

DECIDE FOR

```
DECIDE FOR { FIRST } CONDITION
           { EVERY }
           { WHEN logical-condition statement ... } ...
           [ WHEN ANY statement ... ]
           [ WHEN ALL statement ... ]
           WHEN NONE statement ...
END-DECIDE
```

Dieses Kapitel behandelt folgende Themen:

- Funktion
- Syntax-Beschreibung
- Beispiele

Eine Erläuterung der in dem Syntax-Diagramm verwendeten Symbole entnehmen Sie dem Abschnitt *Syntax-Symbole*.

Verwandte Statements: DECIDE ON | IF | IF SELECTION | ON ERROR

Gehört zur Funktionsgruppe: *Logische Bedingungen*

Funktion

Das Statement DECIDE FOR dient dazu, in Abhängigkeit von mehreren Bedingungen eine oder mehrere Handlungen auszuführen.

Anmerkung:

Falls unter einer bestimmten Bedingung *keine* Handlung ausgeführt werden soll, geben Sie das Statement IGNORE in der betreffenden Klausel des DECIDE FOR-Statements an.

Syntax-Beschreibung

FIRST CONDITION	Nur die erste wahre Bedingung soll verarbeitet werden. Siehe auch <i>Beispiel 1</i> .
EVERY CONDITION	Jede wahre Bedingung soll verarbeitet werden. Siehe auch <i>Beispiel 2</i> .
WHEN <i>logical-condition statement</i>	Mit dieser Klausel geben Sie eine oder mehrere logische Bedingungen (<i>logical-condition</i>) an, die verarbeitet werden sollen (siehe Abschnitt <i>Logische Bedingungen im Leitfaden zur Programmierung</i>).
WHEN ANY <i>statement</i>	Mit WHEN ANY können Sie das (die) Statement(s) angeben, die ausgeführt werden sollen, wenn irgendeine der angegebenen Bedingungen erfüllt ist.
WHEN ALL <i>statement</i>	Mit WHEN ALL können Sie das (die) Statement(s) angeben, die ausgeführt werden sollen, wenn alle angegebenen Bedingungen erfüllt sind. Diese Klausel kann nur in Verbindung mit dem Schlüsselwort EVERY eingesetzt werden.
WHEN NONE <i>statement</i>	Mit WHEN NONE können Sie das (die) Statement(s) angeben, die ausgeführt werden sollen, wenn keine der angegebenen Bedingungen erfüllt ist.
END-DECIDE	Das für Natural reservierte Wort END-DECIDE muss zum Beenden des DECIDE FOR-Statements benutzt werden.

Beispiele

- Beispiel 1 — DECIDE FOR-Statement mit FIRST-Option
- Beispiel 2 - DECIDE FOR-Statement mit EVERY-Option

Beispiel 1 — DECIDE FOR-Statement mit FIRST-Option

```

** Example 'DECEX1': DECIDE FOR (with FIRST option)
*****
DEFINE DATA LOCAL
1 #FUNCTION (A1)
1 #PARM (A1)
END-DEFINE
*
INPUT #FUNCTION #PARM
*
DECIDE FOR FIRST CONDITION
  WHEN #FUNCTION = 'A' AND #PARM = 'X'
    WRITE 'Funktion A with parameter X selected.'
  WHEN #FUNCTION = 'B' AND #PARM = 'X'
    WRITE 'Funktion B with parameter X selected.'
  WHEN #FUNCTION = 'C' THRU 'D'
    WRITE 'Funktion C or D selected.'
  WHEN NONE
    REINPUT 'Please enter a valid function.'
    MARK *#FUNCTION
END-DECIDE
*
END

```

Ausgabe des Programms DECEX1:

```
#FUNCTION A #PARM Y
```

Drücken Sie dann die EINGABE-Taste:

```
PLEASE ENTER A VALID FUNCTION
#FUNCTION A #PARM Y
```

Beispiel 2 - DECIDE FOR-Statement mit EVERY-Option

```
** Example 'DECEX2': DECIDE FOR (with EVERY option)
*****
DEFINE DATA LOCAL
1 #FIELD1 (N5.4)
END-DEFINE
*
INPUT #FIELD1
*
DECIDE FOR EVERY CONDITION
  WHEN #FIELD1 >= 0
    WRITE '#FIELD1 is positive or zero.'
  WHEN #FIELD1 <= 0
    WRITE '#FIELD1 is negative or zero.'
  WHEN FRAC(#FIELD1) = 0
    WRITE '#FIELD1 has nein decimal digits.'
  WHEN ANY
    WRITE 'Any of the above conditions is true.'
  WHEN ALL
    WRITE '#FIELD1 is zero.'
  WHEN NONE
    IGNORE
END-DECIDE
*
END
```

Ausgabe des Programms DECEX2:

```
#FIELD1 42
```

Drücken Sie dann die EINGABE-Taste:

```
Page      1
```

```
05-01-11 14:56:26
```

```
#FIELD1 is positive or zero.
#FIELD1 has nein decimal digits.
Any of the above conditions is true.
```