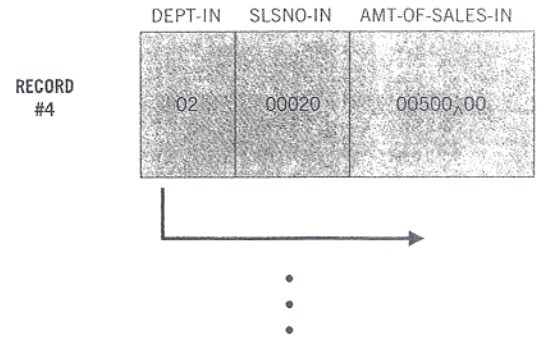


This section has focused on definitions related to control break processing and on an actual illustration of the output produced by a control break procedure. In the next section, we focus on ways to code this procedure. You may wish to examine Figure 10.2 for the pseudocode planning tool used to prepare the program. Study the pseudocode carefully so that you understand the structure to be used in the programs. The program that will perform the above control break procedure is shown in Figure 10.3 and will be discussed in detail in the next section, along with the hierarchy chart in Figure 10.4.

Figure 10.2 Pseudocode for sample control break procedure.

```
MAIN-MODULE
START
    Open the Files
    PERFORM Heading-Rtn
    PERFORM UNTIL no more records
        READ a Record
            AT END
                Move 'NO ' to Are-There-More-Records
            NOT AT END
                PERFORM Detail-Rtn
        END-READ
    END-PERFORM
    PERFORM End-Of-Job-Rtn
STOP

DETAIL-RTN
    EVALUATE TRUE
        WHEN First-Record = 'YES'
            Move Dept No to Hold Area
            Move 'NO ' to First-Record
        WHEN there is a change in Dept No
            PERFORM Control-Break-Rtn
    END-EVALUATE
    IF Line Counter > 25
    THEN
        PERFORM Heading-Rtn
    END-IF
    Move Input Data to Detail Line
    Write a Record
    Add 1 to Line Counter
    Add Amt to Dept Total

CONTROL-BREAK-RTN
    Move Dept Total to Output
    Write Summary Line
    Add 1 to Line Counter
    Initialize Dept Total
    Store Dept No

HEADING-RTN
    Write Headings
    Initialize Line Counter

END-OF-JOB-RTN
    Write last Dept Total
    Close Files
```

Figure 10.3 Program for sample control break procedure.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. SAMPLE.
*****************************************************************
*  The program creates a departmental sales report using a      *
*  control break procedure.  Comments are printed in            *
*  lowercase to set them apart from the program                 *
*  instructions. ORGANIZATION IS LINE SEQUENTIAL is a           *
*  clause used with all sequential files processed by PCs.      *
*****************************************************************
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT SALES-IN  ASSIGN TO 'C:\CHAPTER10\DATA103.DAT'
            ORGANIZATION IS LINE SEQUENTIAL.
     SELECT PRINT-OUT ASSIGN TO PRINTER
            ORGANIZATION IS LINE SEQUENTIAL.
*
 DATA DIVISION.
 FILE SECTION.
 FD  SALES-IN.
 01  SALES-REC-IN.
     05  DEPT-IN                 PIC XX.
     05  SLSNO-IN                PIC X(5).
     05  AMT-OF-SALES-IN         PIC 9(4)V99.
 FD  PRINT-OUT.
 01  PRINT-REC                   PIC X(100).
 WORKING-STORAGE SECTION.
 01  WORK-AREAS.
     05  ARE-THERE-MORE-RECORDS  PIC X(3)      VALUE 'YES'.
     05  FIRST-RECORD            PIC X(3)      VALUE 'YES'.
     05  WS-HOLD-DEPT            PIC XX        VALUE ZEROS.
     05  WS-DEPT-TOTAL           PIC 9(5)V99   VALUE ZEROS.
     05  WS-LINE-CT              PIC 99        VALUE ZEROS.
     05  WS-PAGE-CT              PIC 99        VALUE ZEROS.
 01  HEADING-1.
     05                          PIC X(49)     VALUE SPACES.
     05                          PIC X(21)
       VALUE 'MONTHLY STATUS REPORT'.
     05                          PIC X(9)      VALUE SPACES.
     05                          PIC X(5)      VALUE 'PAGE'.
     05  HL-PAGE-NO-OUT          PIC 99.
     05                          PIC X(14)     VALUE SPACES.
 01  HEADING-2.
     05                          PIC X(10)     VALUE SPACES.
     05                          PIC X(10)     VALUE 'DEPT'.
     05                          PIC X(20)
       VALUE 'SALESPERSON NO'.
     05                          PIC X(12)
       VALUE 'AMT OF SALES'.
     05                          PIC X(48)     VALUE SPACES.
 01  DETAIL-LINE.
     05                          PIC X(11)     VALUE SPACES.
     05  DL-DEPT-OUT             PIC XX.
     05                          PIC X(9)      VALUE SPACES.
     05  DL-SLSNO-OUT            PIC X(5).
     05                          PIC X(14)     VALUE SPACES.
     05  DL-AMT-OF-SALES-OUT     PIC $$,$$$.99.
     05                          PIC X(50)     VALUE SPACES.
 01  GROUP-REC.
     05                          PIC X(60)     VALUE SPACES.
     05                          PIC X(18)
       VALUE 'TOTAL FOR DEPT IS '.
     05  DEPT-TOTAL-OUT          PIC $$$,$$$.99.
     05                          PIC X(16)     VALUE SPACES.
*
```

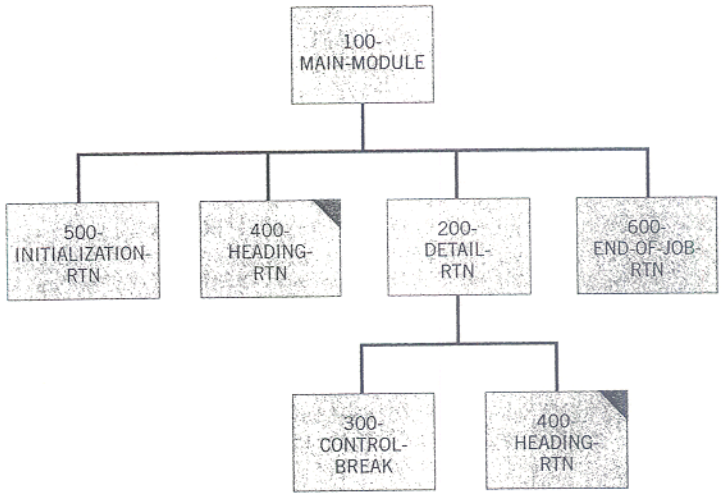
Figure 10.3 (continued)

```
 PROCEDURE DIVISION.
****************************************
*  Controls direction of program logic.  *
****************************************
 100-MAIN-MODULE.
     PERFORM 500-INITIALIZATION-RTN
     PERFORM 400-HEADING-RTN
     PERFORM UNTIL ARE-THERE-MORE-RECORDS = 'NO '
         READ SALES-IN
             AT END
                 MOVE 'NO ' TO ARE-THERE-MORE-RECORDS
             NOT AT END
                 PERFORM 200-DETAIL-RTN
         END-READ
     END-PERFORM
     PERFORM 600-END-OF-JOB-RTN
     STOP RUN.
*************************************************************
*  Performed from 100-main-module. Controls department     *
*  break and pagination. The first instruction moves the   *
*  first record's Dept No to the hold area.                *
*************************************************************
 200-DETAIL-RTN.
     EVALUATE TRUE
         WHEN FIRST-RECORD ='YES'
             MOVE DEPT-IN TO WS-HOLD-DEPT
             MOVE 'NO ' TO FIRST-RECORD
         WHEN DEPT-IN NOT = WS-HOLD-DEPT
             PERFORM 300-CONTROL-BREAK
     END-EVALUATE
     IF WS-LINE-CT > 25
         PERFORM 400-HEADING-RTN
     END-IF
     MOVE DEPT-IN TO DL-DEPT-OUT
     MOVE SLSNO-IN TO DL-SLSNO-OUT
     MOVE AMT-OF-SALES-IN TO DL-AMT-OF-SALES-OUT
     WRITE PRINT-REC FROM DETAIL-LINE
         AFTER ADVANCING 2 LINES
     ADD 1 TO WS-LINE-CT
     ADD AMT-OF-SALES-IN TO WS-DEPT-TOTAL.
*********************************************************
*  Performed from 200-detail-rtn, prints                *
*  department totals, resets control fields & totals.   *
*********************************************************
 300-CONTROL-BREAK.
     MOVE WS-DEPT-TOTAL TO DEPT-TOTAL-OUT
     WRITE PRINT-REC FROM GROUP-REC
         AFTER ADVANCING 2 LINES
     ADD 1 TO WS-LINE-CT
     MOVE ZEROS TO WS-DEPT-TOTAL
     MOVE DEPT-IN TO WS-HOLD-DEPT.
***********************************************
*  Performed from 100-main-module 200-detail-rtn  *
*  Prints out headings and resets line counter.   *
***********************************************
 400-HEADING-RTN.
     ADD 1 TO WS-PAGE-CT
     MOVE WS-PAGE-CT TO HL-PAGE-NO-OUT
     WRITE PRINT-REC FROM HEADING-1
         AFTER ADVANCING PAGE
     WRITE PRINT-REC FROM HEADING-2
         AFTER ADVANCING 2 LINES
     MOVE ZEROS TO WS-LINE-CT.
```

Figure 10.3 (continued)

```
*************************************************
*  Performed from 100-main-module. Opens files.  *
*************************************************
 500-INITIALIZATION-RTN.
     OPEN INPUT  SALES-IN
          OUTPUT PRINT-OUT.
*********************************************************
*  Performed from 100-main-module, performs end-of-job  *
*  functions and closes files                           *
*********************************************************
 600-END-OF-JOB-RTN.
*********************************************************
*  The following 2 instructions force the printing of   *
*  the last control total after an at end has occurred  *
*********************************************************
     MOVE WS-DEPT-TOTAL TO DEPT-TOTAL-OUT
     WRITE PRINT-REC FROM GROUP-REC
         AFTER ADVANCING 2 LINES
     CLOSE SALES-IN
           PRINT-OUT.
```

Figure 10.4 Hierarchy chart for sample control break procedure.



DEBUGGING TIP FOR EFFICIENT PROGRAM TESTING

Note that a control break procedure is used if records are in sequence by a control field and the number of records in each control field is variable. If we know in advance the number of records in each control field, you may use a `PERFORM ... TIMES` or a `PERFORM ... VARYING` instead. In Figure 10.3, if there were always 20 salespeople in each department, we could code the following:

```
200-DETAIL-RTN.
    PERFORM 250-READ-ADD-AND-PRINT 20 TIMES
    PERFORM 300-CONTROL-BREAK.
```

`250-READ-ADD-AND-PRINT` would read each record, print the record, and add the sales amount to `WS-DEPT-TOTAL`. `300-CONTROL-BREAK` would appear as in Figure 10.3.