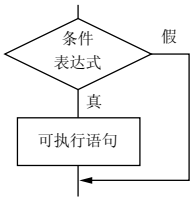
（2）IF 语句

名称、符号、功能		举例
名称	IF 语句	〈语句结构〉
符号	IF	（1）IF、THEN 结构
功能	（1）如果条件表达式的求值为真，那么执行可执行语句。 （2）条件表达式只能包含求值为布尔型的数据。 （3）IF 语句必须与 END_IF 语句配对使用。 （4）一个条件表达式可以有两个或者更多条可执行语句。 （5）条件表达式可以是布尔型的变量。 （6）最大允许 8 层嵌套。在 IF 语句中的 FOR 和 CASE 语句也算是嵌套。	IF〈条件表达式〉THEN〈可执行语句〉; END_IF;

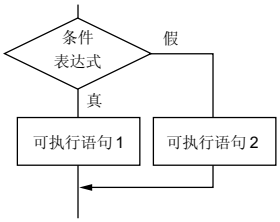
如果条件表达式的求值为真，
那么执行可执行语句。
如果条件表达式的求值为假，
那么什么也不执行。



（2）IF、ELSE 结构

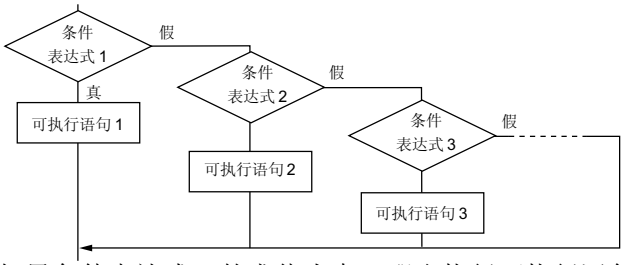
```
IF〈条件表达式〉THEN〈可执行语句 1〉;
  ELSE〈可执行语句 2〉;
END_IF;
```

如果条件表达式的求值为真，
那么执行可执行语句 1。
如果条件表达式的求值为假，
那么执行可执行语句 2。



（3）IF、ELSIF 结构

```
IF〈条件表达式 1〉THEN〈可执行语句 1〉;
  ELSIF〈条件表达式 2〉THEN〈可执行语句 2〉;
  ELSIF〈条件表达式 3〉THEN〈可执行语句 3〉;
    ⋮
END_IF;
```



如果条件表达式 1 的求值为真，那么执行可执行语句 1。
如果条件表达式 2 的求值为真，那么执行可执行语句 2。
如果条件表达式 3 的求值为真，那么执行可执行语句 3。
如果没有条件表达式的求值为真，那么什么也不执行。

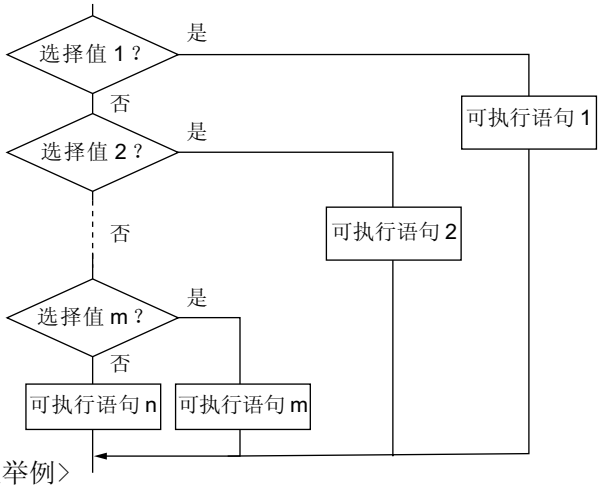
〈编程举例〉

下面所示的样例程序在 N=0 时，将 10 赋给 M；在 N=1 时，将 20 赋给 M；在 N=2 时，将 30 赋给 M 并将 40 赋给 L。*
如果 N 取 0、1、2 之外的数值，那么将 123 赋给 M。

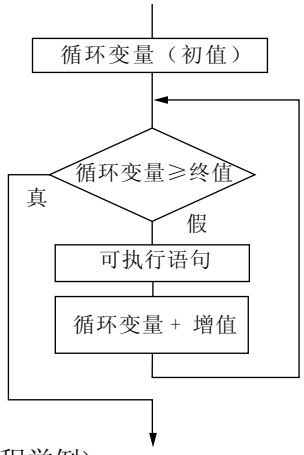
```
IF  N=0  THEN  M:= 10;
ELSIF  N= 1  THEN  M:= 20;
ELSIF  N= 2
    THEN   M:= 30;
                L:= 40;
ELSE  M:= 123;
END_IF;
```

* 一个条件表达式有两个或者更多可执行语句的样例：

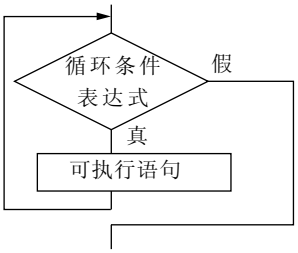
（3）CASE 语句

名称、符号、功能		举例
名称	CASE 语句	〈语句结构〉
符号	CASE	CASE〈条件表达式〉OF 〈选择值 1〉:〈可执行语句 1〉; 〈选择值 2〉:〈可执行语句 2〉; … 〈选择值 m〉:〈可执行语句 m〉; ELSE〈可执行语句 n〉 END_CASE;
功能	（1）如果条件表达式的求值为真，那么根据执行数值选择要执行的可执行语句。 （2）条件表达式和选择值只能是整型数据。 （3）CASE 语句必须与 OF 和 END_CASE 语句配对使用。 （4）一个条件表达式可以有两个或者更多条可执行语句。 （5）最大允许 8 层嵌套。在 CASE 语句中的 FOR 和 IF 语句也算是嵌套。 （6）选择值可以是数字（例如 1、2、2…）或者范围（例如 1..10）。	 〈编程举例〉 下面所示的样例程序在 v=10（v 是 a、b 和 c 的平均值）时，将 1 赋给 singal；在 v=20 时，将 2 赋给 singal；当 v 是 21 至 31 之间的任意值时，将 3 赋给 singal，在其他情况下将 0 赋给 singal。 v:= (a + b + c)/ 3 ; CASE v OF 10: signal :=1; 20: signal :=2; 21..30: signal :=3; ELSE signal :=0; END_CASE;

（4）FOR 语句

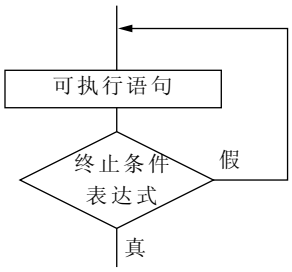
名称、符号、功能		举例
名称	FOR 语句	〈语句结构〉 FOR<a:= 初值>TO<终值>BY<增值>DO<可执行语句> END_FOR;
符号	FOR	
功能	（1）FOR 语句执行循环处理。循环条件由初值、终值和增值给出。 （2）初值、终值和增值可以使用的数据类型为任意整型（整型、双精度整型、无符号整型或无符号双精度整型）。 （3）FOR 语句必须与 TO、BY、DO 和 END_FOR 语句配对使用。 （4）最大允许 8 层嵌套。在 FOR 语句中的 CASE 和 IF 语句也算是嵌套。	〈编程举例〉 下面所示的样例程序在从 a ＝ 1 开始每隔一个变址位置将 b 赋给数组 f。 (a ＝ 1、a ＝ 3、a ＝ 5、a ＝ 7、a ＝ 9) FOR a:=1 TO 10 BY 2 DO f[a]:=b END_FOR;

（5）WHILE 语句

名称、符号、功能		举例
名称	WHILE 语句	〈语句结构〉 WHILE<循环条件>DO <可执行语句>; END_WHILE;
符号	WHILE	

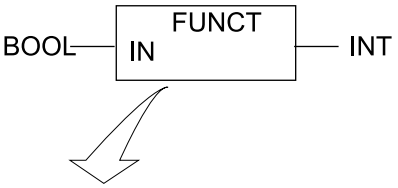
功能	（1）WHILE 语句执行循环处理。如果循环条件的求值为真，那么执行可执行语句。
	（2）WHILE 语句必须与 DO 和 END_WHILE 语句配对使用。
	
	注）：如果在执行循环处理前重复条件已经为假，那么将不执行语句。
		<编程举例>
		下面所示的样例程序在 b>1 时用 2 除 b，并将结果（商）赋给 b。商的小数部分被舍弃。
		
		a:=1;
		WHILE; b>1 DO
		b:=b/2
		f[a]:=b;
		a:=a+1;
		END_WHILE;

（5）REPEAT 语句

名称、符号、功能		举例
名称	REPEAT 语句	〈语句结构〉 REPEAT <可执行语句>; UNTIL<终止条件> END_ REPEAT;
符号	REPEAT	

功能	（1）REPEAT 语句执行循环处理。可执行语句将一直执行直到循环条件的求值为真为止。
	（2）REPEAT 语句必须与 UNTIL 和 END_ REPEAT 语句配对使用。
	
	注）：即使在执行循环处理前终止条件已经为真，但是执行语句总是多执行一次。
		〈编程举例〉
		下面所示的样例程序计算从 1 到 10 的和。
		
		a:=1;
		total:=0;
		REPEAT
		Total:=total+a;
		a:=a+1;
		UNTIL a>10
		END_REPEAT;

（7）RETURN 语句

名称、符号、功能		举例
名称	RETURN 语句	〈语句结构〉 RETURN;
符号	RETURN	〈可执行语句〉 将控制权返还用户函数“FUNCT”时
功能	（1）RETURN 语句将控制权从用户函数或者用户函数块交还给调用 POU。	 <FUNCT>函数 如果 N 是真，那么 FUNCT 返回 500，如果 N 是假，那么 FUNCT 返回 100。 IF IN=TRUE THEN FUNCT:=500; （* 将返回值设为 500*） RETURN; （* 不执行退返回 *） END_IF; FUNCT:=100; （* 下列语句 *） RETURN;

（8）EXIT 语句

名称、符号、功能		举例
名称	EXIT 语句	〈语句结构〉 EXIT;

符号	EXIT	〈编程举例〉
		下面所示的样例程序在 M+N 大于 20 时，其中 M 是初值，终止循环。
		（* 从 FOR 循环终止 *）
		FOR N:=1 TO 10 BY 1 DO
		M:=M+N;
		IF M > 20 THEN EXIT
		END_IF;
		END_FOR;
		（* 从 WHILE 循环终止 *）
		N:=1;
		WHILE N <= 10 DO
		M:=M+N;
		IF M > 20 THEN EXIT
		END_IF;
		N:=N+1;
		END_WHILE;
		（* 从 REPEAT 循环终止 *）
		N:=1;
		REPEAT
		M:=M+N;
		IF M > 20 THEN EXIT
		END_IF;
		N:=N+1;
		UNTIL N > 10
		END_REPEAT;

功能	（1）EXIT 语句终止处理并将控制权交给紧随使用 EXIT 语句的循环结构的语句。	