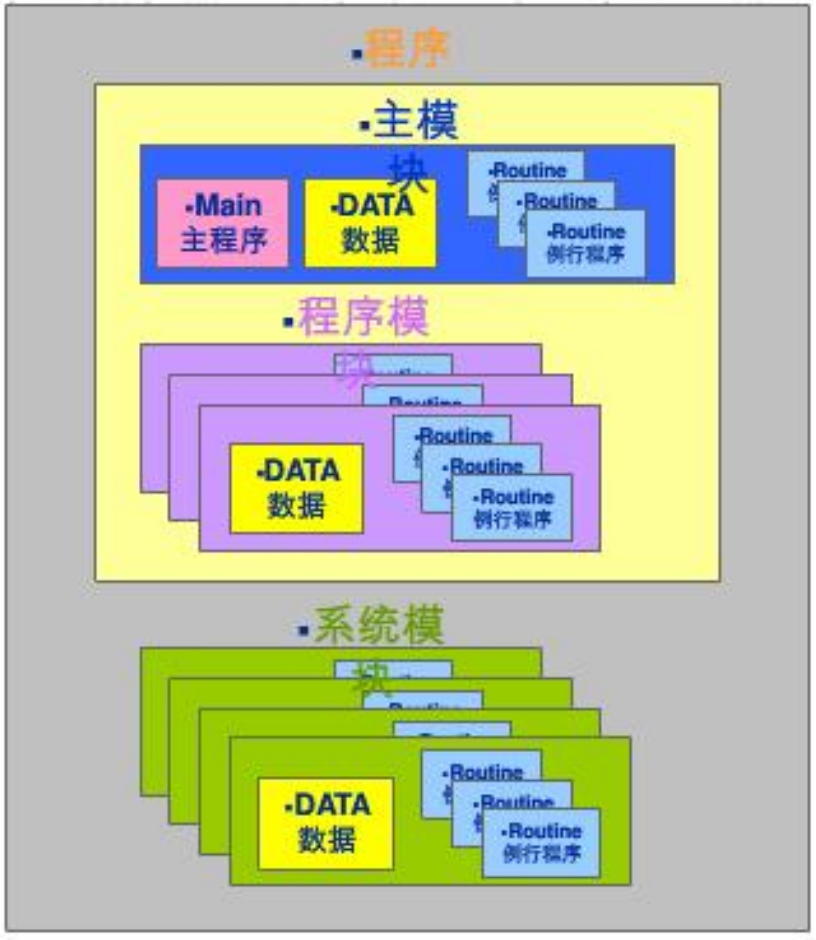
程序存储器
---内存中用来存放数据的部分



·系统参数

·EIO PROC
MMC SIO
MOC SYS

·Flash Disk
快速硬盘
hd0a:\

·USB bd0:\
·USB USB:\

程序结构

模块（Module）按照类型可以分为：

- 主模块（Main Module）
- 程序模块（Program Module）
- 系统模块（System Module）

主模块：
主程序
程序数据
例行程序

程序模块：
程序数据
例行程序

系统模块：
程序数据
例行程序

程序存储器

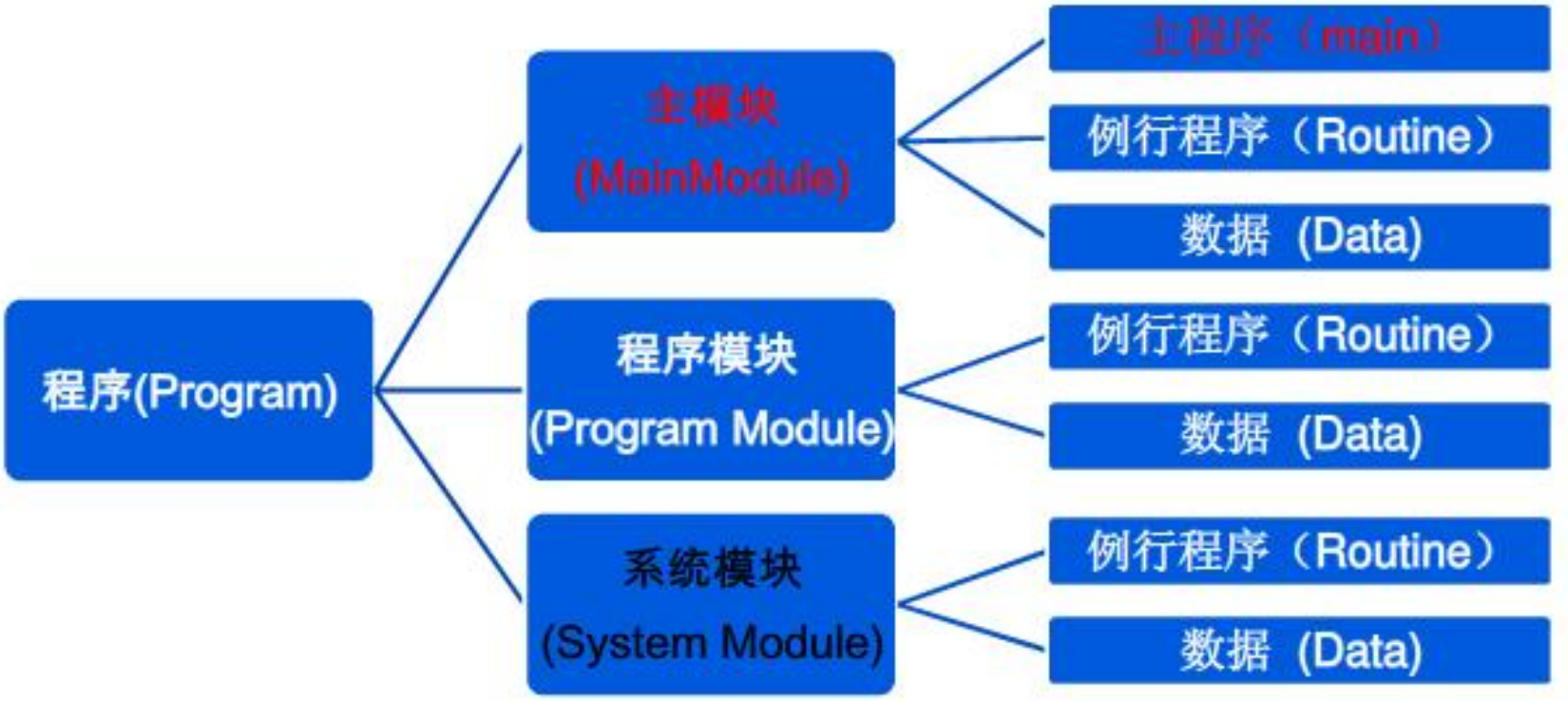
系统模块：
系统数据
例行程序

所有ABB机器人都自带有两个系统模块，USER模块与BASE模块，根据机器人应用不同，有些机器人会配备相应相应的系统模块。建议不要对任何自动生成的系统模块进行修改。

程序存储器

- 机器人程序存储器是由程序模块和系统模块组成
- 机器人程序存储器中，只允许存在一个主程序
- 所有例行程序与数据无论存在于哪个模块，全部被系统共享
- 所有例行程序与数据除特殊定义外，名称必须是唯一的。

程序结构


程序(Program)
主模块
(MainModule)
主程序（main）
例行程序（Routine）
数据（Data)
程序模块
(Program Module)
例行程序（Routine）
数据（Data)
系统模块
(System Module)
例行程序（Routine）
数据（Data)

程序结构

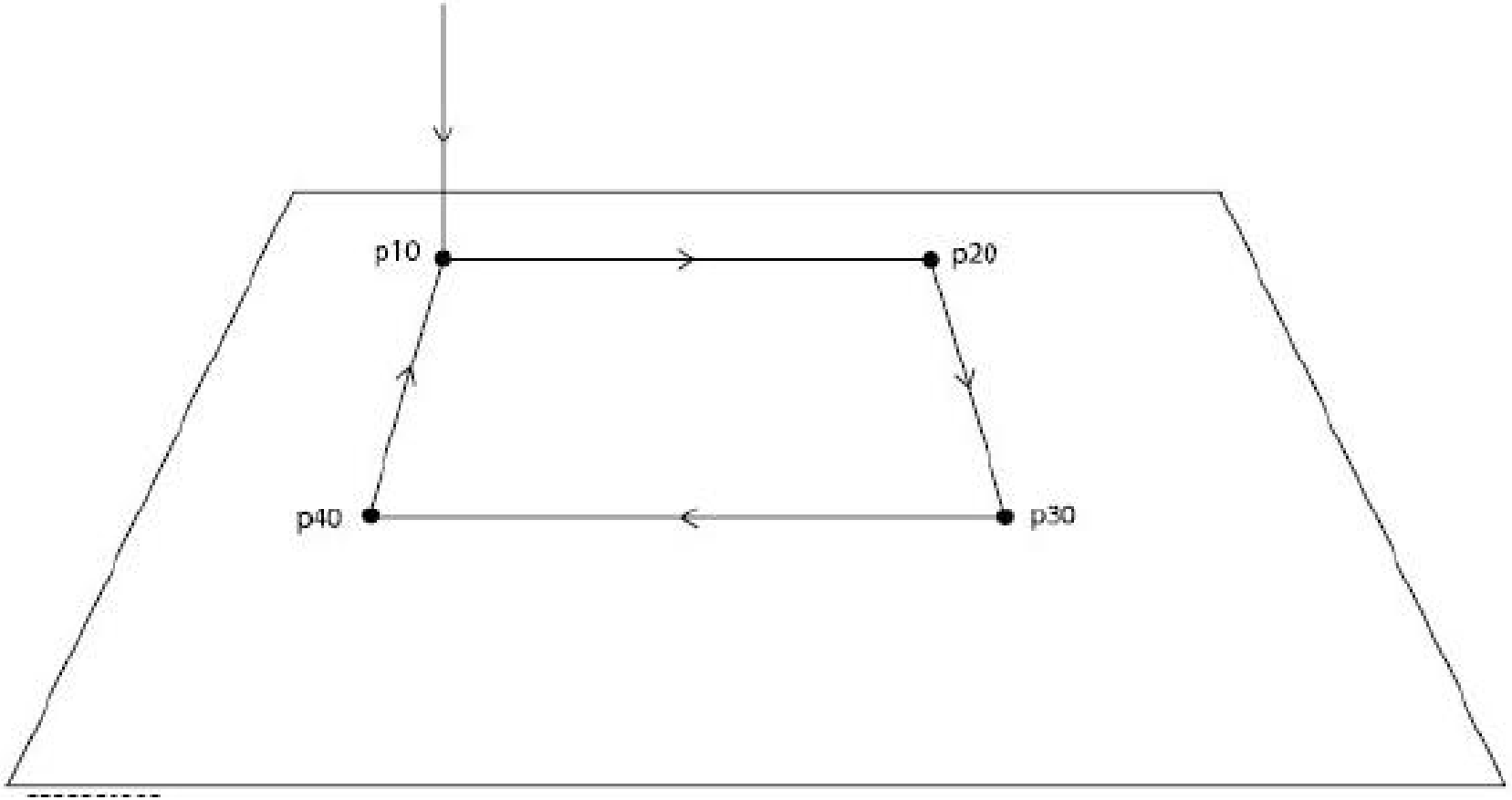
机器人应用程序一般由三部分组成：

- 程序数据
- 一个主程序Main

主程序是一个特别的例行程序，是机器人程序运行的起始，控制机器人程序流程。

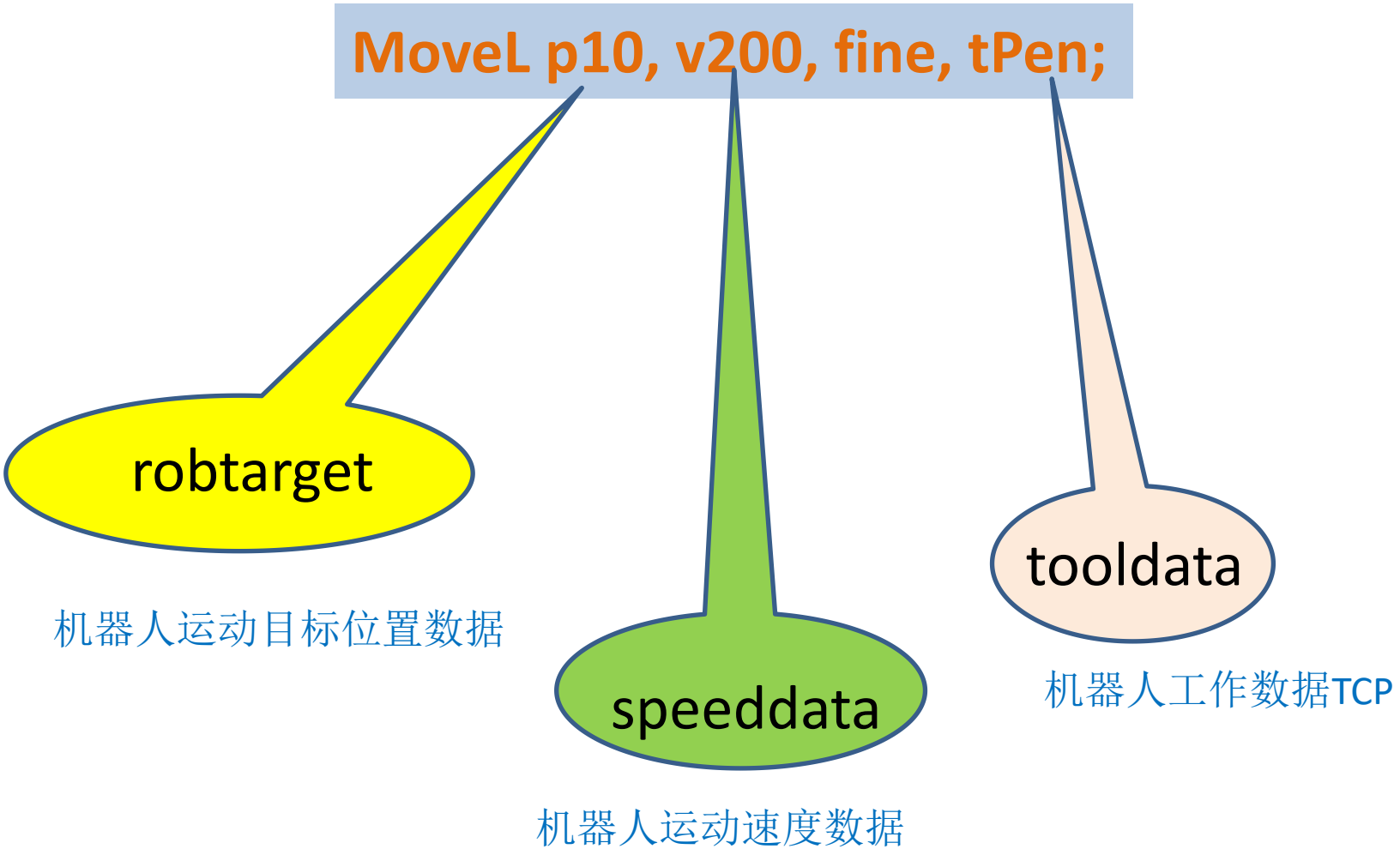
- 几个例行程序

一个简单的编程实例



```
MODULE TEST
PERS tooldata tPen := [ TRUE, [[200, 0, 30], [1, 0, 0 ,0]], [0.8,[62, 0, 17], [1, 0, 0, 0], 0, 0,0]];
CONST robtarget p10 := [ [600, -100, 800], [0.707170, 0, 0.707170,0], [0, 0, 0, 0], [ 9E9, 9E9,
9E9, 9E9, 9E9, 9E9] ];
CONST robtarget p20 := [ [600, 100, 800], [0.707170, 0, 0.707170,0], [0, 0, 0, 0], [ 9E9, 9E9,
9E9, 9E9, 9E9, 9E9] ];
CONST robtarget p30 := [ [800, 100, 800], [0.707170, 0, 0.707170,0], [0, 0, 0, 0], [ 9E9, 9E9,
9E9, 9E9, 9E9, 9E9] ];
CONST robtarget p40 := [ [800, -100, 800], [0.707170, 0, 0.707170,0], [0, 0, 0, 0], [ 9E9, 9E9,
9E9, 9E9, 9E9, 9E9] ];
PROC main()
MoveL p10, v200, fine, tPen;
MoveL p20, v200, fine, tPen;
MoveL p30, v200, fine, tPen;
MoveL p40, v200, fine, tPen;
MoveL p10, v200, fine, tPen;
ENDPROC
ENDMODULE
```

程序数据


MoveL p10, v200, fine, tPen;
robtarget
机器人运动目标位置数据
speeddata
机器人运动速度数据
tooldata
机器人工作数据TCP

程序数据的类型和分类

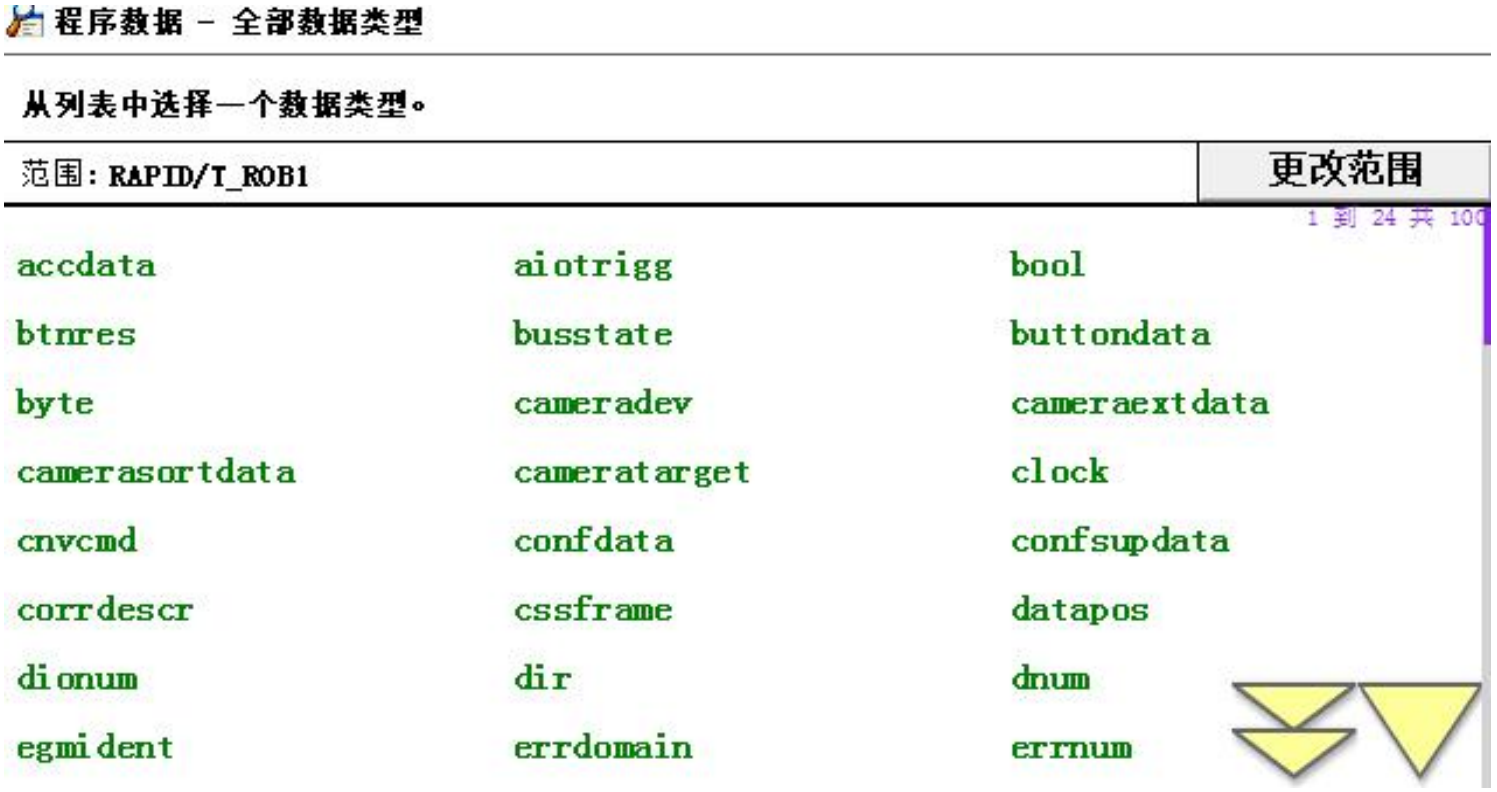


ABB 机器人的程序数据共有76 个，并且可以根据实际情况进行程序数据的创建，为ABB 机器人的程序设计带来了无限可能性。

num

实数

string

字符串

bool

布尔量（True/False）

变量名的命名规则

变量名只能是字母(a-z A-Z)，数字(0-9)，下划线(_)的组合，并且之间不能包含空格，数字和下划线不能放在变量名首位，不超过24个字。

例如：reg1、m_pos1等

程序数据的存储类型

- 变量VAR
- 可变量PERS
- 常量CONST

变量VAR

变量型数据在程序执行的过程中和停止时，会保持当前的值。但如果程序指针被移到主程序后，数值会丢失。

举例说明：

VAR num length:=0;

名称为length 的数字数据

VAR string name:=”John”；

名称为name 的字符数据

VAR bool finish:=FALSE;

名称为finish 的布尔量数据

变量VAR

```
MODULE Module1
  VAR num length:=0;
  VAR string name:="John";
  VAR bool finished:=FALSE;
  PROC main()
    length := 10 - 1;
    name := "john";
    finished := TRUE;
  ENDPROC

ENDMODULE
```

num 表示程序数据类型
*提示：在定义数据时，可以定义变量数据的初始值。
如length 的初始值为0，
name 的初始值为John，
finish 的初始值为FALSE。
*注意：在程序中执行变量型数据的赋值，在指针复位后将恢复为初始值。


可变量

PERS

可变量最大的特点是，无论程序的指针如何，都会保持最后赋予的值。

举例说明：

PERS num nbr:=1;名称为nbr 的数字数据

PERS string test:=”Hello”;名称为test 的字符数据

在机器人执行的RAPID 程序中也可以对可变量存储类型程序数据进行赋值的操作。

在程序执行以后，赋值的结果会一直保持，直到对其进行重新赋值。

常量

CONST

常量的特点是在定义时已赋予了数值，并不能在程序中进行修改，除非手动修改。

举例说明:

CONST num abc:=9.81;

名称为abc 的数字数据

CONST string cba:=”Hello”;

名称为cba 的字符数据

三种数据的存储类型

```
MODULE Module1
  VAR num length:=0;
  VAR string name:="John";
  VAR bool finished:=FALSE;
  PERS string text:="Hello";
  PERS num nbr:=1;
  CONST num gravity:=9.81;
  CONST string greating:="Hello";
  PROC main()
    length := 10 - 1;
```