CALL LOOP

Structured Mode-Syntax

```
CALL LOOP operand1 [operand2] ...40
  statement ...
END-LOOP
```

Reporting Mode-Syntax

```
CALL LOOP operand1 [operand2] ...40
  statement ...
[LOOP]
```

Dieses Kapitel behandelt folgende Themen:

- Funktion
- Einschränkung
- Syntax-Beschreibung
- Beispiel

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

Verwandte Statements: CALL | CALL FILE | CALLNAT | DEFINE SUBROUTINE | ESCAPE | FETCH | PERFORM

Gehört zur Funktionsgruppe: *Aufrufen von Programmen und Unterprogrammen*

Funktion

Das Statement `CALL LOOP` dient dazu, eine Verarbeitungsschleife zu generieren, die den Aufruf eines Nicht-Natural-Programms beinhaltet.

Im Gegensatz zum `CALL`-Statement erzeugt das `CALL LOOP`-Statement eine Verarbeitungsschleife, die dazu dient, das Nicht-Natural-Programm wiederholt aufzurufen. Zu der `CALL`-Verarbeitung siehe `CALL`-Statement.

Einschränkung

Innerhalb einer CALL LOOP-Verarbeitungsschleife dürfen die Statements AT BREAK, AT START OF DATA und AT END OF DATA nicht verwendet werden.

Syntax-Beschreibung

Operanden-Definitionstabelle:

Operand	Mögliche Struktur					Mögliche Formate												Referenzierung erlaubt	Dynam. Definition
operand1	C	S				A												ja	nein
operand2	C	S	A	G		A	U	N	P	I	F	B	D	T	L	C		ja	ja

Syntax-Element-Beschreibung:

operand1	Der Name des aufgerufenen Nicht-Natural-Programms (*operand1*) kann entweder als Konstante angegeben werden oder — falls je nach Programmlogik verschiedene Programme aufgerufen werden sollen — als alphanumerische Variable mit Länge 1 bis 8. Ein Programmname muss linksbündig in der Variablen stehen.
operand2	Mit dem CALL LOOP-Statement können Sie bis zu 40 Parameter angeben. Der Aufbau der Parameterliste entspricht der für das CALL-Statement. In der Parameterliste verwendete Felder können schon vorher definiert werden oder erst im CALL LOOP-Statement selbst.
statement ...	Die mit CALL LOOP initiierte Verarbeitungsschleife muss mit einem ESCAPE-Statement beendet werden.
END-LOOP	Ein END-LOOP-Statement muss benutzt werden, um die Verarbeitungsschleife zu schließen.

Beispiel

```
DEFINE DATA LOCAL
1 PARAMETER1 (A10)
END-DEFINE
CALL LOOP 'ABC' PARAMETER1
  IF PARAMETER1 = 'END'
     ESCAPE BOTTOM
  END-IF
END-LOOP
END
```