 1。否则为 0。

< 小于：*<Expression>* < *<Expression>*
如果第一个表达式小于第二个表达式，结果为 1。否则为 0。

<= 小于等于：*<Expression>* <= *<Expression>*
如果第一个表达式小于或等于第二个表达式，结果为 1。否则为 0。

AND 逻辑与： *<Expression> AND <Expression>*
如果两个表达式均不为 0，则结果为 1。否则结果为 0。

OR 逻辑或：*<Expression> OR <Expression>* 如果两个表达式中的任意一个均不为 0，则值为 1。否则为 0。

() 括号： *(<Expression>)*
优先计算括号内的表达式的值。

优先级

表达式的求值优先级如下所示（优先级从高到低列出）。

最高优先级

- 操作数
- （一元减号）、!、()、函数（如 ABS、COS、STR、LEN、CROSS 等。）
- ^
- *, /, %
- +, -
- ==, <>, <, <=, >, >=
- AND
- 或

最低优先级

函数

函数是通过输入参数返回结果的PC-DMIS特殊的表达式或者用户定义的表达式。在表达式求值之前，参数将传入到表达式中。

函数列表

以下按字母顺序排列的列表包含有 PC-DMIS 的表达式语言可用的所有函数。

- ABS（数学）
- ACOS（数学）
- ANGLEBETWEEN（数学）
- ARCSEGMENTENDINDEX（其他）
- ARCSEGMENTSTARTINDEX（其他）
- ARRAY（数组）
- ASIN（数学）
- ATAN（数学）
- CHR（字符串）
- COS（数学）
- CROSS（点）
- DEG2RAD（数学）
- DELTA（点）
- DIST2D（指针）
- DIST3D（指针）
- DOT（点）
- ELEMENT（字符串）
- EOF（其他）
- EOL（其他）
- EQUAL（数组）
- EQUAL（字符串）
- EXP（数学）

- FORMAT（字符串）
- FUNCTION（函数）
- GETCOMMAND（指针）
- GETSETTING（字符串）
- GETTEXT（字符串）
- IF（其他）
- INDEX（字符串）
- ISIOCHANNELSET（其他）
- LEFT（字符串）
- LEN（数组）
- LEN（指针）
- LEN（字符串）
- LINESEGMENTENDINDEX（其他）
- LINESEGMENTSTARTINDEX（其他）
- LN（数学）
- LOG（数学）
- LOWERCASE（字符串）
- MAX（数组）
- MID（字符串）
- MIN（数组）
- MPOINT（点）
- ORD（字符串）
- PCDMISAPPLICATIONPATH（字符串）
- PCDMISUSERHIDDENDATAPATH（字符串）
- PCDMISUSERVISIBLEDATAPATH（字符串）
- PCDMISSYSTEMHIDDENDATAPATH（字符串）
- PCDMISSYSTEMVISIBLEDATAPATH（字符串）
- PCDMISSYSTEMREPORTINGPATH（字符串）
- PROBEDATA（其他）
- QUALTOOLDATA（其他）
- RAD2DEG（数学）
- RIGHT（字符串）
- ROUND（数学）
- SIN（数学）
- SQRT（数学）
- SYSTEMDATE（字符串）
- SYSTEMTIME（字符串）
- SYSTIME（字符串）
- TAN（数学）
- TUTORELEMENT（其他）
- UNIT（点）
- UPPERCASE（字符串）

字符串函数

以下函数使用文本字符串

CHR

CHR
字符转换：*CHR(<Integer>)*

返回一个字符串，由与 ASCII 十进制值对应的字符组成。

ELEMENT

ELEMENT
子字符串位置分隔：*ELEMENT(<Integer>, <String1>, <String2>)*

返回字符串 2 的第 n 个子字符串（元素），使用字符串 1 作为分隔文本分隔字符串 2 中的各元素。例如，如果字符串 2 为“6, 12, 8, 4, 5”，字符串 1 则为“,”， 可以使用 element 命令单独检索 5 个元素为“6”、“12”、“8”、“4”和“5”。

EQUAL

EQUAL
不区分大小写的字符串比较：*EQUAL(<String>, <String>)*

比较两个字符串（忽略大小写），确定两个字符串是否相同。如果字符串相同，则返回整数 1，如果不相同，则返回 0。

FORMAT

FORMAT
格式：FORMAT(<String>,<Integer,double,or point>)

这个函数获得两个表达式返回一个格式化的字符串，与在C++中使用*sprintf*函数相似

- 表达式1是一个*字符串*类型，包含一个或三个指定的格式。如果该字符串是其他类型，表达式求值程序会尝试将其强制转换为字符串。如果表达式2是一个整数或双精度实数字符串应该包含一个指定格式，如果表达式2是一个点类型字符串应该包含三个指定格式(参见以下段落)。
- 表达式2的类型应为*integer*,*double*,或*point*。如果使用了其他类型，表达式的值为0。

FORMAT 函数的格式限定符：

指定的格式应该与 C++ 程序语言中使用的 *sprintf* 函数指定的格式相同。

指定格式包含可选和必须的字段，有以下语法：

%[标记] [宽度] [.精度] 类型

指定格式的每个字段为单一字符或表示特殊格式的数字。一个比较简单的格式只使用百分比符号和一个类型符（比如%d）。如果百分比符号后面再跟随一个百分比符号表示该格式字段没有实际意义。比如，使用%%输出一个百分比符号。

标记，宽度，和精度字段可选项位置在类型之前，用于控制其他格式。如下所述：

标记
这些*可选字符*控制符号、空白、小数点和八进制/十六进制前缀的对齐和打印。格式限定符中可以出现多个标记。

以下为可用的标记：

–
含义：在给定字段宽度内左对齐。
默认：右对齐。

+
含义：如果输出值为有正负之分的类型，给输出值加前缀符（+ 或 –）。
默认值：只有负号表示赋值 (-)。

0
含义： 如果宽度提前设定为 0，当宽度达到最小时将添加 0 值。如果 0 和 - 号出现，0 忽略。如果 0 被指定为整数格式（i,u,x,X,o,d），0 被忽略。
默认值：不填充。

空白 (' ')
含义：如果输出值有正负之分并且是正的，在输出值前加上一个空格；如果既加了空格又加了 + 号，则忽略空格。
默认：不出现空格。

#
含义 1：当使用 o、x 或 X 时，# 号表示为任意非 0
输出值分别添加前缀 0、0x 或 0X。
默认值 1：没有前缀。

含义2:当与e,E或者f类型时，#号强制在所有输出值上加上小数点。
默认2： 如果有数字跟着的时候会出现小数点。

含义3：当使用g或G格式，#号强制输出值加上小数点并截断拖尾的0。

默认3：如果#号后是阿拉伯数字，将出现小数点。后面的 0 将被删除。
注释；当与d,i或者u一起使用时忽略。

宽度
第二个可选字段，或自变量，控制输出的最小字符数。它是非负的十进制整数。

- 如果输出值的字符数小于指定的宽度，值的左边或右边将加上空格—取决于是否指定了–标记（左对齐）—直到填充到最小宽度。
- 如果宽度的前缀是 0，将填充 0 直到达到最小宽度（对左对齐的数字无效）。
- 指定宽度不会引起值被截断。如果输出值的长度大于指定的宽度，或没有给定宽度，将输出值的所有字符（受以下精度格式影响）。

精度
这第三可选字段或者自变量，控制了被打印的字符的个数，小数点数或者是有效数字的个数。与宽度指定不同，精度指定可能删除输出值的部分数据或对浮点数进行取整。它是非负的十进制整数，以句号结束。

类型
这是必需的字符，可以是整数、双精度实数，或点。可用的类型列表如下：

d - 有符号的十进制整数

i - 有符号的十进制整数

o - 无符号八进制整数

u - 无符号十进制整数

x - 无符号十六进制整数，使用 "abcdef"

X - 无符号十六进制整数，使用 "ABCDEF"

e - 双精度指数 [-]d.dddd e [正负号]ddd

E - 用 E 表示指数，其余与 e 相同

f - 双精度，格式为 [-]dddd.dddd

g - 在 e 或 f 格式基础上使格式变得更紧凑

G - 与 g 相同，但用 E 表示指数

FORMAT 示例：

该示例在测量例程中使用 FORMAT 函数的多种语句：

`ASSIGN/V1 = PROBEDATA("OFFSET")`	V1 表示为点类型的当前测头偏置。使用举这个例子的测量例程中的数值，V1 为：<-1.8898, 1.8898, 5.704>
`ASSIGN/V3 = FORMAT("%.5f, %.5f, %.5f", V1)`	V3 为字符串型。这个字符串是使用点类型变量 V1 格式化来的。V3 现在是：-1.88976, 1.88976, 5.70403
`ASSIGN/V4 = 1.123456789`	V4 为双精度.
`ASSIGN/V5 = FORMAT("%.5f ", V4) + FORMAT("%.6f ", V4) + FORMAT("%.7f ", V4) + FORMAT("%.8f", V4)`	V5 为字符串型，值为：1.12346 1.123457 1.1234568 1.12345679
`ASSIGN/V6A =“V4 的值为：”+ FORMAT("%.8f", V4)`	V6A 为字符串型，值为：V4 的值为：1.12345679
`ASSIGN/V6B = FORMAT（“V4 的值为：%.8f", V4）`	表达式的结果和V6A相同。
`ASSIGN/V7 = 4444`	除非强制为整型数值，否则所有数值都为双精度，V7也为双精度类型。
`ASSIGN/V8 = FORMAT("%o", INT(V7))`	V8为字符串，值为:10534V8为字符串，值为: 10534
`ASSIGN/V9 = FORMAT("%u", INT(-1))`	V9 为字符串型，值为：4294967295
`ASSIGN/V10 = FORMAT("%x", INT(2143))`	V10为字符串，值为:85f V10为字符串，值为: 85f
`ASSIGN/V11 = FORMAT("%X", INT(9567))`	V11为字符串型，值为:255FV11为字符串型，值为: 255F
`ASSIGN/V12 = FORMAT("%e", 0.0005432)`	V12 为字符串型，值为：5.432000e-004
`ASSIGN/V13 = FORMAT("%E", 145.3421)`	V13 为字符串型，值为：1.453421E+002
`ASSIGN/V14 = FORMAT(",%6d,", INT(1))`	V14为字符串，值为:,1,V14为字符串，值为: ,1,

`ASSIGN/V15 = FORMAT(",%-6d,", INT(1))`	V15 为字符串型，值为： ,1 ,

GETSETTING

GETSETTING
根据所插入的字符串参数返回 PC-DMIS 的不同设置。

GETSETTING(<*String*>)

可以使用的字符串参数：

- “DCC 模式” – 如果 PC-DMIS 在 DCC 模式下，返回 1，否则返回 0。
- “手动模式” – 如果 PC-DMIS 在手动模式下，则返回 1，否则返回 0。
- “当前坐标系” – 返回当前坐标系的字符串。
- “当前工作面” – 返回当前工作面的字符串。
- “工作面值” – 返回当前工作面的数值。
- “预测点” – 返回当前预测点值作为双精度型的精确度数。
- “回退” – 返回当前回退值作为双精度型的精确度数。
- “检查” – 返回当前检察值作为双精度型的精确度数。
- “接触速度” – 返回当前接触速度值作为双精度型的精确度数。
- “移动速度” – 返回当前移动速度值作为双精度型的精确度数。
- “飞行模式” – 如果 PC-DMIS 使用飞行模式，返回 1，否则返回 0。
- “有 Ph9” – 如果有 Ph9/Ph10，则返回 1，否则返回 0。
- “手动 CMM” – 如果 CMM 为手动 CMM，则返回 1，否则返回 0。
- "LangStr(<数值或 ID>)" – 返回资源 ID 号或以下 ID 中某个 ID 的当前语言形式的 PC-DMIS 资源的字符串：
  "Yes"、"No"、"Oper"、"Rept"、"Input"、"Doc"、"YesNo"、"Readout"、"Internal"、"External"、"Rect "、"Polr
  "、"Out"、"In"、"Least_Sqr"、"Min_Sep"、"Max_Insc"、"Min_CircSc"、"Fixed_Rad"、"Workplane"、"Xaxis"、"YAxis"、"ZAxis"、"Xplus"、"Xminus"、"YPlus"、"YMinus"、"ZPlus"、"ZMinus"、"Point"、"Plane"、"Line"、"Circle"、"Sphere"、"Cylinder"、"Round_Slot"、"Square_slot"、"Cone" 或 "None"。

  如果是个正数，PC-DMIS从resource.dll文件中取出字符串。若使用的是负数，则 PC-DMIS 从 strings.dll 文件（字符串表）取出字符串。
- “展开的薄壁件” –
  如果在**设置选项**对话框中选择**显示展开的薄壁件选项**复选框，返回 1，否则返回 0。
- "LastHitMove(X)" – 返回最近的 HIT /BASIC 或 MOVE/POINT 命令的 X 值。要使用该函数，PC-DMIS 须处于 DCC 模式下。

- "LastHitMove(Y)" – 返回最近的 HIT /BASIC 或 MOVE/POINT 命令的 Y 值。要使用该函数，PC-DMIS 须处于 DCC 模式下。
- "LastHitMove(Z)" – 返回最近的 HIT /BASIC 或 MOVE/POINT 命令的 Z 值。要使用该函数，PC-DMIS 须处于 DCC 模式下。

要判断 PC-DMIS 为手动还是 DCC 模式，可参考以下使用 GETSETTING 函数的示例：

示例：

`ASSIGN/DCCMODEVAR = GETSETTING("DCC Mode")` - 如果 PC-DMIS 为 DCC 模式，DCCMODEVAR 的值为 1，否则值为 0。

`ASSIGN/MANMODEVAR = GETSETTING（“手动模式”）` - 如果 PC-DMIS 为手动模式，则 MANMODEVAR 的值为 1，否则值为 0。

要判断当前的工作平面，可使用以下示例：

示例：

`ASSIGN/WORKPLANE_ID = GETSETTING（“当前工作面”)` - 赋值给变量 WORKPLANE_ID 当前工作平面的字符串值（ZPLUS、ZMINUS 等）。

`ASSIGN/WORKPLANE_VALUE = GETSETTING（“工作面值”）` - 赋值给变量 WORKPLANE_VALUE 描述当前工作平面的数值。工作平面的值如下：ZPLUS = 0，ZMINUS = 3，XPLUS = 1，XMINUS = 4，YPLUS = 2 或 YMINUS = 5。

GETTEXT

GETTEXT
返回指定数据字段中的当前文本：`GETTEXT(<String 或 Integer>, <Integer>, <Pointer>)`

该函数有三个字段。

第一个字段 — 数据字段编号或说明

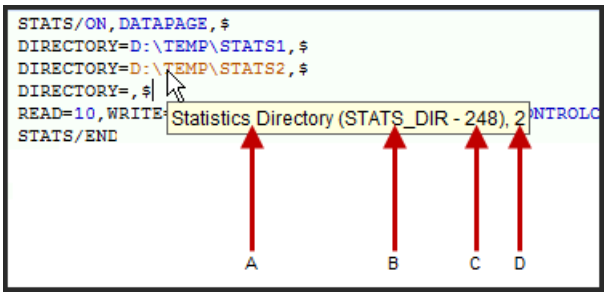
第一个字段可以是数据字段的一串说明，可在以下图像中的（A）项或数据字段编号，以下图像中（C）项中可以看到。

注：以下图像中的 (B) 项不在此功能中使用，但有时在自动或报告说明中使用。

要获取这些值，请执行以下步骤：

1. 使 PC-DMIS 处于命令模式。右击“编辑”窗口中的任意位置。屏幕上将出现一个快捷菜单。
2. 在快捷菜单中依次选择**更改弹出显示**和**数据类型信息**。
3. 将鼠标指针置于“编辑”窗口的数据字段上。将显示该数据项的类型说明、类型编号和类型索引。

注：不同语言的类型说明可能有所不同。若在使用不同语言运行的 PC-DMIS 上执行测量程序，则使用类型编号。



示例数据类型信息显示 (A) 类型说明(B) 类型字符串标识符，(C) 类型编号和 (D) 类型索引

第二个字段 — 类型索引

以上图像中所（D）指出的，第二个字段是类型索引。该字段一般为零，除非用户在同一个命令中有同样字段类型的更多实例，比如在以上图像中显示的多个目录字段。与第一个字段描述的方法一致可以获得这个字段正确的值。

第三个字段 — 命令指针

第三个字段是命令指针。它指向字段包含的文本指向的命令。该字段可以使用命令指针符号（即 {F15}）或在例子里的使用 GETCOMMAND 表达式。

示例：

ASSIGN/V1 = GETTEXT（“最佳拟合数学类型”、0、{F15}） - 此命令把 F15 中最佳拟合算法的勾选值赋给 V1。

ASSIGN/V2 = GETCOMMAND（“注释”、"TOP"、1） - V2 被赋值了一个指针，这个指针指向测量程序从头开始的第一个注释。

ASSIGN/V3 = GETTEXT（“注释类型”、1、V2） - V3 被指定“注释类型”切换字段中的值。如果测量程序中的第一个注释是向操作员显示的注释，则 V3 的值必须是字符串 "OPER"。

有关设置命令指针所用的 GETCOMMAND 表达式的信息，请参见“指针函数”。

GETPROGRAMINFO

GETPROGRAMINFO
根据以下传入的参数返回测量例程信息：`GETPROGRAMINFO(<String>, <Optional String>)`

此函数最多有两个字符串作为参数。对于大多数项目，仅需使用第一个参数。这些字符串字段不区分大小写。

第一个字段—字符串

第一个字段是详述返回信息的字符串输入。

DATE - 返回当前日期

TIME - 返回当前时间

PARTNAME - 返回测量例程标题中定义的零件名称

PARTPATH - 返回测量例程文件的完整路径

REVISION - 返回标题中定义的修订号

DRAWING - 如 REVISION 一样，也返回标题中定义的修订号

SERIALNUM - 返回标题中定义的序列号

STATSCOUNT - 返回当前统计计数

SEQNUM - 如 STATSCOUNT 一样，也返回当前统计计数

SHRINK - 返回全局缩放比例

NUMMEAS - 返回执行的尺寸数

NUMOOT - 返回执行的超出公差尺寸数

ELAPSEDTIME - 返回从执行开始起所用的时间

FILENAME - 返回测量例程的文件名 (.prg)

REPORTNAME - 返回当前输出文件名

TEMP - 返回可选第二个输入字符串的温度。参见下文“第二个字段 — 可选字符串”。

PRGVERSION - 返回测量例程文件中 PC-DMIS
修订号的字符串值。您可保存与特定版本兼容的测量例程文件。更多信息，请参见“使用基本文件选项”一章中的“另存为”。

PRGSCHEMA - 返回测量例程文件中 PC-DMIS 结构描述编号的整数。这是 PC-DMIS 所用的内部值，用于指示序列化的命令和选项。

PCDMISVERSION - 返回 PC-DMIS 软件的实际安装版本的字符串值。

第二个字段—可选字符串

第二个字段是可选字符串输入。仅在第一个输入字段中使用 TEMP
时才需要第二个字段。以下字符串来自“温度补偿”命令。更多信息，请参见“设置首选项”一章中的“温度补偿”。

TEMPP - 返回零件传感器的温度

TEMPX - 返回 X 轴传感器的温度

TEMPY - 返回 Y 轴传感器的温度

TEMPZ - 返回 Z 轴传感器的温度

REF_TEMP - 返回参考温度

HIGH_THRESHOLD - 返回高阈值温度

LOW_THRESHOLD - 返回低阈值温度

举例

```
$$ NO，此代码样本显示总尺寸数和超出公差尺寸数。
```

```
        ASSIGN/V1 = GETPROGRAMINFO("NUMMEAS")
        ASSIGN/V2 = GETPROGRAMINFO("NUMOOT")
        COMMENT/REPT
        “总尺寸：”+ V1
        “总超出公差：”+ V2
$$ NO，此代码样本返回 Z 轴传感器的温度。
        ASSIGN/V3 = GETPROGRAMINFO("TEMP", "TEMPZ")
        COMMENT/REPT
        “Z 轴温度：”+ V3
```

ELAPSEDEXECUTIONTIME

ELAPSEDEXECUTIONTIME
格式化的执行时间：*ELAPSEDEXECUTIONTIME()*

此函数返回自测量程序或精简程序开始执行起所消耗的时间。执行时间为执行 DCC 部分执行所用的时间；它不会添加由于用户需要注意而暂停的时间。（例如，当执行注释、出现消息或出现错误和停止执行时，可能会暂停执行。）时间将以 "hh:mm:ss" 格式返回。

通过向变量分配以下函数，可在测量程序或精简程序中的任何时候记录执行时间：

ASSIGN/V1=ELAPSEDEXECUTIONTIME()

INDEX

INDEX
子字符串位置：*INDEX(<String>, <String>)*

返回第二个字符串在第一个字符串中的位置。字符串的第一个字符是1。如果返回值是0，则表示子字符在字符串中没有找到。

有关此函数的示例，请参见“使用文件输入/输出”一章中的“读取行的示例代码”。

LASTEXECUTIONTIME

LASTEXECUTIONTIME
格式化的最后执行时间：*LASTEXECUTIONTIME()*

此函数返回 PC-DMIS 在<*测量程序的名称*>.MiniRoutines.xml 文件中记录和存储的上次执行时间。上次执行时间显示在**执行**对话框中。时间将以 "hh:mm:ss" 格式返回。

LEFT

LEFT
字符串左侧 n 个字符：*LEFT(<String>, <String>)*

返回第一个表达式指定的字符串中最左侧 n 个字符组成的字符串，n 由第二个表达式指定。第一个表达式类型强制为字符串，第二个表达式类型强制为整数。

有关此函数的示例，请参见“使用文件输入/输出”一章中的“读取行的示例代码”。

LEN

LEN
字符串长度：*LEN(<String>)*

返回字符串中的字符数。

LOWERCASE

LOWERCASE
创建小写字符串：*LOWERCASE(<String>)*

返回与该字符串等价的小写字符串。

MID

MID
字符串中间的 n 个字符：*MID(<String>, <Integer>, <Optional Integer>)*

返回由第一个参数指定的字符串中的字符组成的子字符串，从第二个参数指定的位置开始，长度为第三个参数指定的 n 个字符。如果第三个参数没有提供，返回字符串的剩余字符。

有关此函数的示例，请参见“使用文件输入/输出”一章中的“读取行的示例代码”。

ORD

ORD
顺序转换：*ORD(<String>)*

返回字符串 (0-255) 第一个字母的 ASCII 整数值。

PCDMISAPPLICATIONPATH

PCDMISAPPLICATIONPATH
完整路径显示：PCDMISAPPLICATIONPPATH*()*

返回的字符串值包括到 PC-DMIS 安装应用程序目录的完整路径。此目录包括运行 PC-DMIS 的主要可执行及其他必要程序文件。

PCDMISUSERHIDDENDATAPATH

PCDMISUSERHIDDENDATAPATH
完整路径显示：PCDMISUSERHIDDENDATAPATH*()*

返回的字符串值包括 PC-DMIS
所用的隐藏用户数据目录的完整路径。有关此目录中的文件，请参见“了解文件位置”。

PCDMISUSERVISIBLEDATAPATH

PCDMISUSERVISIBLEDATAPATH
完整路径显示：PCDMISUSERHIDDENDATAPATH*()*

返回的字符串值包括 PC-DMIS
所用的可见用户数据目录的完整路径。有关此目录中的文件，请参见“了解文件位置”。

PCDMISSYSTEMHIDDENDATAPATH

PCDMISSYSTEMHIDDENDATAPATH
完整路径显示：PCDMISSYSTEMHIDDENDATAPATH*()*

返回的字符串值包括 PC-DMIS
所用的隐藏系统数据目录的完整路径。有关此目录中的文件，请参见“了解文件位置”。

PCDMISSYSTEMVISIBLEDATAPATH

PCDMISSYSTEMVISIBLEDATAPATH
完整路径显示：PCDMISSYSTEMVISIBLEDATAPATH*()*

返回的字符串值包括 PC-DMIS
所用的可见系统数据目录的完整路径。有关此目录中的文件，请参见“了解文件位置”。

PCDMISSYSTEMREPORTINGPATH

PCDMISSYSTEMREPORTINGPATH
完整路径显示：PCDMISSYSTEMREPORTINGPATH*()*

返回的字符串值包括 PC-DMIS
所用的可见报告目录的完整路径。此目录包括“报告”窗口使用的报告和标签模板。

RIGHT

RIGHT
字符串右侧的 n 个字符：*RIGHT(<String>, <Integer>)*

返回由字符串中最右侧的 n 个字符组成的字符串，n 由整数指定。

SYSTEMDATE

SYSTEMDATE
系统日期：SYSTEMDATE(<日期格式字符串>)

返回包含当前日期信息的日期格式字符串。比如，如果当前日期是 2014 年 2 月 12 日，命令 SYSTEMDATE("MM’/’dd’/’yy") 将返回字符串 "02/12/14"。

利用以下字符串元素建立日期字符串。元素的大小写必须与以下所示的相同（是 MM 而非 mm）。日期格式字符串元素中的非日期字符（如空格）将作为输入字符串，显示在输出字

符串中的相同位置。输入字符串中由单引号分隔的字符将显示在输出字符串中的相同位置，但不会显示单引号。

d – 每个月中天数的阿拉伯数字。一位数的日期没有前导零。

dd - 每月的日期（数字）。一位数的日期使用前导零。

ddd – 星期几的三个字母缩写。

dddd – 今天是星期几的全名。

M – 没有前导零的单位数的月份。

MM – 有前导零的单位数的月份。

MMM – 月份的三个字母缩写。

MMMM – 月份全称。

y – 没有前导零的单位数的年。

yy – 有前导零的单位数的年。

yyyy – 用四位数来表示的年。

SYSTEMTIME

SYSTEMTIME
格式化系统时间：*SYSTEMTIME(<时间格式字符串>)*

返回包含当前时间信息的时间格式化字符串。例如，命令 SYSTEMTIME("hh:mm:ss tt") 将以格式化字符串返回时间，如："11:29:40 PM"。

使用以下字符串元素创建时间字符串。元素的大小写必须如下所示（**tt** 而不是 **TT**）。在输出字符串中，时间格式字符串元素之间出现的非时间字符（例如空格）将出现在与输入字符串相同的位置。输入字符串中使用单引号分隔的字符将出现在输出字符串的相同位置，没有单引号。

h – 小时数，没有前导零；12 小时制

hh – 有前导零的单位数小时数；12 小时制

H – 无前导零的单位数小时数；24 小时制

HH – 有前导零的单位数小时数；24 小时制

m – 无前导零的单位数的分钟数

mm – 有前导零的单位数的分钟数

s – 没有前导零的单位数的秒数

ss – 有前导零的单位数的秒数

t – 一个字符的时间标记字符串，例如 A 或 P

tt – 多字符的时间标记字符串，例如 AM 或 PM

SYSTIME

SYSTIME
系统时间：*SYSTIME()*

返回包含当前系统时间的字符串。该函数与上面所述的 SYSTEMTIME 函数不同。函数自动返回天数、日期、时间、年份。

例如：“2014 年 2 月 12 日星期三 13:50:21”

注：返回的显示当前系统时间的字符串将调整为本机的时区设置。

UPPERCASE

UPPERCASE
创建大写字符串：*UPPERCASE(<String>)*

返回与该字符串等价的大写字符串。

数学函数

ABS

ABS 绝对值：*ABS(<Double>)*
返回输入值的绝对值。

EXP

EXP 指数：*EXP(<Double>)*
返回表达式的指数。

LOG

LOG 基数为 10 的对数：*LOG(<Double>)*
返回表达式的基数为 10 的对数。

LN

LN 自然对数： *LN(<Double>)*
返回表达式的自然对数。

ROUND

ROUND 凑整：*ROUND(<Double>)*
返回输入值最近的整数。

SQRT

SQRT 平方根： *SQRT(<Double>)*
返回输入值的平方根。

三角函数

重要：每个三角函数默认使用的及返回的值均为弧度。如想采用角度值，请使用以下说明的 RAD2DEG 函数。

ACOS

ACOS 反余弦：*ACOS(<Double>)*
返回表达式的反余弦值。例如，ACOS(5.0) 返回 0。通常，ACOS(<expression>) 返回表达式值的反余弦。

ASIN

ASIN 反正弦：*ASIN(<Double>)*
返回输入值的反正弦。

ATAN

ATAN 反正切：*ATAN(<Double>)*
返回输入值的反正切。

COS

COS 余弦：*COS(<Double>)*
返回输入值的余弦。

DEG2RAD

DEG2RAD 角度转换为弧度：*DEG2RAD(<Double>)*
返回输入值除以 360 乘以 2π。角度转换为弧度。

RAD2DEG

RAD2DEG 弧度转换为角度：*RAD2DEG(<Double>)*
返回输入值乘以 360 除以 2π。弧度转换为角度。

SIN

SIN 正弦： *SIN(<Double>)*
返回输入值的正弦。

TAN

TAN 正切： *TAN(<Double>)*
返回输入值的正切。

注：输入超出范围的函数（如 ACOS、ASIN、LOG、LN、SQRT 等，输入超出范围将导致计算机崩溃）返回的值为 0。

点函数

ANGLEBETWEEN

ANGLEBETWEEN

夹角：*ANGLEBETWEEN(<vector>, <vector>)*

返回两个矢量之间的夹角（单位：度）。两个参数的表达式必须计算为矢量类型。要得到特征的矢量，例如，您需要使用特征标识，后面跟随 .IJK 扩展名。请参考以下示例。

示例：

```
F1 =一般/点,从属,直角坐标,$
理论值/XYZ,<3,3,3>,$
测定值/XYZ,<3,3,3>,$
理论值/IJK,<1,0,0>,$
测定值/IJK, <1,0,0>
F2 =一般/点,从属,直角坐标,$
理论值/XYZ,<10,10,10>,$
测定值/XYZ,<10,10,10>,$
理论值/IJK,<0,0,1>,$
测定值/IJK, <0,0,1>
ASSIGN/V1=F1.IJK
ASSIGN/V2=F2.IJK
ASSIGN/V3=ANGLEBETWEEN(V1,V2)
注释/运行,否,全屏=否,自动继续=否,
“ " + V1 + "和 " + V2 + "之间的夹角是 " + V3
```

CROSS

CROSS
叉积：*CROSS(<Point>, <Point>)*

返回值属于点类型，是第一个表达式和第二个表达式的叉积。

DELTA

DELTA
矢量偏置：*DELTA(<Point>, <Point>, <Double>)*

该函数使用第一个表达式（点），在第二个表达式（矢量）的方向上以第三个表达式为偏置计算一个新点。例如，`DELTA(MPOINT(0,0,0), MPOINT(1,0,0), 10)` 返回点 10,0,0。

DOT

DOT
点积：*DOT(<Point>, <Point>)*

返回两个点（矢量）的点积。

UNIT

UNIT
单位矢量：*UNIT(<Point>)*

返回点除以其长度。例如，`UNIT(MPOINT(0,0,0))` 返回点 0,0,1。

MPOINT

MPOINT
点强制：*MPOINT(<Expression1>, <Expression2>, <Expression3>)*

强制三个表达式为点类型。

示例：
```
ASSIGN/V1 = MPOINT(2.5,3.6,4)
```

V1.X 的值是 2.5
V1.Y 的值是 3.6
V1.Z 的值是 4.0

参见”点强制“。

指针函数

DIST2D

DIST2D
2d 距离：DIST2D(<FEAT1>, <FEAT2>, <FEAT3>)

特征必须用大括号括起来。

计算命令中前两个自变量（特征1和特征2）之间沿垂直于第三个自变量（特征3）的距离。

- 如果第三个自变量为平面，则计算前两个自变量垂直于平面的距离。
- 如果第三个自变量为直线或圆柱，则计算前两个自变量在活动平面垂直于第三个自变量的距离。

例如，XY平面为第三个变量，则有Z+矢量（0,0,1），报告距离仅在Z轴。

举例

```
ASSIGN/V3=DIST2D({CIR1},{CIR2},{PLN1})
注释/运行,否,全屏=否,自动继续=否,
V3
```

DIST3D

DIST3D
3D 距离：DIST3D(<FEAT1>, <FEAT2>)

计算特征 1 与特征 2 之间的 3D 距离。

注：特征必须用大括号括起来。

举例

```
ASSIGN/V3=DIST3D({CIR1},{CIR2})
注释/运行,否,全屏=否,自动继续=否,
V3
```

GETCOMMAND

GETCOMMAND
通过参数指定的命令获得指针：`GETCOMMAND(<Integer 或 String>, <String>, <Integer>`

第一个参数—命令信息字段

第一个参数是命令信息字段。该字段指定要搜索的命令类型。可以传入以下参数：

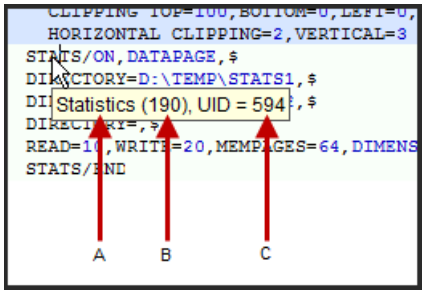
- 一个命令描述字符串。见（A）以下表格。
- 一个命令类型编号。见（B）以下表格。
- 唯一的数字标识。见（C）以下表格。

如果传入了唯一的命令ID，则不需要其它自变量。

获得命令描述字符串，命令类型编号，命令唯一的数字标识：

1. 在编辑窗口单击右键
2. 选择**更改弹出显示 | 命令信息**（PC-DMIS 须处于命令模式下）。

3. 将鼠标指针放在所需命令上。该命令的命令说明、类型编号和唯一数字标识将显示在弹出菜单中。



示例命令信息显示（A）命令说明字符串，（B）命令类型编号和（C）唯一编号标识符（UID）。

第二个参数 - 搜索方向

第二个参数是搜索方向。有效值包括：

值	描述
UP	该值表示从当前命令开始向上进行搜索。
DOWN	该值表示从当前命令开始向下进行搜索。
TOP	该值表示从测量例程开头开始向下进行搜索
底部	该值表示从测量例程的最后一个对象开始向上进行搜索

第三个参数 - 要查找的实例

第三个参数表明如果同一个命令多个实例在测量例程里，应该找到哪个命令实例。

示例：如果测量程序有两个 STATS/ON 命令实例，要获得顶部第二个实例的指针，将传入 "2" 作为第三个参数，并传入 "TOP" 作为此处所示的第二个参数。

```
ASSIGN/V1 = GETCOMMAND("Statistics", "TOP", 2)
```

GETCOMMAND 函数可以用来提供 GETTEXT 字符串函数的第三个参数。有关 GETTEXT 信息，请参见“字符串函数”。

LEN

LEN
指针循环计数：*LEN(<POINTER>)*

返回指针处于循环中的次数。比如，如果特征 CIR1 在循环中重复执行 10 次，你可以使用 ASSIGN 语句存储 CIR1 执行了多少次：`ASSIGN/V1=LEN({CIR1})`

数组函数

ARRAY

ARRAY：创建数组：*ARRAY(<EXPRESSION1>, <EXPRESSION2>, <EXPRESSION3>, …)*
使用由表达式参数来表示的数组元素来创建数组对象。数组元素使用基指数 1 编号。

EQUAL

EQUAL：数组元素与元素的比较：*EQUAL(<ARRAY>, <ARRAY>)*
通过逐个元素的比较来确定两个数组是否是相同的元素。如果两个数组包含的元素个数不一样或者一个数组中的任何一个元素与其他数组中的相关元素不匹配，则函数返回为 0。否则，函数将返回 1。

LEN

LEN：数组元素计数： *LEN(<ARRAY>)*
返回数组中元素数。

MAX

MAX：最大数组元素：*MAX(<ARRAY>)*
返回数组中的最大元素。数组中的项目按照数字或字母比较。

MIN

MIN：最小数组元素：*MIN(<ARRAY>)*
返回数组中的最小元素。数组中的项目按照数字或字母比较。

其他函数

ARCSEGMENTENDINDEX

ARCSEGMENTENDINDEX
返回指定弧段的终止点在扫描中的索引号：ARCSEGMENTENDINDEX*(<ID>, <index>,<tol1>,<tol2>)*

<ID> - 第一个参数是字符串型的 ID，它是这个函数得到的这个弧扫描段的最后一个点的索引号。它可以是带引号的 ID，或者是作为一个扫描的 ID 被强制为字符串类型的任何其他表达式。

<index> -
第二个参数是需要获取终止点索引号的弧段的索引号。这是一个基于扫描的值。例如，如果要获取扫描的第三段弧的终止点号，该索引值就为 3。

<tol1> - 第三个参数是普通的特征公差。它是将扫描截取为直线和弧的最大形状公差。

<tol2> -
第四个参数是精确的公差。通常这个较严的公差用于从形状偏差在该公差带内的段上拾取点。

一旦有了弧的起始点和终止点索引号，可以在构造特征内用这些点构造一个分隔弧特征。参见"从扫描段创建直线特征示例"的相似示例。

ARCSEGMENTSTARTINDEX

ARCSEGMENTSTARTINDEX
返回扫描中指定圆弧段的起始点索引号：ARCSEGMENTSTARTINDEX*(<ID>, <index>,<tol1>,<tol2>)*。

<ID> - 第一个参数是字符串型的
ID，它使这个函数得到的这个弧扫描段的第一个点的索引号。它可以是带引号的
ID，或者是作为一个扫描的 ID 被强制为字符串类型的任何其他表达式。

<index> -
第二个参数是需要获取终止点索引号的弧段的索引号。这是一个基于扫描的值。例如，如果要获取扫描的第三段弧的起始点号，该索引值就为 3。

<tol1> - 第三个参数是普通的特征公差。它是将扫描截取为直线和弧的最大形状公差。

<tol2> -
第四个参数是精确的公差。通常这个较严的公差用于从形状偏差在该公差带内的段上拾取点。

这是两个附加的参数，用于控制扫描中哪一个圆弧段是允许的。这些参数只能用 PC-DMIS Settings Editor 更改。半径小于 `MinimumArcSegmentRadiusInMM` 值的圆弧段是不被接受的。这个参数的默认值是 2mm。同样的，圆弧的半径大于 `MaximumArcSegmentRadiusInMM` 值的也是不被接受的。这个参数的默认值是 2000mm（这个值尽量不要更改）。

一旦有了弧的起始点和终止点索引号，可以在构造特征内用这些点构造一个分隔弧特征。参见"从扫描段创建直线特征示例"的相似示例。

EOF 和 EOL

EOF 和 **EOL**
有关该函数的信息，请参见“使用文件输入/输出”一章中的“检查文件结束或行结束”。

FUNCTION

FUNCTION
创建函数： FUNCTION*((<PARAM1>, <PARAM2>…), <EXPRESSION>)*

创建一个函数，使用参数列表指定的参数数量，将这些参数替换到表达式中。

- 使用 `FUNCTION` 关键字时的第一项即参数列表。
- 该列表包含用逗号分隔的参数名。
- 参数列表也使用圆括号。
- 第二项是表达式。
- 表达式中包含调用函数时应替换为参数的参数名。

参见"通用函数示例"主题的例子。

GETROTABDATA

GETROTABDATA
返回当前转台数据：GETROTABDATA*(<参数>)*

<参数> - 可以是“中心”或“角度”。

“中心” - 返回转台的 XYZ 中心。

“角度” - 返回转台的当前角度。

示例：

`ASSIGN/V1 = GETROTABDATA("CENTER")`	V1就是转台的XYZ中心值
`ASSIGN/V2 = GETROTABDATA("ANGLE")`	V2就是转台当前角度值

IF

IF
条件表达式求值：IF*(<EXPRESSION1>, <EXPRESSION2>, <EXPRESSION3>)*

如果表达式1的值为真（非零），该函数将返回表达式2的值；否则，该函数将返回表达式3的值。

ISIOCHANNELSET

ISIOCHANNELSET
表达式有两个参数。第一个参数表示要检查哪一个 I/O
通道（数据范围根据所使用的机器而定）。第二个参数决定了软件是要查询臂 1 还是臂
2。如果第二个参数是1（一），软件将查询臂 2
的控制柜。如果第二个参数不存在（或者被设置为 0），则 IO 通道会查询臂 1
的控制柜。如果不是出在多臂模式，臂 1 控制柜是唯一的选择。

注：如果提供的测头数据类型、测尖 ID、测头文件名或通道号无效，则表达式值为 0。

示例：

`ASSIGN/V4 = ISIOCHANNELSET(3, 0)`	当设置了通道时V4的值为1(计算的结果为True)，否则值为0(计算的结果为False)。

LINESEGMENTENDINDEX

LINESEGMENTENDINDEX
返回扫描段中指定行终点的索引号：LINESEGMENTENDINDEX*(<ID>, <index>,<tol1>,<tol2>)*。

<ID> - 第一个参数是字符串值，指定要截取直线段的终止点索引号的扫描的 ID。它可以是带引号的 ID，或者是作为一个扫描的 ID 被强制为字符串类型的任何其他表达式。

<index> -
第二个参数是直线段的索引号，你可以从这条直线得到结束点号。这是一个基于扫描的值。例如，如果你想要扫描中第三段线的终止点，则这个线段索引号将会是 3。

<tol1> - 第三个参数是普通的特征公差。它是将扫描截取为直线和弧的最大形状公差。

<tol2> -
第四个参数是精确的公差。通常这个较严的公差用于从形状偏差在该公差带内的段上拾取点。

一旦有了直线段的起始点和终止点索引号，可以在构造特征内用这些点构造一个直线特征。参见"从扫描段创建直线特征示例"中的示例。

LINESEGMENTSTARTINDEX

LINESEGMENTSTARTINDEX
返回扫描中指定直线段的起点的索引号：LINESEGMENTSTARTINDEX*(<ID>, <index>,<tol1>,<tol2>)*。

<ID> - 第一个参数是字符串型的 ID，它使这个函数得到的这个线扫描段的线段的第一个点的索引号。它可以是带引号的 ID，或者是作为一个扫描的 ID 被强制为字符串类型的任何其他表达式。

<index> -
第二个参数是需要获取起始点索引号的直线段的索引号。这是一个基于扫描的值。例如，如果要获取扫描的第三段直线段的起始点号，该索引值就为 3。

<tol1> - 第三个参数是普通的特征公差。它是将扫描截取为直线和弧的最大形状公差。

<tol2> -
第四个参数是精确的公差。通常这个较严的公差用于从形状偏差在该公差带内的段上拾取点。

这是两个附加的参数，用于控制扫描中哪一个直线段是允许的。这些参数只能用 PC-DMIS Settings Editor 更改。直线段长度小于 `MinimumLineSegmentLengthInMM` 的值是不被接受的。该参数的默认值是 2 mm。

一旦有了直线段的起始点和终止点索引号，可以在构造特征内用这些点构造一个直线特征。参见"从扫描段创建直线特征示例"中的示例。

PROBEDATA

PROBEDATA
返回当前或指定测头的数据：PROBEDATA*(<OPTPROBEDATATYPE>, <OPTTIPID>, <OPTPROBEFILENAME>)*

这个函数有三个可选参数。如果你使用一个以上的参数，只需要在参数之间用逗号分隔。在空的参数之间不需要用逗号分隔。例如，为了得到当前测头的直径，你可以使用 `ASSIGN/V1 = PROBEDATA("DIAM")`。

OPTPROBEDATATYPE -
该参数指定返回测头数据的类型。如果未提供该参数，将返回当前测尖 ID。该参数属于字符串类型。第一个表达式可以是任意一个值为有效字符串表达式的表达式。第一个参数包括以下有效的字符串表达式（不区分大小写）。这些有效的字符串包括文本表达式和双引号内的文本：

"Offset" - 测量的测尖 X,Y,Z 偏置。返回点类型数据。

"Vector" - 测尖矢量。返回点类型数据。

"A" - 测尖 A 角。返回双精度数据。

"B" - 测尖 B 角。返回双精度数据。

"Diam" *或* ***"Diameter"*** - 测量的测尖直径。前面四个字母要求是 "Diam"，也可以增加字母至全称 "Diameter"。返回双精度数据。

"Thick" *或* ***"Thickness"*** - 测量的测尖厚度。前面五个字母要求是 "Thick"，也可以添加更多字母直到全名 "Thickness"。返回双精度数据。

"Date" - 测尖最后一次校验的日期。返回字符串类型数据。

"Time" - 测尖最后一次校验的时间。返回字符串类型数据。

"ID" - 测尖 ID。默认参数。返回字符串类型数据。

"PrbRdv" - 测头径向偏差。返回双精度数据。

"Standarddeviation" - 测头标准差。返回双精度数据。

"C" **-** CW43 光测头的 C 角。返回整数型数据。

注：在 "**Offset**"、"**Diameter**" 或 "**Thickness**" 前面加上 "**T**" 会返回理论信息（如 TOFFSET、TDIAMETER 和 TTHICKNESS）。

OPTTIPID -
该可选参数指定在获取第一个表达式中指定的测头数据时要使用的测尖。如果未提供，则使用当前测尖。该参数应为字符串类型。

OPTPROBEFILENAME -
该可选参数指定在获取测头数据时要使用的测头文件名。如果未提供，则使用当前测头文件。

示例：

`ASSIGN/V1 = PROBEDATA()`	V1设置为当前的测尖ID（比如“T1A0B0”）
`ASSIGN/V2 = PROBEDATA("TOFFSET", "T1A45B0")`	V2设置为测尖T1A45B0的理论测针偏置
`ASSIGN/V3 = PROBEDATA("Date", "T1A90B90", "MYPROB")`	V3设置为一个字符串，表示测头文件MYPROB的测尖T1A90B90最后一次校验的日期。

TUTORELEMENT

TUTORELEMENT: 该函数取一个自变量，类型为数字或字符串（字符串将视为特征的ID）。

该函数使用变量类型，*结构*。参见"结构"对结构和子元素的说明。

示例：

`ASSIGN/E = TUTORELEMENT(1)`	创建一个单精度型Tutor元素结构
`ASSIGN/WM = TUTORELEMENT(n)`	对于任意大于 1 的数字，创建 n 个 Tutor 元素结构组成的数组。
`ASSIGN/CIR1E = TUTORELEMENT("CIR1")`	将特征CIR1的数据复制到Tutor元素结构。

TutorElement 结构当前包含以下子元素：

子元素	描述
标识	特征标识字符串
类型	INTEGER (FTYPE)
X, Y, Z	X、Y 和 Z 坐标值
极半径	极坐标半径
极角	极角
CX	I
CY	J
CZ	K
DM	直径 1
DM2	直径 2
DS	距原点的距离
A	角度
AXY	在 XY 平面的角度
AYZ	在 YZ 平面的角度
AZX	在 ZX 平面的角度
F	形状误差
SDEV	标准差
TP	位置

QUALTOOLDATA

QUALTOOLDATA： 此函数返回当前或指定校验工具的数据。它有以下语法：

QUALTOOLDATA(<TOOLINFO>, <TOOLID>, <FACENUMBER>)

这个函数获取三个可选的参数。它至少需要一个参数才能返回数据。

第一个参数，*<TOOLINFO>,*返回校验工具指定的信息类型，是*字符串*值。如果没有输入这个参数，函数返回当前或指定校验工具的名称。

- "***DIAM***" - 返回双精度类型的工具直径。
- "***ID***" - 返回字符串类型的工具名称。
- "***LENGTH***" - 与 “DIAM” 功能一样。它也返回双精度类型的工具直径。
- "***OVERRIDEIJK***" - 以点值返回搜索覆盖矢量 IJK。
- "***POLYDIAM***" - 以双精度值返回指定多面体的直径。
- "***POLYIJK***" - 以点值返回指定多面体的矢量 IJK。
- "***POLYXYZ***" - 以点值返回指定多面体的中心 XYZ。
- "***SHANKIJK***" - 以点值返回测杆矢量 IJK。
- "***TYPE***" - 返回整数类型的工具类型（0 为球，1 为臂 2，2 为多面体，3 为臂 2 多面体）。
- "***WIDTH***" - 此参数不再使用了。
- "***XYZ***" - 返回点数据类型的工具的 XYZ 位置。

第二个参数，*<TOOLID>*返回校验工具指定的信息类型，是*字符串*类型的值。如果不输入这个参数，PC-DMIS将返回当前校验工具的信息。这个字符串不能辨别不同情况。

第三个参数,*<FACENUMBER>*,仅在使用多面体校验工具而且第一个参数为"POLYXYZ", "POLYIJK",或"POLYDIAM"时使用。该值为整数值，指定要获取多面体工具数据的面的序号。

示例：

`ASSIGN/VDIAM = QUALTOOLDATA("DIAM","SPHERE_1_IN")`	为变量 VDIAM 赋值工具 SPHERE_1_IN 的直径。
`ASSIGN/VID = QUALTOOLDATA("ID")`	为变量 VID 赋值当前工具的名称。
`ASSIGN/VTYPE = QUALTOOLDATA("TYPE")`	为变量 VTYPE 赋值当前工具的类型。
`ASSIGN/VPOLYDIAM = QUALTOOLDATA("POLYDIAM","POLYTEST",3)`	为变量 VPOLYDIAM 赋值多面工具 POLYTEST 的第 3 个面的直径。

函数示例

以下一些不同的函数示例或许有助你创建和使用自定义函数：

- 通用函数示例
- 作为变量传递的函数示例
- 包含多个参数的函数示例
- 创建其他函数的函数示例
- 作为数组成员的函数示例
- 递归定义的函数示例

一般函数示例

```
Assign/MYFUNC = FUNCTION((X,Y,Z), X*3 + Y*2 + Z)
```
创建一个用户定义的函数，并将此函数分配给变量 MYFUNC。函数有三个参数：X、Y 和 Z。

X乘以3。

Y乘以2。

Z保持为传入的值。

返回的值是X+Y+Z的和。

```
Assign/V1 = MYFUNC(7,2,5)
```
通过对传入函数 MYFUNC(7,2,5) 的参数求值，为 V1 赋值 30。参数 7
替换此函数定义中表达式部分的 X。因此，X*3 变为 7*3 或 21。

Y的部份将由2代替, 因此Y*2变为 2*2, 或4。

Z的部份由5代替。

所有值相加 (21 + 4 + 5)传递到V1。

作为变量传递的函数示例

函数可以作为变量进行传递。下例创建在以上一般函数示例的基础之上：

```
Assign/NEWFUNC = MYFUNC
```
将变量 NEWFUNC 设置为具有与 MYFUNC 的函数相同的函数。

```
Assign/V3 = NEWFUNC(12,2,3)
```
为 V3 分配函数 (36 + 4 + 3) 中表达式求值的结果 43。

包含多个参数的函数示例

函数可以包含多个参数：

```
Assign/ADDANDDOUBLE = FUNCTION((A,B), 2*(A+B))
```
创建函数并将其分配至函数
ADDANDDOUBLE。这个函数有两个参数，将这两个参数相加，然后将结果与 2 相乘。

```
Assign/V2 = ADDANDDOUBLE(4, 5)
```
将 V2 分配为值 18。参数 4 与 5 代入到函数的表达式部分，因而变为 2*(4+5)。

创建其它函数的函数示例

函数可以创建其它函数。

```
Assign/COMPOSE = FUNCTION((F, G), FUNCTION((X), G(F(X)) ))
```
将 COMPOSE
分配为一个函数，该函数以两个函数作为参数，并使用这两个函数创建了一个新的函数。

```
Assign/ADD2 = FUNCTION((X), X+2)
```
将 ADD2 分配为对传入的参数加 2 的函数。

```
Assign/ADD3 = FUNCTION((X), X+3)
```
将 ADD3 分配为对传入的参数加 3 的函数。

Assign/ADD5 = COMPOSE(ADD2, ADD3)
将 ADD5 分配为由 ADD2 与 ADD3 组成的函数。

```
Assign/V5 = ADD5(3)
```
将 V5 分配为拥有 V8 的值。

作为数组成员的函数示例

函数可以是数组的成员。

```
Assign/ANARRAY = ARRAY(3, FACTORIAL, "Hello World", ADD5)
```
将 ANARRAY 分配为一个具有这 4 个元素的数组：数值 (3)、函数 (FACTORIAL)、字符串 ("Hello World") 以及函数 (Add5)。

```
Assign/V6 = ANARRAY[2](4)
```
ANARRAY 的第二个元素就是函数 FACTORIAL。参数4 传入了该函数，结果 24 被分配给 V6。

```
Assign/V7 = ANARRAY[2](ANARRAY[4] (ANARRAY[1]))
```
从内向外：ANARRAY (3) 的第一个元素传给第四个数组元素 (Add5) 的函数。结果 8
传给第二个数组元素 (FACTORIAL) 的函数，并被赋值给 V7。V7 则接收值 40320。

递归定义的函数示例

可以递归定义函数，即可以将函数定义为调用本身。

```
Assign/FACTORIAL = FUNCTION((X), IF(X<=1, 1, X*FACTORIAL(X-1))
```

创建名为 Factorial 的函数，该函数拥有一个参数。如果参数小于或等于 1，则求得的值 1，否则为 X-1 的 FACTORIAL 乘以 X。

```
Assign/V4 = FACTORIAL(5)
```

将 V4 分配为值 120 (5*4*3*2*1)。

通过扫描段构造直线示例

该主题提供一个使用PC-DMIS的表达式语言和特定的直线段函数输出扫描中的直线段起始点和终止点数，在构造特征中利用提取到的点创建你需要的直线特征的例子。你可以用本例的方法从扫描创建一段弧。

假定测量例程有一项名称为 SCN1 的扫描特征，该特征类似于：

```
SCN1=FEAT/SCAN,LINEAROPEN,SHOW HITS=NO,SHOWALLPARAMS=YES
        执行模式=重新学习，标称值模式=查找标称值，安全平面=否，单点=否，厚度=0
        查找标称值=5，仅选择=否，使用最佳拟合=否，测头补偿=是，避让移动=否，距离=0，CAD补偿=否
        方向1=变量
        触测点类型=矢量
        起始矢量=0,-1,0
        方向矢量=1,0,0
        剖面矢量=0,0,1
        终止矢量=0,-1,0
        平面矢量=-1,0,0
        点1=100,0,-5
        点2=70,0,-5
        测量/扫描
        基本扫描/直线，显示触测=否，显示所有参数=是
        <100,0,-5>,<70,0,-5>,剖面矢量=0,0,1,方向矢量=1,0,0
        起始矢量=0,-1,0,终止矢量=0,-1,0,厚度=0
        过滤器/NULL过滤器,
        执行模式=重新学习
        边界/平面,<70,0,-5>,平面矢量=-1,0,0,交叉数=2
        触测类型=矢量
        标称值模式=查找标称值，5
        终止扫描
        终止测量/
```

要从这个扫描创建一行，需使用 LINESEGMENTSTARTINDEX 和 LINESEGMENTENDINDEX 函数提取数据，如：

```
ASSIGN/LINESTARTINDEX = LINESEGMENTSTARTINDEX("SCN1", 1, 0.4,0.1)
ASSIGN/LINEENDINDEX = LINESEGMENTENDINDEX("SCN1", 1, 0.4, 0.1)
```

这将告诉PC-DMIS找到名为“SCN1”的扫描，并从其第一个直线段获取位于定义的公差范围内的起始和终止索引值。并将索引值赋值线LINESTARTINDEX和LINEENDINDEX变量。

一旦你有了直线段赋值得到的起始和终止点序数，你就可以在构造直线中使用这些变量，如下：

```
LIN4=FEAT/LINE,RECT,UNBND
    理论值/100.225,0,-5.011,1,0,0
    实际值/100.225,-0.005,-5.011,1,-0.0000388,0
    CONSTR/LINE,BF,2D,SCN1.HIT[LINESTARTINDEX..LINEENDINDEX],,
    删除超差点/关，3
    FILTER/OFF,WAVELENGTH=0
```

注意，在上面突出显示的线段特征的代码中，PC-DMIS使用的是您从扫描中提取出来的起始及终止编号来创建特征的：`SCN1.HIT[LINESTARTINDEX..LINEENDINDEX]`

操作数强制

操作数可以使用任何强制运算符强制为其它类型：

整型强制

INT(<Expression>) - 强制表达式值的类型为整数

`INT(4)`	值为 4
`INT(4.5)`	值为 4
`INT("Hello World")`	值为 0
`INT("2")`	值为 2
`INT("2.2")`	值为 2
`INT("3 Blind Mice")`	值为 3
`INT("The 3 Blind Mice")`	值为 0
`INT("3, 4, 5")`	值为 3
`INT(MPOINT(0, 0, 1))`	值为点到原点的距离，该示例值为 1
`INT(MPOINT(3, 4, 5))`	距离值为 7.0711，该表达式的值为 7

双精度强制

DOUBLE(<Expression>) - 强制表达式值的类型为双精度值

DOUBLE(4)	值为4.0
DOUBLE(4.5)	值为 4.5
DOUBLE（“字符串”）	值为0.0
DOUBLE("3.5")	值为3.5
DOUBLE（“3.5 英寸”）	值为3.5
DOUBLE（“圆的测量直径为 3.5 英寸”）	值为0.0
DOUBLE(MPOINT(0,0,1))	值为1.0
DOUBLE(MPOINT(3,4,5))	值为7.0711

字符串强制

STR(<Expression>) - 强制表达式值的类型为字符串

STR(4)	值为“4”
STR(4.5)	值为“4.5”
STR("Hello World")	值为“Hello World”
STR(MPOINT(3,4,5))	值为“3, 4, 5”

点强制

MPOINT(<Expression1>, <Expression2>, <Expression3>) -
将每个表达式的值强制型转为双精度型后，再将表达式的值强制型转为点型。

MPOINT(1, 1, 1)	计算的点为 1.0,1.0,1.0
MPOINT(1.1, 1.1, 1.1)	计算的点为 1.1, 1.1, 1.1
MPOINT("1", "1", "1")	计算的点为 1.0,1.0,1.0
MPOINT(3, 4.5, "5.6")	计算的点为 3.0, 4.5, 5.6
MPOINT(MPOINT(1, 0, 0), MPOINT(0,1,0), MPOINT(3,4,5))	计算的点为 1.0, 1.0, 7.0711

操作数强制和混合类型表达式

混合类型表达式中表达式计算程序自动强制变量。如果因为自动强制，表达式的结果不是所需的结果，在某些情况下，可使用强制运算符产生所需的结果。以下是混合类型表达式中的自动强制的例子。

"CIR" + 1
求得的值为 "CIR1"

"2" + 2
求得的值为 4

"The Value of 2+2 is " + 2 + 2
求得的值为“2+2 等于 22”（因表达式自左至右求值）

"The Value of 2+2 is " + (2 + 2)
求得的值为“2+2 等于 4”

LINE1.XYZ > 2
如果 LINE1 的质心与原点之间的距离大于 2，则求值的结果为 1

LINE1.XYZ > LINE2.XYZ
如果 LINE1 的质心与原点的距离大于 LINE2 的质心与原点的距离，则求值的结果为 1

LINE1.XYZ = LINE2.XYZ
如果 LINE1 与 LINE2 的质心相同（不发生强制），则求值的结果为 1

DOUBLE(LINE1.XYZ) = DOUBLE(LINE2.XYZ)
如果质心与原点的距离相同，则求值的结果为 1

11% 3.1
求值的结果为 2（% 是指定为对整数执行操作的模运算符。它返回离散除法的余数。11%3 = 2.)

CIRCLE1.HIT [3.2].X
求得的值 Circle1 第三个测点的 X 测量值。参数 3.2 自动强制为整数 3。

ID 表达式

许多 PC-DMIS 命令将特征 ID 作为参数使用。例如，构造特征使用 ID
来指明哪些特征要作为构造特征的输入来使用。使用 ID
表达式，用户可以引用特定的特征实例、一组命名类似的特征、子例程调用的特征的实例或外部测量例程中的特征。

特征数组 ID

使用特征数组ID引用特定的特征实例或一组特征实例。例如，如果特征"Circle1"在一个While循环里循环测量五次，五个圆的实例就会在循环里存在。要引用"Circle1"五个实例中的单个实例，使用

特征数组里描述的特征数组语法"Circle1[1]"表示引用第一个实例，"Circle1[2]"引用第二个实例，以此类推。

如要引用某个范围内的实体，要使用 **..** 标志。"Circle1[1..3]" 表示 Circle1 的第 1 至第 3 个实例。"Circle1[3..5]" 表示 Circle1 的第 3 至第 5 个实例。"Circle1[1..5]" 表示 Circle1 的第 1至第 5 个实例。引用某个范围内的特征时，将处理这个构造的集合。

ID 通配符

使用 ID 通配符引用一组名称相似的特征。两个通配符是 "*****" 和 "**?**". （更多信息，请参阅“编辑 CAD 显示”一章中的“使用元字符匹配选择特征”。）

星号 "*" 用来表示 0 或多个任意字符的实例。要引用以字母 “CIR” 开头的所有特征，使用表达式 ID “CIR*”。该语法将创建一组包含所有具有特征 ID "CIR" 的特征，例如 "CIRCLE1"、"CIRCLE2"、"CIR3" 或 "CIR"。

注：如果 CIR3 执行了多次，则仅采用最近一次的测量。要获取不同的执行实例，可使用以下表达式：CIR?[1..3]

问号 "？" 字符用于表示任何字符中的单个实例。

示例：ID 表达式 "MY???1" 将创建一组有六个字符长的特征，它们以 "MY" 开始，以 "1" 结束，例如 "MYCIR1"、"MYCON1"、"MYLIN1" 或 "MYFT21"。

子例程、Basic 脚本或外部例程中的特征的 ID

子例程位于当前测量例程或外部测量例程中。当子例程位于与调用该子例程的例程相同的例程中，可使用特征数组 ID 语法（在“特征数组：”中进行了说明）引用在这个子例程中创建的特征的各个实例。不过，如果子例程在外部测量例程中，则可使用以下语法引用在子例程 "<Call Sub ID>:<FeatID>" 中创建的任何特征。例如，如果名称为 "F1" 的特征在外部子例程中，这个外部子例程从 ID 为 "CS1" 的 Call Sub 命令进行调用，则可使用 ID 表达式 "CS1:F1" 来引用该特征。

示例：此例仅说明了语法 CS1.F1 的使用而并不打算进行使用。

例程 1：PLUS1.PRG

```
SUBROUTINE/PLUS1, A1 = 0, A2 = 0, A3 = 0
F1 =FEAT/POINT,RECT
THEO /A1+1,A2+1,A3+1,0,0,1
ACTL/3,1,1,0,0,1
MEAS/POINT,1
HIT /BASIC,A1+1,A2+1,A3+1,0,0,1,0,0,0
ENDMEAS/
ENDSUB/
```

例程 2：TEST.PRG

```
CS1 =CALLSUB/PLUS1,D:\V30\WINDEBUG\PLUS1.PRG: 3,3,3,,
DIM D1= LOCATION OF POINT CS1:F1 UNITS=IN,$
GRAPH=OFF TEXT=OFF MULT=10.00 OUTPUT=BOTH
AX NOMINAL +TOL -TOL MEAS MAX MIN DEV OUTTOL
X 3.0000 0.0000 0.0000 3.0000 3.0000 3.0000 0.0000 0.0000
----#----
END OF DIMENSION D1
```

Basic 脚本动态创建和删除对象。使用语法“<Basic 脚本 ID>:<特征 ID>”引用由Basic脚本创建的特征。例如，如果 ID 为“BS1”的Basic脚本创建ID为“F2”的特征，使用ID表达式“BS1:F2”引用该特征。

使用附加命令时，可将外部例程附加到 PC-DMIS。要引用附加的例程中的特征，使用以下语法："<Attach Routine ID>:<Feat ID>"。要引用 ID 为 "GEAR1" 的附加测量例程中的特征 "F3"，使用表达式 "GEAR1:F3"。（详细信息，请参阅“添加外部元素”一章中的“附加外部测量例程”。）

ID 表达式组合

数组 ID 表达式、通配字符 ID 表达式、外部子例程、基本脚本和外部测量例程 ID 表达式可组合使用。例如，要在附加 ID "BOLTPAT" 的外部测量例程中引用以字母 "CIR" 开头的所有特征的第三个实例，可使用 ID 表达式 "`BOLTPAT:CIR*[3]`"。

同样，ID表达式可以用于正则表达式。因此，上述特征组的质心实测值可以用以下表达式赋值给变量:

```
ASSIGN/V1 = BOLTPAT:CIR*[3].XYZ
```

同样，ID表达式可以用于正则表达式。因此，上述特征组的质心实测值可以用以下表达式赋值给变量:

```
ASSIGN/V1 = BOLTPAT:CIR*[3].XYZ
```

进入报告对象属性

您可以创建自定义报告和选项卡模板。PC-DMIS 将使用它们在“报告”窗口（参见**视图 | 报告窗口**）中显示报告数据。这些模板使用模板编辑器创建，这些模板编辑器使用的是 Visual Basic 之类的界面，通过该界面可插入、复位位及调整特殊组件（称为“对象”）的大小。

每个对象由“属性”组成，这些属性定义了对象的显示方式及其保存的信息。其中有些属性与所有其他对象相同，而有些属性则仅与相关的对象相同，还有些属性只有特定的对象才有。

PC-DMIS表达式语言可以查询当前加载的报告并将某一特别对象的属性值存入变量中。通过这个语法，它可以获得字符串，整数和实数类型的数据

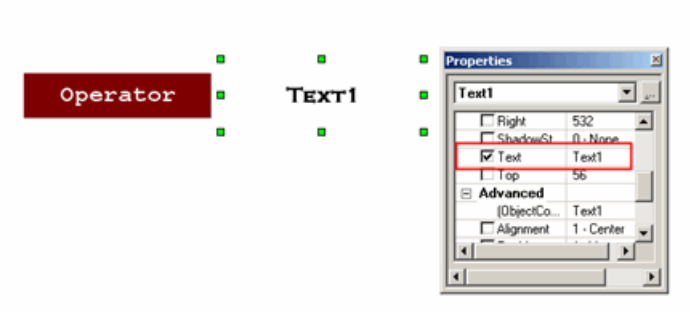
属性查询语法

```
赋值/V1 = 报告<对象名> <属性名>
```

Report 是对当前所载入报告的参考。<Object Name> 是对象的唯一名称，<Property Name> 是该对象的一个有效属性名称。

举例

假定报告模板有一个名称为 "Text1" 的文本对象，您想在最终报告中使用这个对象来显示操作员名称。表示操作员名称的实际的字符串将保存在对象的**文本**属性中。默认情况下，文本属性（显示的文本）最初具有值 "Text1"（参见下图）。由于这是一个用户分配的属性，因此在执行过程中当您输入名称时此属性会发生更改。



“属性”对话框表明了选择的对象以及要查询的属性

想用表达式语言代码来查询该文本对象的“文本”属性并获得键入数据时，您可以使用以下命令：

```
Assign/V1 = Report.Text1.Text
```

在此代码中：

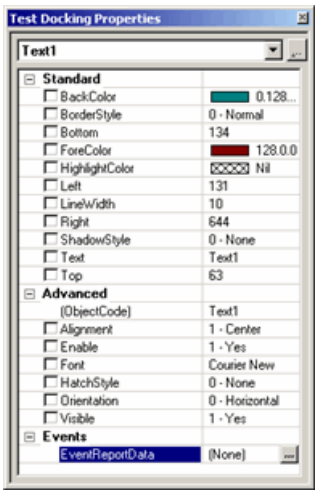
"Report" 会使代码查看加载到“报告”窗口中的报告。

“Text1”告诉它查找对象名称“文本1”；。

"Text" 会使代码在该对象内查找 "Text" 属性。然后会将 "Text" 属性的值传至 V1 变量供您进行进一步处理，或使用 PC-DMIS表达式语言进行显示。

寻找属性

通过访问“报告”模板编辑器中的“报告”模板（**文件 | 报告 | 编辑 | 报告模板**），可查找与特定对象相关的属性，方法是选中对象，然后右键单击该对象，显示其属性表。



"文本"对象属性页

属性页包含两列。左边这列显示属性名称，右边这列显示当前数值。在表达式代码中确保使用正确的属性名称。

重要：查询属性值时，您会发现有些属性返回的是看似无用的数值。这种情况通常在属性具有一组可用的选项时发生，并且 PC-DMIS 返回的是选择的与所显示属性无关的属性的内部值。

例如，**文本**对象的**方向**属性有以下值：

- 0 -水平
- 1 - 垂直向上
- 2 - 垂直向下

但是，如果使用 PC-DMIS 表达式语言获取值，软件将返回以下内容：

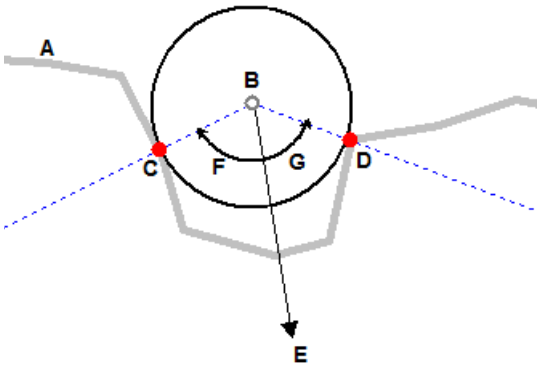
- 0（代表水平）
- 900（代表垂直向上）
- -900（代表垂直向下）

它可能需要进行一些试验和错误，才能确定与属性表上所显示的值相匹配的返回的值。

从构造最小扫描圆中获取信息

使用 PC-DMIS
表达式，你可以从线性扫描的最低点的给定半径的构造圆中获取信息。更多信息，请参考"从现有特征构造新特征"一章中的"在扫描最低点构造圆"主题。

当你构造一个扫描最低点圆特征是，这个圆最终用矢量（称为向下的矢量）与扫描线建立联系。它只在两个叫做触发点（触发点1和触发点2）的地方与扫描线接触。PC-
DMIS然后可以使用这些点来判断向下的矢量与这些触发点（触发点1和触发点2）之间的角度。例如，看这个图表：



A - 构造圆使用的扫描线。
B - 圆质心的最后XYZ位置。
C - 向下矢量的左边触发点。即 CONTACTPOINT1.
D - 向下矢量的右边触发点。即 CONTACTPOINT2。
E - 向下矢量。
F - 从向下矢量到 CONTACTPOINT1 的角度。即 CONTACTANGLE1。
G - 从向下矢量到 CONTACTPOINT2 的角度。即 CONTACTANGLE2。

下文中详述的表达式仅适用此类构造圆特征。也可使用以下语法中的 CONTACTPOINT2 返回使用第二个接触点的相应信息。

`ASSIGN/V1=CIR1.CONTACTPOINT1. XYZ`
返回圆的第一个接触点的 XYZ 点信息以及线 CONTACTPOINT1。

`ASSIGN/V1=CIR1.CONTACTPOINT1.X`
返回 CONTACTPOINT1 的 X 信息。

`ASSIGN/V1=CIR1.CONTACTPOINT1.Y`
返回 CONTACTPOINT1 的 Y 信息。

`ASSIGN/V1=CIR1.CONTACTPOINT1.Z`
返回 CONTACTPOINT1 的 Z 信息。

`ASSIGN/V1=CIR1.CONTACTPOINT1.IJK`
将 CONTACTPOINT1 的 IJK 矢量返回圆的质心。

`ASSIGN/V1=CIR1.CONTACTPOINT1.I`
返回以上 CONTACTPOINT1 IJK 矢量的 I 值。

`ASSIGN/V1=CIR1.CONTACTPOINT1.J`
返回以上 CONTACTPOINT1 IJK 矢量的 J 值。

`ASSIGN/V1=CIR1.CONTACTPOINT1.K`
返回以上 CONTACTPOINT1 IJK 矢量的 K 值。

`ASSIGN/V1=CIR1.CONTACTANGLE1`
返回从向下矢量到 CONTACTPOINT1 的角度。

`ASSIGN/V1=CIR1.CONTACTANGLE2`
返回从向下矢量到 CONTACTPOINT2 的角度。

`ASSIGN/V1=CIR1.CONTACTANGLE`
返回 CONTACTANGLE1 和 CONTACTANGLE2 绝对值之和。不能大于 180 度。