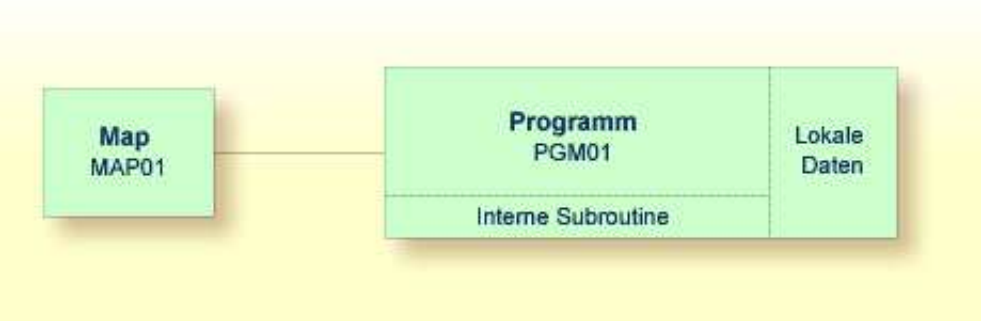
Interne Subroutinen

Natural unterscheidet zwei Arten von Subroutinen: interne Subroutinen, die direkt im Programm definiert werden, und externe Subroutinen, die als separate Objekte außerhalb des Programms gespeichert werden (dies wird später in diesem Tutorial beschrieben).

Sie werden jetzt eine interne Subroutine in Ihrem Programm definieren, mit der ein Stern (*) in die neue Benutzervariable #MARK geschrieben wird. Diese Subroutine wird aufgerufen, wenn ein Mitarbeiter 20 oder mehr Urlaubstage hat.

Wenn Sie mit den Übungen in diesem Kapitel fertig sind, wird Ihre Beispielanwendung folgendermaßen strukturiert sein:



Dieses Kapitel enthält die folgenden Übungen:

- Interne Subroutine definieren
- Interne Subroutine ausführen

Interne Subroutine definieren

Sie werden jetzt die interne Subroutine in Ihr Programm einfügen.

Interne Subroutine definieren

1. Geben Sie Folgendes unter der Benutzervariablen #NAME-END ein:

   ```
   1 #MARK              (A1)
   ```

   Diese Variable wird von der Subroutine benutzt. Daher muss Sie zuerst definiert werden.

2. Um die Subroutine zu definieren, geben Sie Folgendes vor dem END-Statement ein:

   ```
   DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
     MOVE '*' TO #MARK
   END-SUBROUTINE
   ```

   Wenn diese Subroutine ausgeführt wird, schreibt sie einen Stern (*) in die Benutzervariable #MARK.

Anmerkung:
Statt des Statements `MOVE '*' TO #MARK` können Sie auch die folgende Variante des `ASSIGN`- oder `COMPUTE`-Statements benutzen: `#MARK := '*'`.

3. Ändern Sie das `DISPLAY`-Statement wie folgt:

```
DISPLAY NAME 3X DEPT 3X LEAVE-DUE 3X '>=20' #MARK
```

Hiermit wird eine neue Spalte in der Ausgabe angezeigt. Die Überschrift dieser Spalte ist ">=20". Die Spalte enthält einen Stern (*), wenn der betreffende Mitarbeiter 20 oder mehr Urlaubstage hat.

Interne Subroutine ausführen

Jetzt, nachdem Sie die interne Subroutine definiert haben, können Sie den entsprechenden Code zum Aufruf dieser Subroutine eingeben.

Interne Subroutine ausführen

1. Geben Sie Folgendes vor dem `DISPLAY`-Statement ein:

```
IF LEAVE-DUE >= 20 THEN
  PERFORM MARK-SPECIAL-EMPLOYEES
ELSE
  RESET #MARK
END-IF
```

Wenn ein Mitarbeiter gefunden wird, der 20 oder mehr Urlaubstage (`LEAVE-DUE`) hat, wird die neue Subroutine mit dem Namen `MARK-SPECIAL-EMPLOYEES` ausgeführt. Wenn ein Mitarbeiter weniger als 20 Urlaubstage hat, wird der Inhalt von `#MARK` zurückgesetzt.

Ihr Programm sollte nun folgendermaßen aussehen:

```
DEFINE DATA
LOCAL
  1 #NAME-START        (A20)
  1 #NAME-END          (A20)
  1 #MARK              (A1)
  1 EMPLOYEES-VIEW VIEW OF EMPLOYEES
    2 FULL-NAME
      3 NAME (A20)
    2 DEPT (A6)
    2 LEAVE-DATA
      3 LEAVE-DUE (N2)
END-DEFINE
*
RP1. REPEAT
*
  INPUT USING MAP 'MAP01'
*
  IF #NAME-START = '.' THEN
    ESCAPE BOTTOM (RP1.)
  END-IF
*
  IF #NAME-END = ' ' THEN
    MOVE #NAME-START TO #NAME-END
  END-IF
*
  RD1. READ EMPLOYEES-VIEW BY NAME
```

```
    STARTING FROM #NAME-START
    ENDING AT #NAME-END
*
    IF LEAVE-DUE >= 20 THEN
      PERFORM MARK-SPECIAL-EMPLOYEES
    ELSE
      RESET #MARK
    END-IF
*
    DISPLAY NAME 3X DEPT 3X LEAVE-DUE 3X '>=20' #MARK
*
  END-READ
*
  IF *COUNTER (RD1.) = 0 THEN
    REINPUT 'No employees meet your criteria.'
  END-IF
*
END-REPEAT
*
DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
  MOVE '*' TO #MARK
END-SUBROUTINE
*
END
```

2. Führen Sie das Programm mit `RUN` aus.
3. Geben Sie in der daraufhin erscheinenden Map "JONES" ein und drücken Sie EINGABE.

   Die Liste der Mitarbeiter sollte jetzt eine zusätzliche Spalte enthalten.
4. Geben Sie `EDIT` in der `MORE`-Zeile ein, um zum Programmeditor zurückzukehren.
5. Speichern Sie das Programm mit `STOW`.
6. Geben Sie einen Punkt (.) in der Kommandozeile ein, um zum Menü **Development Functions** zurückzukehren.

Sie können nun mit den nächsten Übungen fortfahren: *Verarbeitungsregeln und Helproutinen*.