第９章　事務計算アルゴリズム（応用編）

指導目標

事務処理のプログラムを作成する上での、基本的なプログラミング技法について学習する。

事務処理には、印刷処理（改ページ）、集計処理（コントロールブレイク）、ファイル更新処理（マッチング）などがある。これらの処理をプログラミングする場合、最も重要はことは処理を良く理解してフローチャートなどに表現することである。フローチャートに表現できれば、各言語の規約に従ってコーディングするだけである。

また、各処理には基本的（古典的）なパターンがあり、このパターンを自分のものにしていく事が重要である。自分のものにするためには、パターンに慣れさせる必要があり、これら基本パターンの例題を数多く取り上げ、じっくりと指導する。

内容のあらまし

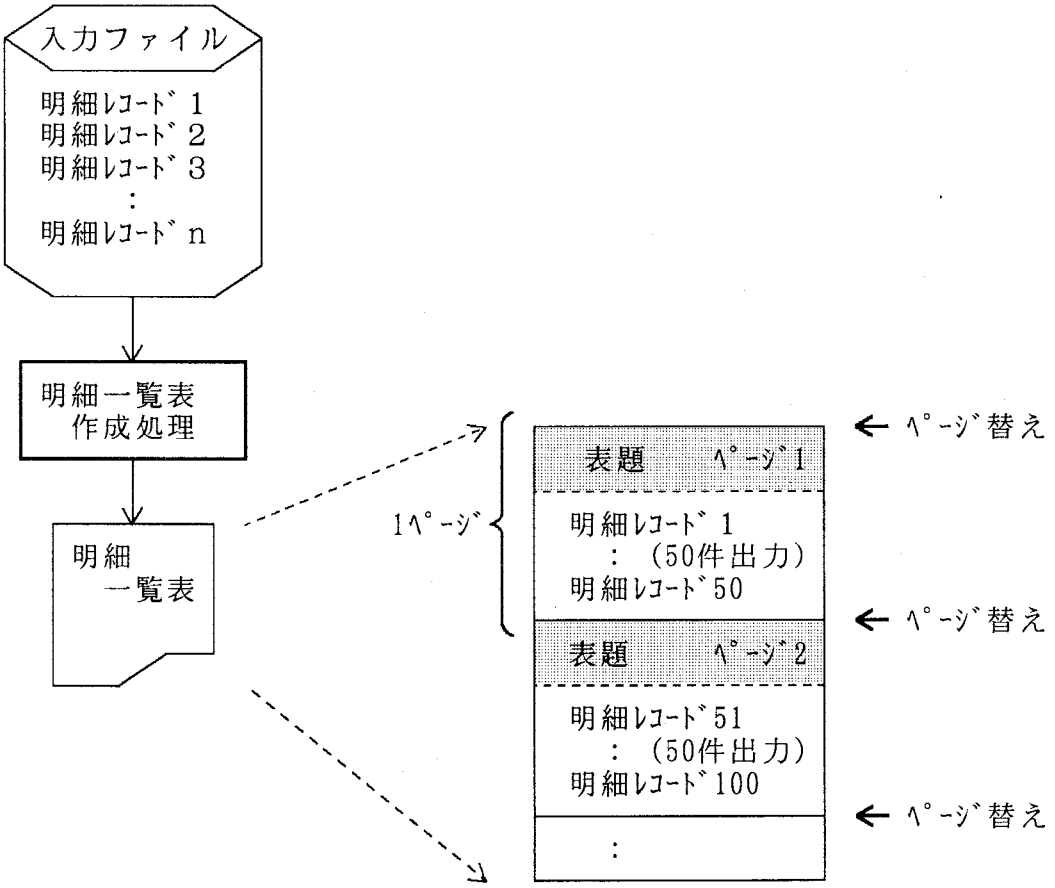
内　　容	説　明	議　論	机上実習	計算機実習
ページ替え	帳票の作成に関連する知識とWRITE文の行送りを説明する。		ｺﾝﾄﾛｰﾙﾌﾞﾚｲｸ条件を増やして複雑な例題へと展開させる。	実際に欠かせないｱﾙｺﾞﾘｽﾞﾑであり簡単な例題から複雑な例題へと展開する。
シーケンスﾁｪｯｸ	キー項目の新旧の入れ替えのﾀｲﾐﾝｸﾞとその方法を説明する。			
ヒストグラム	配列の扱いと印刷する時の考え方を横、縦のﾋｽﾄｸﾞﾗﾑで説明する。		配列の扱いについて図解等で説明できるようにする。	
ファイルの突合せ	マッチング処理について１対１対ｎの考え方を説明する。		図解と具体的な値で処理の流れを自習する。	
ファイルの更新	図解等で処理をよく理解させ、順ﾌｧｲﾙと索引ﾌｧｲﾙでの違いを説明する。	ﾌｧｲﾙ編成と更新処理の関係を議論する。	例題を与えて適切な更新処理とは何かを考えさせる。	
事例	処理の概要を解説し、ﾌﾟﾛｸﾞﾗﾑを読ませ問題点や改良点を説明する。	問題点や改良点を見つけて議論する。	より簡潔なｺｰﾃﾞｨﾝｸﾞに書き直させる。	簡潔になったﾌﾟﾛｸﾞﾗﾑを実行させる。

第９章　事務計算アルゴリズム

９．１　ページ替え

改ページは、ＷＲＩＴＥ文のＡＤＶＡＮＣＩＮＧ　ＰＡＧＥ指定で行う。
改行は、ＡＤＶＡＮＣＩＮＧ　ＬＩＮＥＳ指定で行う。
レコードを書き出す前に、改ページまたは改行を行う場合は、ＡＦＴＥＲを指定する。
書き出した後で行う場合はＢＥＦＯＲＥを指定する。

```
WRITE  ﾚｺｰﾄﾞ名  FROM  一意名１

      { AFTER  }             { 一意名２／定数  LINES }
      { BEFORE } ADVANCING   { PAGE                   }
```



コントロールブレークとは、制御の流れを変えることである。集計処理では、レコードのキー項目の値が同じ間はデータ項目の合計処理を行い、キー項目が変わるとそれまでの合計値を印刷する。このように繰り返し処理がある条件で変化することをコントロールブレークが発生したという。
集計処理のある条件とは、キー項目の値が変わった場合であり、他に集計レコードの終了もコントロールブレークの条件になる。

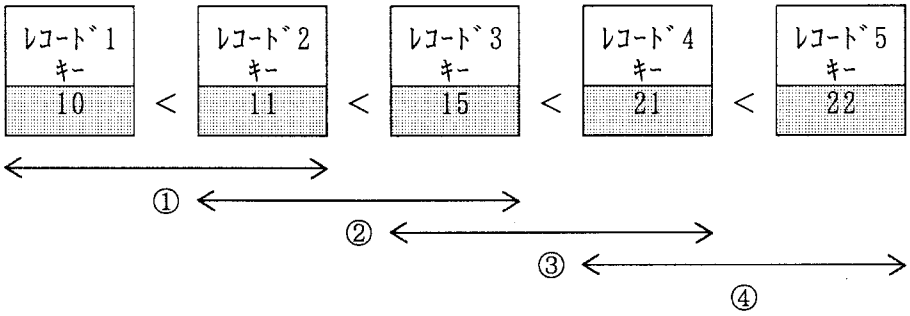
ページ替えのプログラム例

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  頁替え.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT 入力ﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
    SELECT 明細ﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
DATA DIVISION.
FILE SECTION.
FD  入力ﾌｧｲﾙ.
01  入力ﾚｺｰﾄﾞ.
    03  FILLER  PIC X(80).
FD  明細ﾌｧｲﾙ.
01  明細ﾚｺｰﾄﾞ.
    03  FILLER  PIC X(80).
WORKING-STORAGE SECTION.
01  終了ﾌﾗｸﾞ     PIC X(3) VALUE "OFF".
    88  終了               VALUE "ON".
01  最大行数     PIC 9(3) VALUE 10.
01  行数         PIC 9(3) VALUE 99.
01  頁数         PIC 9(3) VALUE 0.

01 表題.
    03           PIC X(01).
    03           PIC X(50) VALUE "          明細一覧表      ".
    03  表題頁   PIC ZZ9B.
    03           PIC X(5)  VALUE "ﾍﾟｰｼﾞ".

PROCEDURE DIVISION.
開始.
    OPEN INPUT 入力ﾌｧｲﾙ, OUTPUT 明細ﾌｧｲﾙ
    READ 入力ﾌｧｲﾙ
       AT END  SET 終了 TO TRUE
    END-READ
    PERFORM UNTIL 終了
        IF 行数 > 最大行数 THEN
            COMPUTE 表題頁, 頁数 = 頁数 + 1
            WRITE 明細ﾚｺｰﾄﾞ FROM 表題 AFTER ADVANCING PAGE
            MOVE SPACE TO 明細ﾚｺｰﾄﾞ
            WRITE 明細ﾚｺｰﾄﾞ AFTER ADVANCING 1
            MOVE ZERO TO 行数
         END-IF
         WRITE 明細ﾚｺｰﾄﾞ FROM 入力ﾚｺｰﾄﾞ AFTER ADVANCING 1
         COMPUTE 行数 = 行数 + 1
         READ 入力ﾌｧｲﾙ
            AT END  SET 終了 TO TRUE
         END-READ
    END-PERFORM
    CLOSE 入力ﾌｧｲﾙ, 明細ﾌｧｲﾙ
    STOP RUN.
```

９．２　シーケンスチェック

レコードのキー項目が、整列済み（昇順、降順）であることをチェックする。

レコードを入力してキー項目を保存して置き、次のレコードを入力して保存してある前のレコードのキー項目と比較する。この処理をファイルが終了するまで繰り返す。



プログラム作成上の注意としては、ファイル中の最初のレコードの取扱い、そして、キー項目の入れ替え、ファイル中の最後のレコードの取扱い方である。また、レコードが１件の場合でも正しく動作することも重要である。

シーケンスチェックのプログラム例

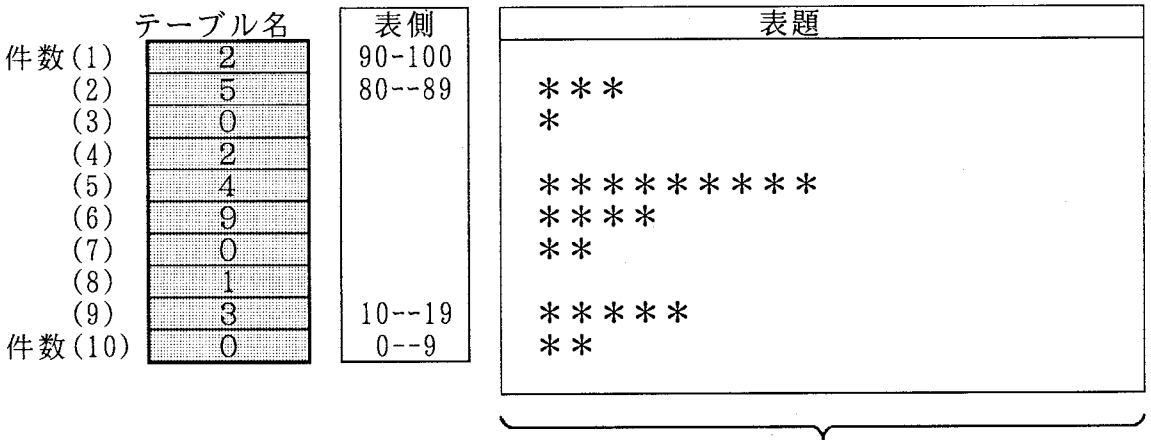
```
IDENTIFICATION DIVISION.
PROGRAM-ID. 整列済み.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT 入力ﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
DATA DIVISION.
FILE SECTION.
FD  入力ﾌｧｲﾙ.
01  入力ﾚｺｰﾄﾞ.
    03              PIC X(28).
    03  キー値      PIC X(10).
    03              PIC X(42).
WORKING-STORAGE SECTION.
01  前キー値        PIC X(10).
01  現キー値        PIC X(10).

PROCEDURE DIVISION.
開始.
    OPEN INPUT 入力ﾌｧｲﾙ
    READ 入力ﾌｧｲﾙ
        AT END     MOVE  HIGH-VALUE TO 現キー値
        NOT AT END MOVE キー値      TO 現キー値
    END-READ
    MOVE 現キー値 TO 前キー値
    PERFORM UNTIL 現キー値 = HIGH-VALUE
        READ 入力ﾌｧｲﾙ
            AT END     MOVE  HIGH-VALUE TO 現キー値
            NOT AT END MOVE キー値      TO 現キー値
        END-READ
        EVALUATE TRUE
            WHEN 前キー値 < 現キー値
                MOVE 現キー値 TO 前キー値
            WHEN 前キー値 = 現キー値
                DISPLAY "重複エラーがあります。処理を中断します。"
                MOVE  HIGH-VALUE TO 現キー値
            WHEN OTHER
                DISPLAY "シーケンスエラー発生、処理を中断します。"
                MOVE  HIGH-VALUE TO 現キー値
        END-EVALUATE
    END-PERFORM
    CLOSE 入力ﾌｧｲﾙ
    STOP RUN.
```

注：キー項目の値にHIGH-VALUEは存在しないものとしている。

９．３　ヒストグラム（histogram）の作成

集計データの結果をヒストグラムで出力する。

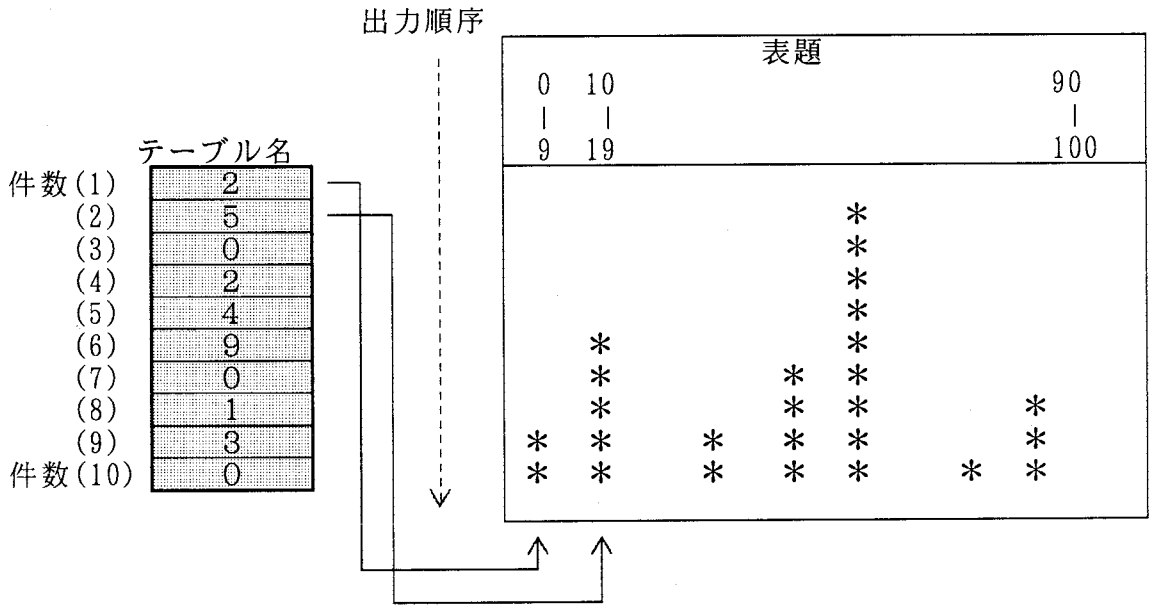
（１）横ヒストグラム



要素の順番と表側の順番に
注意すること。

星印を設定する項目も配列にする。
```
MOVE  "＊"  TO FIELD2(J)
```
配列を使用しないで部分文字列を使う場合
```
COMPUTE  K =  件数(I)  +  6
MOVE  ALL  "＊"  TO  ｸﾞﾗﾌﾚｺｰﾄﾞ(7:K)
```

（２）縦のヒストグラム



ヒストグラムのプログラム例（横ヒストグラム）

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  横グラフ.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT 入力ﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
    SELECT ｸﾞﾗﾌﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
DATA DIVISION.
FILE SECTION.
FD  入力ﾌｧｲﾙ.
01  入力ﾚｺｰﾄﾞ.
    03  機種名   PIC X(10).
    03  価格     PIC 9(07).
    03  FILLER   PIC X(13).
FD  ｸﾞﾗﾌﾌｧｲﾙ.
01  ｸﾞﾗﾌﾚｺｰﾄﾞ.
    03  FIELD1   PIC X(06).
    03            PIC X(01).
    03  FIELD2   PIC X      OCCURS 100 TIMES.
WORKING-STORAGE SECTION.
01  終了ﾌﾗｸﾞ      PIC X(3)  VALUE "OFF".
01  I             PIC 9(4)  COMPUTATIONAL.
01  J             PIC 9(4)  COMPUTATIONAL.
01  K             PIC 9(4)  COMPUTATIONAL.
01  集計ﾃｰﾌﾞﾙ.
    03  件数      PIC 9(3)  OCCURS  10 TIMES.

01  表題1         PIC X(50) VALUE "                横グラフ      ".
01  表題2.
    03             PIC X(27) VALUE "価格帯 1---+---10----+---20".
    03             PIC X(30) VALUE "----+---30----+---40----+---50".
    03             PIC X(30) VALUE "----+---60----+---70----+---80".
    03             PIC X(22) VALUE "----+---90----+----100".
01  表題3         PIC X(06) VALUE "(万円)".
01  表側定数.
    03             PIC X(6) VALUE "90-100".
    03             PIC X(6) VALUE "80--89".
    03             PIC X(6) VALUE "70--79".
    03             PIC X(6) VALUE "60--69".
    03             PIC X(6) VALUE "50--59".
    03             PIC X(6) VALUE "40--49".
    03             PIC X(6) VALUE "30--39".
    03             PIC X(6) VALUE "20--29".
    03             PIC X(6) VALUE "10--19".
    03             PIC X(6) VALUE " 0---9".
01  表側    REDEFINES  表側定数.
    03  価格帯   PIC X(6) OCCURS  10 TIMES.
```

（続く）

（続き）

```
PROCEDURE DIVISION.
開始.
    OPEN INPUT 入力ﾌｧｲﾙ, OUTPUT ｸﾞﾗﾌﾌｧｲﾙ
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 10
        MOVE ZERO TO 件数(I)
    END-PERFORM
    READ 入力ﾌｧｲﾙ
       AT END MOVE "ON" TO 終了ﾌﾗｸﾞ
    END-READ
    PERFORM UNTIL 終了ﾌﾗｸﾞ = "ON"
        COMPUTE I = (価格 / 100000) + 1
        IF I > 10 THEN  MOVE 10 TO I END-IF
        IF I =  0 THEN  MOVE  1 TO I END-IF
        COMPUTE I = 11 - I
        COMPUTE 件数(I) = 件数(I) + 1
        READ 入力ﾌｧｲﾙ
            AT END  MOVE "ON" TO 終了ﾌﾗｸﾞ
        END-READ
    END-PERFORM
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題1 AFTER PAGE
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題2 AFTER  2
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題3 AFTER  1
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 10
        MOVE SPACE TO ｸﾞﾗﾌﾚｺｰﾄﾞ
        MOVE 価格帯(I) TO FIELD1
        PERFORM VARYING J FROM 1 BY 1 UNTIL J > 100
                                         OR J > 件数(I)
            MOVE "*" TO  FIELD2(J)
        END-PERFORM
        WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ AFTER  1
    END-PERFORM
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題2 AFTER  1

    CLOSE 入力ﾌｧｲﾙ, ｸﾞﾗﾌﾌｧｲﾙ
    STOP RUN.
```

ヒストグラムのプログラム例（縦ヒストグラム）

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  縦グラフ.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT 入力ﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
    SELECT ｸﾞﾗﾌﾌｧｲﾙ ASSIGN TO 作成者語
                    ORGANIZATION IS SEQUENTIAL.
DATA DIVISION.
FILE SECTION.
FD  入力ﾌｧｲﾙ.
01  入力ﾚｺｰﾄﾞ.
    03  機種名   PIC X(10).
    03  価格     PIC 9(07).
    03  FILLER   PIC X(13).
FD  ｸﾞﾗﾌﾌｧｲﾙ.
01  ｸﾞﾗﾌﾚｺｰﾄﾞ.
    03  番号     PIC ZZZZ.
    03  記号     PIC X(1).
    03  FIELD2   OCCURS  10 TIMES.
        05          PIC X.
        05  FLD2    PIC X.
        05          PIC X(3).
WORKING-STORAGE SECTION.
01  終了ﾌﾗｸﾞ     PIC X(3)  VALUE "OFF".
01  I            PIC 9(4)  COMPUTATIONAL.
01  J            PIC 9(4)  COMPUTATIONAL.
01  K            PIC 9(4)  COMPUTATIONAL.
01  集計ﾃｰﾌﾞﾙ.
    03  件数     PIC 9(3)  OCCURS  10 TIMES.

01  表題1        PIC X(50) VALUE "                縦グラフ      ".
01  表題2.
    03           PIC X(05) VALUE "    +".
    03           PIC X(30) VALUE "------------------------------".
    03           PIC X(18) VALUE "-----------------+".
01  表題3.
    03           PIC X(05) VALUE "     ".
    03           PIC X(30) VALUE " 0   10   20   30   40   50   ".
    03           PIC X(17) VALUE "60   70   80   90".
01  表題4.
    03           PIC X(05) VALUE "     ".
    03           PIC X(30) VALUE " |    |    |    |    |    |   ".
    03           PIC X(17) VALUE " |    |    |    |".
01  表題5.
    03           PIC X(05) VALUE "     ".
    03           PIC X(30) VALUE " 9   19   29   39   49   59   ".
    03           PIC X(17) VALUE "69   79   89  100".
```

（続く）

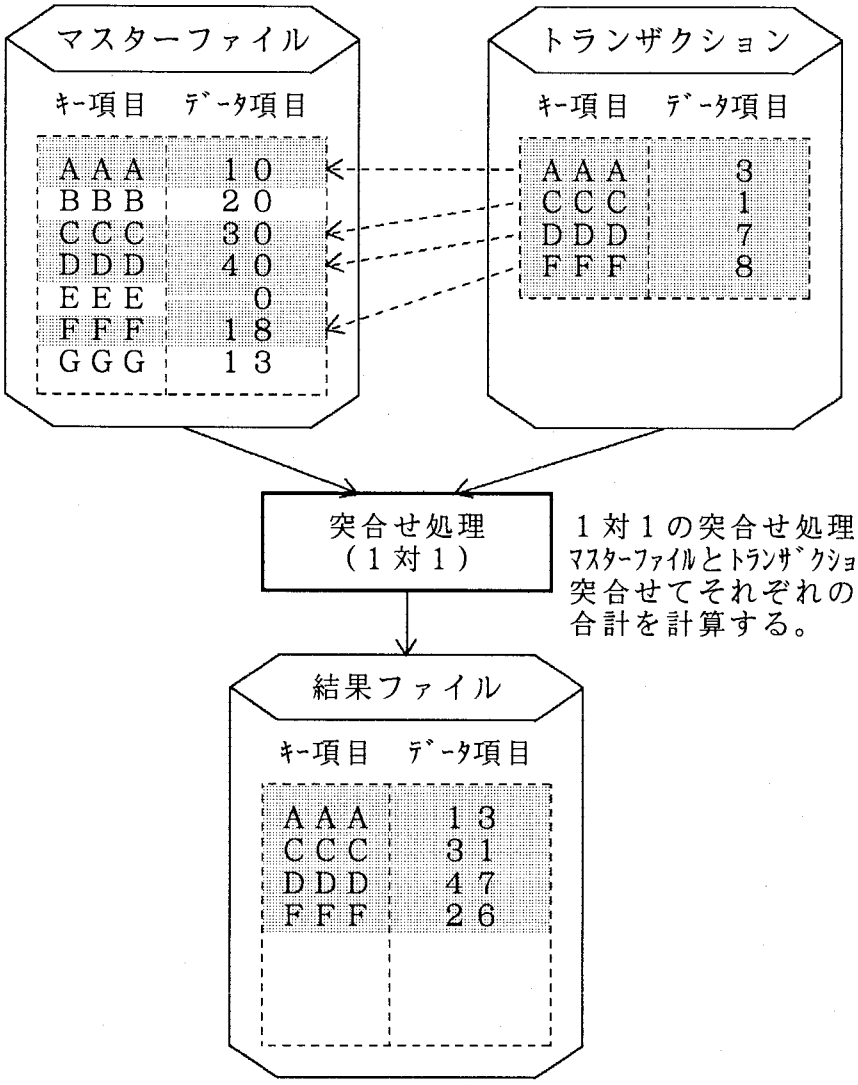
```
PROCEDURE DIVISION.
開始.
    OPEN INPUT 入力ﾌｧｲﾙ, OUTPUT ｸﾞﾗﾌﾌｧｲﾙ
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 10
        MOVE ZERO TO 件数(I)
    END-PERFORM
    READ 入力ﾌｧｲﾙ
       AT END MOVE "ON" TO 終了ﾌﾗｸﾞ
    END-READ
    PERFORM UNTIL 終了ﾌﾗｸﾞ = "ON"
        COMPUTE I = (価格 / 100000) + 1
        IF I > 10 THEN  MOVE 10 TO I END-IF
        IF I =  0 THEN  MOVE  1 TO I END-IF
        COMPUTE 件数(I) = 件数(I) + 1
        READ 入力ﾌｧｲﾙ
            AT END  MOVE "ON" TO 終了ﾌﾗｸﾞ
        END-READ
    END-PERFORM
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題1 AFTER PAGE
    PERFORM VARYING I FROM 100 BY -1 UNTIL I = 0
        MOVE SPACE TO ｸﾞﾗﾌﾚｺｰﾄﾞ
        MOVE "|"   TO 記号
        DIVIDE I BY 10 GIVING J  REMAINDER K
        IF K = 0  THEN  MOVE I TO 番号  END-IF
        PERFORM VARYING J FROM 1 BY 1 UNTIL J > 10
            IF 件数(J) >= I THEN  MOVE "*" TO  FLD2(J) END-IF
        END-PERFORM
        WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ AFTER  1
    END-PERFORM
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題2 AFTER  1
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題3 AFTER  1
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題4 AFTER  1
    WRITE ｸﾞﾗﾌﾚｺｰﾄﾞ FROM 表題5 AFTER  1

    CLOSE 入力ﾌｧｲﾙ, ｸﾞﾗﾌﾌｧｲﾙ
    STOP RUN.
```

９．４　ファイルの突合せ

別々のファイルの特定のキー項目による突き合わせを行い、キー項目が一致するか否かの処理を行うことを「突合せ（matching）処理」という。
突合せを行うファイルのレコードは、それぞれ一つ以上のキー項目がある。突合せ処理のファイルのレコードは、そのキー項目が昇順（ascending）、降順（descending）のいずれかで整列（分類）されている必要がある。

９．４．１　１対１の突合せ処理

突合せ処理を行うファイルで、キー項目が同じ値を持つレコードが他にない場合に「１対１の突合せ」という。



１対１の突合せ処理の例として、ﾏｽﾀｰﾌｧｲﾙとﾄﾗﾝｻﾞｸｼｮﾝのｷｰ項目で突合せてそれぞれのﾃﾞｰﾀ項目の合計を計算する。

1対1の突合せ処理のプログラム例

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  1対1.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT ﾏｽﾀｰﾌｧｲﾙ  ASSIGN TO 作成者語
                     ORGANIZATION IS SEQUENTIAL.
    SELECT ﾄﾗﾝｻﾞｸｼｮﾝ ASSIGN TO 作成者語
                     ORGANIZATION IS SEQUENTIAL.
    SELECT 結果ﾌｧｲﾙ  ASSIGN TO 作成者語
                     ORGANIZATION IS SEQUENTIAL.
DATA DIVISION.
FILE SECTION.
FD  ﾏｽﾀｰﾌｧｲﾙ.
01  ﾏｽﾀｰﾚｺｰﾄﾞ.
    03  FILLER       PIC X(10).
FD  ﾄﾗﾝｻﾞｸｼｮﾝ.
01  ﾄﾗﾝｻﾞｸｼｮﾝﾚｺｰﾄﾞ.
    03  FILLER       PIC X(10).
FD  結果ﾌｧｲﾙ.
01  結果ﾚｺｰﾄﾞ.
    03  Xキー値      PIC X(06).
    03  合計項目     PIC 9(04).
WORKING-STORAGE SECTION.
01  ﾏｽﾀｰ領域.
    03  Mキー値      PIC X(06).
    03  ﾃﾞｰﾀ項目     PIC 9(04).
01  ﾄﾗﾝｻﾞｸｼｮﾝ領域.
    03  Tキー値      PIC X(06).
    03  ﾃﾞｰﾀ項目     PIC 9(04).
```

（続く）

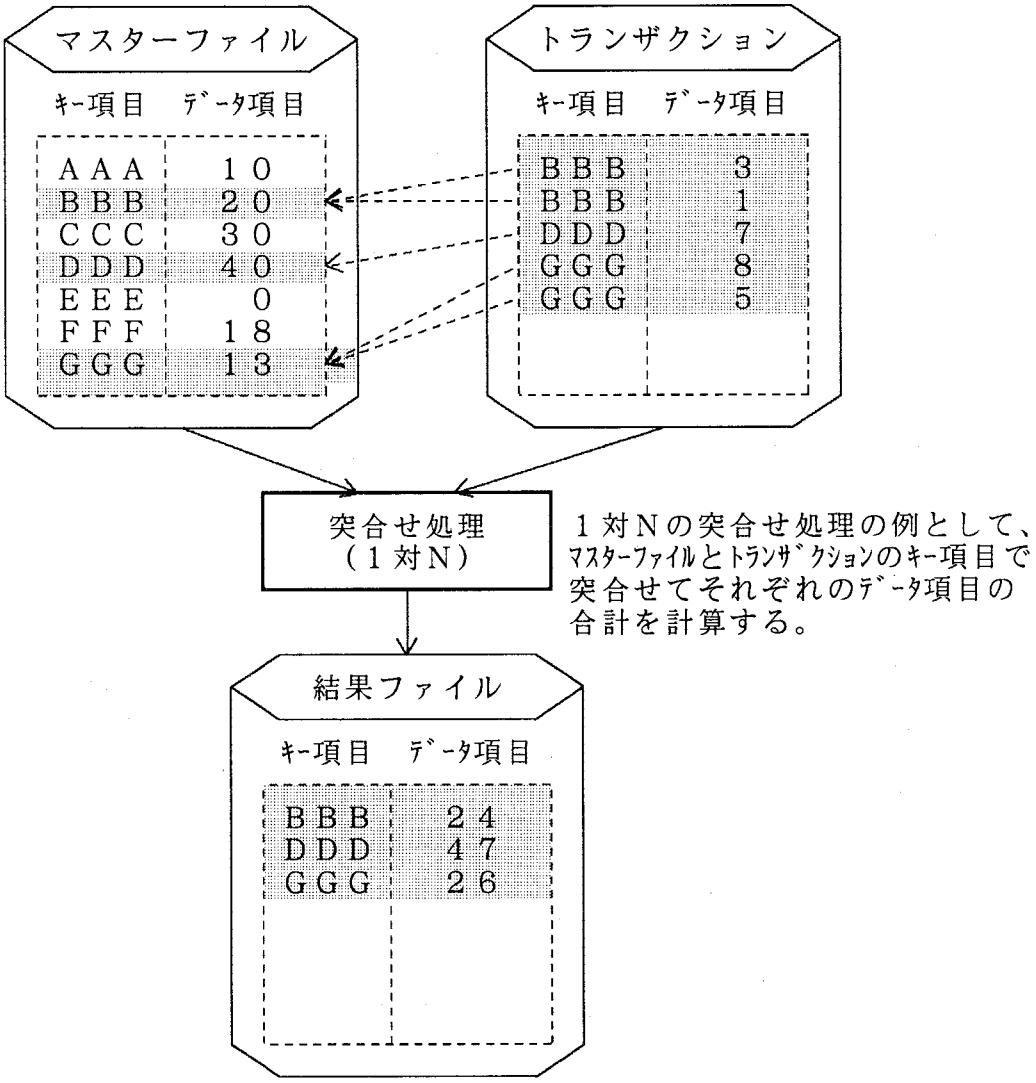
（続き）

```
PROCEDURE DIVISION.
開始.
    OPEN INPUT  ﾏｽﾀｰﾌｧｲﾙ, ﾄﾗﾝｻﾞｸｼｮﾝ
         OUTPUT 結果ﾌｧｲﾙ
    READ ﾏｽﾀｰﾌｧｲﾙ INTO ﾏｽﾀｰ領域
        AT END  MOVE  HIGH-VALUE TO Ｍキー値
    END-READ
    READ ﾄﾗﾝｻﾞｸｼｮﾝ INTO ﾄﾗﾝｻﾞｸｼｮﾝ領域
        AT END  MOVE  HIGH-VALUE TO Ｔキー値
    END-READ
    PERFORM UNTIL Ｔキー値 = HIGH-VALUE
        IF Ｍキー値 < Ｔキー値
            READ ﾏｽﾀｰﾌｧｲﾙ INTO ﾏｽﾀｰ領域
                AT END  MOVE  HIGH-VALUE TO Ｍキー値
            END-READ
        ELSE
            PERFORM
                IF Ｍキー値 = Ｔキー値
                    MOVE  Ｍキー値 TO  Ｘキー値
                    COMPUTE 合計項目 = ﾃﾞｰﾀ項目 OF ﾏｽﾀｰ領域
                                     + ﾃﾞｰﾀ項目 OF ﾄﾗﾝｻﾞｸｼｮﾝ領域
                    WRITE 結果ﾚｺｰﾄﾞ
                    READ ﾏｽﾀｰﾌｧｲﾙ INTO ﾏｽﾀｰ領域
                       AT END  MOVE  HIGH-VALUE TO Ｍキー値
                    END-READ
                    READ ﾄﾗﾝｻﾞｸｼｮﾝ INTO ﾄﾗﾝｻﾞｸｼｮﾝ領域
                       AT END  MOVE  HIGH-VALUE TO Ｔキー値
                    END-READ
                ELSE
                    DISPLAY "処理中止（Ｔｷｰがﾏｽﾀｰに存在しません）"
                    MOVE  HIGH-VALUE TO Ｔキー値
                END-IF
            END-PERFORM
        END-IF
    END-PERFORM
    CLOSE ﾏｽﾀｰﾌｧｲﾙ, ﾄﾗﾝｻﾞｸｼｮﾝ, 結果ﾌｧｲﾙ
    STOP RUN.
```

注：キー項目にHIGH-VALUEは存在しないものとする。

９．４．２　１対Ｎの突合せ処理

突合せ処理を行うファイルで、一方のファイル（例ではマスターファイル）にはキー項目が重複するレコードがなく、他方のファイル（例ではトランザクション）に、複数個の同じキー項目の値を持つレコードある場合の突合せを、「１対Ｎの突合せ」という。



1対Nの突合せ処理のプログラム例

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  1対N.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT ﾏｽﾀｰﾌｧｲﾙ  ASSIGN TO 作成者語
                     ORGANIZATION IS SEQUENTIAL.
    SELECT ﾄﾗﾝｻﾞｸｼｮﾝ ASSIGN TO 作成者語
                     ORGANIZATION IS SEQUENTIAL.
    SELECT 結果ﾌｧｲﾙ  ASSIGN TO 作成者語
                     ORGANIZATION IS SEQUENTIAL.
DATA DIVISION.
FILE SECTION.
FD  ﾏｽﾀｰﾌｧｲﾙ.
01  ﾏｽﾀｰﾚｺｰﾄﾞ.
    03  FILLER       PIC X(10).
FD  ﾄﾗﾝｻﾞｸｼｮﾝ.
01  ﾄﾗﾝｻﾞｸｼｮﾝﾚｺｰﾄﾞ.
    03  FILLER       PIC X(10).
FD  結果ﾌｧｲﾙ.
01  結果ﾚｺｰﾄﾞ.
    03  Xキー値      PIC X(06).
    03  合計項目     PIC 9(04).
WORKING-STORAGE SECTION.
01  ﾏｽﾀｰ領域.
    03  Mキー値      PIC X(06).
    03  ﾃﾞｰﾀ項目     PIC 9(04).
01  ﾄﾗﾝｻﾞｸｼｮﾝ領域.
    03  Tキー値      PIC X(06).
    03  ﾃﾞｰﾀ項目     PIC 9(04).
```

（続く）

（続き）

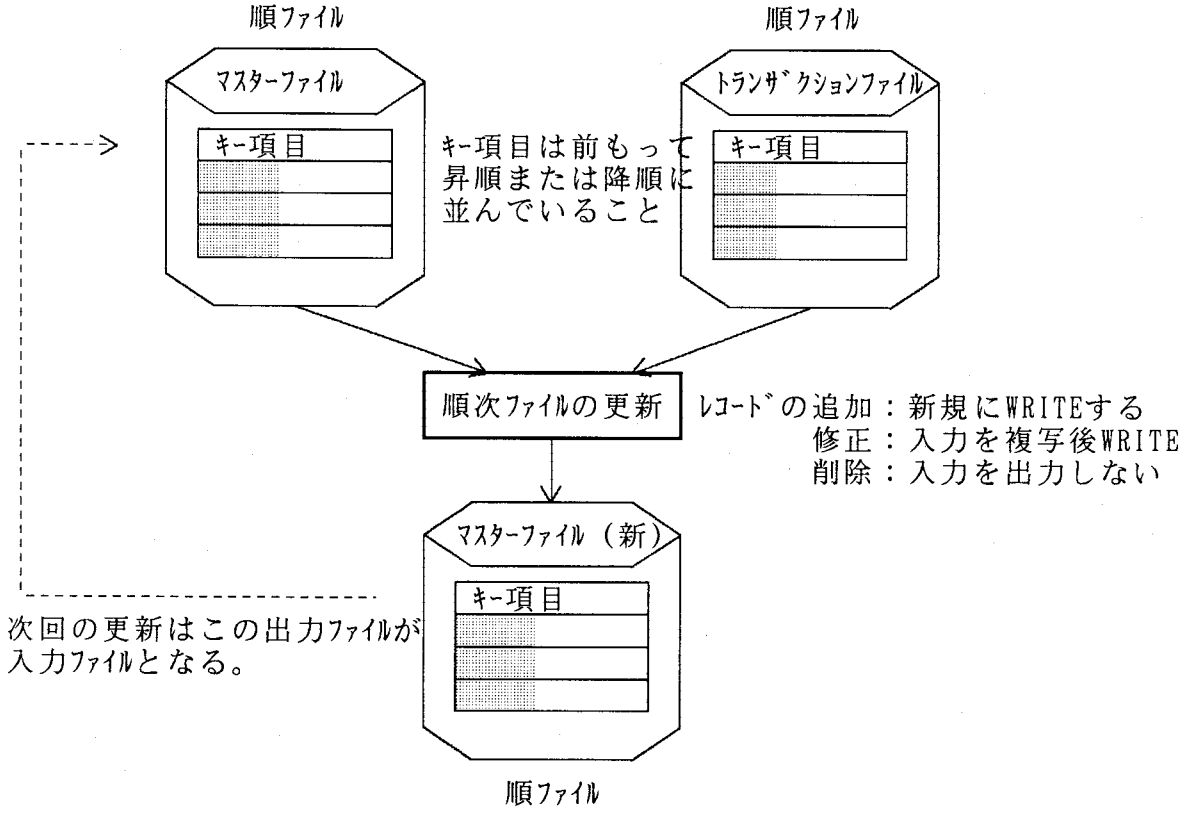
```
PROCEDURE DIVISION.
 開始.
     OPEN INPUT  ﾏｽﾀｰﾌｧｲﾙ, ﾄﾗﾝｻﾞｸｼｮﾝ
          OUTPUT 結果ﾌｧｲﾙ
     READ ﾏｽﾀｰﾌｧｲﾙ INTO ﾏｽﾀｰ領域
         AT END  MOVE  HIGH-VALUE TO Mキー値
     END-READ
     READ ﾄﾗﾝｻﾞｸｼｮﾝ INTO ﾄﾗﾝｻﾞｸｼｮﾝ領域
         AT END  MOVE  HIGH-VALUE TO Tキー値
     END-READ
     PERFORM UNTIL Tキー値 = HIGH-VALUE
         IF Mキー値 < Tキー値
             READ ﾏｽﾀｰﾌｧｲﾙ INTO ﾏｽﾀｰ領域
                 AT END  MOVE  HIGH-VALUE TO Mキー値
             END-READ
         ELSE
             IF Mキー値 = Tキー値
                 MOVE Mキー値 TO  Xキー値
                 MOVE ﾃﾞｰﾀ項目 OF ﾏｽﾀｰ領域  TO  合計項目
                 PERFORM TEST AFTER UNTIL Mキー値 NOT = Tキー値
                     COMPUTE 合計項目 = 合計項目
                                      + ﾃﾞｰﾀ項目 OF ﾄﾗﾝｻﾞｸｼｮﾝ領域
                     READ ﾄﾗﾝｻﾞｸｼｮﾝ INTO ﾄﾗﾝｻﾞｸｼｮﾝ領域
                         AT END  MOVE  HIGH-VALUE TO Tキー値
                     END-READ
                 END-PERFORM
                 WRITE 結果ﾚｺｰﾄﾞ
                 READ ﾏｽﾀｰﾌｧｲﾙ INTO ﾏｽﾀｰ領域
                     AT END  MOVE  HIGH-VALUE TO Mキー値
                 END-READ
             ELSE
                 DISPLAY "シーケンスエラー"
                 READ ﾄﾗﾝｻﾞｸｼｮﾝ INTO ﾄﾗﾝｻﾞｸｼｮﾝ領域
                     AT END  MOVE  HIGH-VALUE TO Tキー値
                 END-READ
             END-IF
         END-IF
     END-PERFORM
     CLOSE ﾏｽﾀｰﾌｧｲﾙ, ﾄﾗﾝｻﾞｸｼｮﾝ, 結果ﾌｧｲﾙ
     STOP RUN.
```

注：キー項目にはHIGH-VALUEが存在しないものとする。

９．５　ファイルの更新

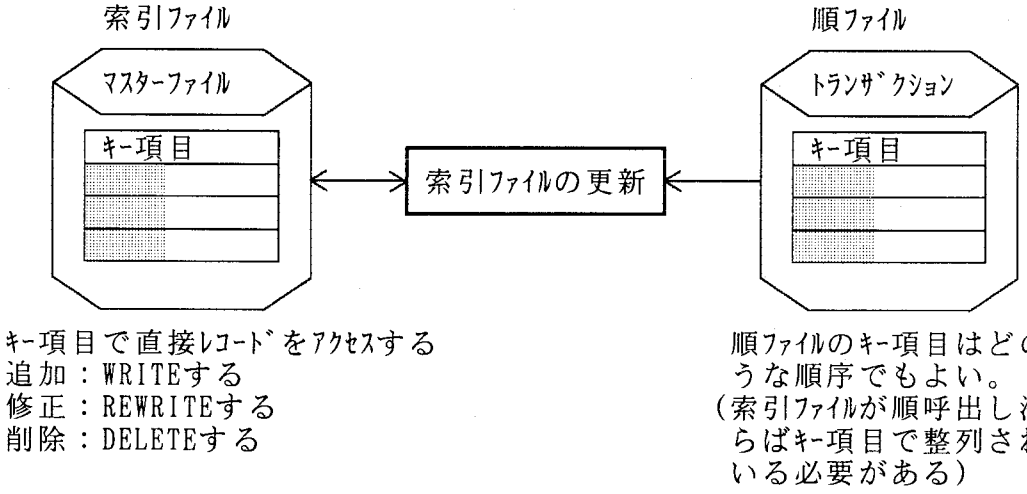
９．５．１　順ファイルの更新

突合せ処理の形態として、「併合（merge）処理」がある。併合処理は、レコードが特定のキー項目で整列（分類）済みの複数ファイルを、そのキー項目で突合せて一つのファイルにまとめる。



９．５．２　索引ファイルの更新

突合せ処理の形態として、「更新（update）処理」がある。更新処理は、トランザクションファイルのレコードの内容を、マスターファイルのレコードに対して追加、修正、削除を行いマスターファイルを更新する。



９．６　事例

具体的な事例としては、次のとおり。

参考：「ＣＯＢＯＬ自習の課題集」
全国ｺﾝﾋﾟｭｰﾀ･ｶﾚｯｼﾞ連絡協議会　西日本ﾌﾞﾛｯｸ教育部会編より

９．６．１　打撃成績表の作成

打撃ファイルを入力して打撃成績表を作成する。

①打撃ファイルの形式

項目	選手コード		選手名	打数	ヒット	ﾎｰﾑﾗﾝ	打点	空　白
	チーム	背番号						
桁数	2	2	１５	3	3	3	3	9
属性	X	X	X	9	9	9	9	X

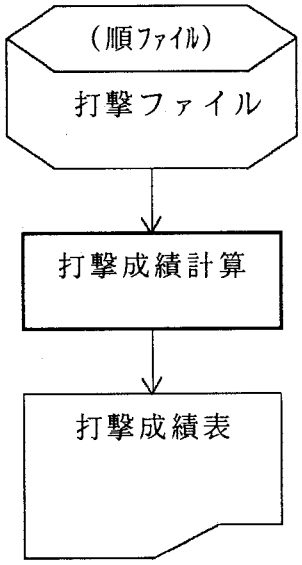
②処理内容

(1)打率は四捨五入して小数第３桁まで求める
　　打率＝ヒット÷打数

(2)大見出しと小見出しをつけて印刷する。

(3)明細は間に空白行をいれず連続して印刷する。

(4)打撃ファイルには１件以上のﾚｺｰﾄﾞが存在する。



③打撃成績表の形式

```
                    ﾀﾞｹﾞｷ　ｾｲｾｷ　ﾋｮｳ

ﾁｰﾑ　ｾﾊﾞﾝｺﾞｳ　ｾﾝｼｭﾒｲ              ﾀﾞｽｳ　ﾋｯﾄ　ﾎｰﾑﾗﾝ　ﾀﾞﾃﾝ　ﾀﾞﾘﾂ

 XX    XX      XXXXXXXXXXXXXXX    ZZ9   ZZ9    ZZ9    ZZ9   Z.999
 XX    XX      XXXXXXXXXXXXXXX    ZZ9   ZZ9    ZZ9    ZZ9   Z.999
 XX    XX      XXXXXXXXXXXXXXX    ZZ9   ZZ9    ZZ9    ZZ9   Z.999
 XX    XX      XXXXXXXXXXXXXXX    ZZ9   ZZ9    ZZ9    ZZ9   Z.999
                    :
                    :
                    :
 XX    XX      XXXXXXXXXXXXXXX    ZZ9   ZZ9    ZZ9    ZZ9   Z.999
```

「打撃成績表の作成」のプログラム例

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 打撃成績.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT SEI-FILE  ASSIGN TO 作成者語.
    SELECT ICH-FILE  ASSIGN TO 作成者語.
DATA DIVISION.
FILE SECTION.
FD  SEI-FILE
    BLOCK CONTAINS 10 RECORDS.
01  SEI-REC.
    03 SEI-CODE.
       05 SEI-TEAM  PIC X(02).
       05 SEI-BANGO PIC X(02).
    03 SEI-NAME     PIC X(15).
    03 SEI-DASU     PIC 9(03).
    03 SEI-HIT      PIC 9(03).
    03 SEI-HR       PIC 9(03).
    03 SEI-DATEN    PIC 9(03).
    03             PIC X(09).
FD  ICH-FILE
    LABEL RECORD OMITTED
    BLOCK CONTAINS  1 RECORDS.
01  ICH-REC        PIC X(132).
WORKING-STORAGE SECTION.
01  MIDASHI-1.
    03             PIC X(35) VALUE SPACE.
    03             PIC X(16) VALUE "ﾀﾞｹﾞｷ ｾｲｾｲｷ ﾋｮｳ".
01  MIDASHI-2.
    03             PIC X(20) VALUE  SPACE.
    03             PIC X(14) VALUE "ﾁｰﾑ  ｾﾊﾞﾝｺﾞｳ".
    03             PIC X(18) VALUE "ｾﾝｼｭﾒｲ".
    03             PIC X(13) VALUE "ﾀﾞｽｳ     ﾋｯﾄ".
    03             PIC X(17) VALUE "ﾎｰﾑﾗﾝ   ﾀﾞﾃﾝ".
    03             PIC X(04) VALUE "ﾀﾞﾘﾂ".
01  ICHIRAN.
    03             PIC X(21).
    03 ICH-TEAM    PIC X(02).
    03             PIC X(04).
    03 ICH-BANGO   PIC X(02).
    03             PIC X(05).
    03 ICH-NAME    PIC X(15).
    03             PIC X(04).
    03 ICH-DASU    PIC ZZ9.
    03             PIC X(04).
    03 ICH-HIT     PIC ZZ9.
    03             PIC X(04).
    03 ICH-HR      PIC ZZ9.
    03             PIC X(04).
    03 ICH-DATEN   PIC ZZ9.
    03             PIC X(04).
    03 ICH-AVG     PIC Z.999.
```

（続く）

```
PROCEDURE DIVISION.
BEGIN SECTION.
    PERFORM INIT-RTN
    PERFORM MAIN-RTN  UNTIL SEI-REC = HIGH-VALUE
    PERFORM TERM-RTN
    STOP RUN.

INIT-RTN SECTION.
INIT1.
    OPEN INPUT  SEI-FILE
         OUTPUT ICH-FILE
    READ SEI-FILE
        AT END
           MOVE HIGH-VALUE TO SEI-REC
        NOT AT END
           MOVE SPACE TO ICHIRAN
           WRITE ICH-REC FROM ICHIRAN   AFTER PAGE
           WRITE ICH-REC FROM MIDASHI-1 AFTER 3
           WRITE ICH-REC FROM MIDASHI-2 AFTER 2
           WRITE ICH-REC FROM ICHIRAN   AFTER 1
    END-READ.

MAIN-RTN SECTION.
MAIN1.
    MOVE SEI-TEAM  TO ICH-TEAM
    MOVE SEI-BANGO TO ICH-BANGO
    MOVE SEI-NAME  TO ICH-NAME
    MOVE SEI-DASU  TO ICH-DASU
    MOVE SEI-HIT   TO ICH-HIT
    MOVE SEI-HR    TO ICH-HR
    MOVE SEI-DATEN TO ICH-DATEN
    COMPUTE ICH-AVG ROUNDED = SEI-HIT / SEI-DASU
        ON SIZE ERROR
           MOVE ZERO TO ICH-AVG
    END-COMPUTE
    WRITE ICH-REC FROM ICHIRAN AFTER 1
    READ SEI-FILE
        AT END
           MOVE HIGH-VALUE TO SEI-REC
    END-READ.

TERM-RTN SECTION.
TERM1.
    CLOSE SEI-FILE
          ICH-FILE.
```

９．６．２　ＣＯＢＯＬ整列処理

学生マスターファイルを入力して、クラス別の成績順位表を作成する。

①学生マスターファイルの形式

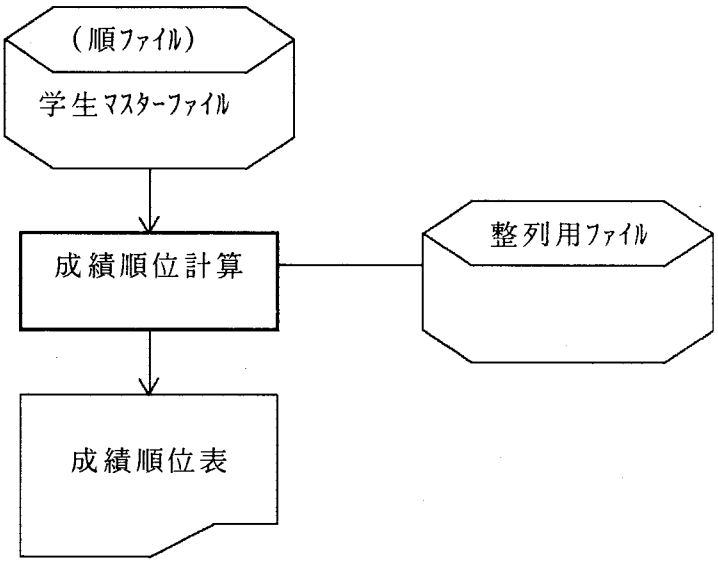
項目	学生番号			氏　名（カナ）	氏　名（漢字）	性別	成　績					趣味特技			空白
	クラス		番号				国語	数学	社会	理科	英語	1	2	3	
	年度	区分													
桁	3	1	2	１２	１８	1	3	3	3	3	3	1	2	3	9
	9	X	9	X	X	X	9	9	9	9	9	X	X	X	X

②処理内容

(1) 1ﾚｺｰﾄﾞから１行の明細を作成する

(2)ｸﾗｽ別（学生番号の前4桁）に、
5科目の合計点の高い順に印刷する

(3) 1ﾍﾟｰｼﾞには20名分を印刷する



クラス別成績順位表の形式

```
                **  成績リスト  **  DATE  99/99/99   PAGE  ZZ9

                                          作成者  XXXXXXXXXX

学生番号  氏　　名          英語  数学  国語  理科  社会  合計

XXXXXX    XXXXXXXXXXXX      ZZ9   ZZ9   ZZ9   ZZ9   ZZ9   ZZ9

XXXXXX    XXXXXXXXXXXX      ZZ9   ZZ9   ZZ9   ZZ9   ZZ9   ZZ9

                （１ページ２０人分）

XXXXXX    XXXXXXXXXXXX      ZZ9   ZZ9   ZZ9   ZZ9   ZZ9   ZZ9
```

「ＣＯＢＯＬ整列処理」のプログラム例

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 整列処理.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT MST-FILE ASSIGN TO 作成者語.
    SELECT SRT-FILE ASSIGN TO 作成者語.
    SELECT CLS-FILE ASSIGN TO 作成者語.
DATA DIVISION.
FILE SECTION.
FD  MST-FILE
    BLOCK CONTAINS 10 RECORDS.
01  MST-REC.
    03 MST-GAKBAN.
      05 MST-CLS.
         07 MST-NEN    PIC 9(03).
         07 MST-KBN    PIC X(01).
      05 MST-BAN       PIC 9(02).
    03 MST-NAME        PIC X(12).
    03 MST-KANJI       PIC X(18).
    03 MST-SEX         PIC X(01).
    03 MST-SEISEKI     OCCURS 5 TIMES.
       05 MST-TEN      PIC 9(03).
    03 MST-SHUMI       OCCURS 3 TIMES.
       05 MST-TOK      PIC X(01).
    03                 PIC X(09).
SD  SRT-FILE.
01  SRT-REC.
    03 SRT-GAKBAN.
      05 SRT-CLS.
         07 SRT-NEN    PIC 9(03).
         07 SRT-KBN    PIC X(01).
      05 SRT-BAN       PIC 9(02).
    03 SRT-NAME        PIC X(12).
    03 SRT-SEISEKI     OCCURS 5 TIMES.
       05 SRT-TEN      PIC 9(03).
    03 SRT-GOKEI       PIC 9(03).
FD  CLS-FILE
    LABEL RECORD OMITTED.
01 CLS-REC.
    03                 PIC X(132).

WORKING-STORAGE SECTION.
77  GOKEI              PIC 9(03) VALUE ZERO.
77  I                  PIC 9(02) VALUE ZERO USAGE BINARY.
77  HANTEI             PIC 9(01) VALUE ZERO.
77  CNT-LINE           PIC 9(02) VALUE 20.
77  CNT-PAGE           PIC 9(03) VALUE ZERO.
```

（続く）

（続き）

```
01  OOMIDASI.
    03                PIC X(16) VALUE SPACE.
    03                PIC X(03) VALUE "**".
    03                PIC X(16) VALUE "成績リスト".
    03                PIC X(06) VALUE "**".
    03                PIC X(05) VALUE "DATE".
    03 OOM-DATE       PIC 99/99/99.
    03                PIC X(08) VALUE "   PAGE".
    03 OOM-PAGE       PIC ZZ9.
01  CHUMIDASI.
    03                PIC X(53) VALUE SPACE.
    03                PIC X(14) VALUE "作成者".
    03                PIC X(14) VALUE "  橋本  太郎  ".
01  KOMIDASI.
    03                PIC X(11) VALUE SPACE.
    03                PIC X(14) VALUE "学生番号".
    03                PIC X(21) VALUE "氏    名".
    03                PIC X(10) VALUE "英語".
    03                PIC X(10) VALUE "数学".
    03                PIC X(10) VALUE "国語".
    03                PIC X(10) VALUE "理科".
    03                PIC X(10) VALUE "社会".
    03                PIC X(10) VALUE "合計".
01  MEISAI.
    03                PIC X(11) VALUE SPACE.
    03 MEI-GAKBAN     PIC X(06).
    03                PIC X(04) VALUE SPACE.
    03 MEI-NAME       PIC X(12).
    03                PIC X(06) VALUE SPACE.
    03 MEI-EIGO       PIC ZZ9.
    03                PIC X(03) VALUE SPACE.
    03 MEI-SUGAKU     PIC ZZ9.
    03                PIC X(03) VALUE SPACE.
    03 MEI-KOKUGO     PIC ZZ9.
    03                PIC X(03) VALUE SPACE.
    03 MEI-RIKA       PIC ZZ9.
    03                PIC X(03) VALUE SPACE.
    03 MEI-SHAKAI     PIC ZZ9.
    03                PIC X(03) VALUE SPACE.
    03 MEI-GOKEI      PIC ZZ9.
```

（続く）

```
PROCEDURE DIVISION.
BEGIN SECTION.
BEGIN1.
    ACCEPT OOM-DATE FROM DATE
    SORT SRT-FILE ON ASCENDING  KEY SRT-CLS
                     DESCENDING KEY SRT-GOKEI
                  INPUT  PROCEDURE INPUT-RTN
                  OUTPUT PROCEDURE OUTPUT-RTN
    STOP RUN.

INPUT-RTN SECTION.
INPUT1.
    OPEN INPUT MST-FILE
    READ MST-FILE
         AT END MOVE 9 TO HANTEI
    END-READ
    PERFORM UNTIL HANTEI = 9
        MOVE ZERO TO GOKEI
        PERFORM VARYING I FROM 1 BY 1 UNTIL I > 5
            COMPUTE GOKEI = GOKEI + MST-TEN(I)
            MOVE MST-TEN(I) TO SRT-TEN(I)
        END-PERFORM
        MOVE GOKEI      TO SRT-GOKEI
        MOVE MST-GAKBAN TO SRT-GAKBAN
        MOVE MST-NAME   TO SRT-NAME
        RELEASE SRT-REC
        READ MST-FILE
             AT END MOVE 9 TO HANTEI
        END-READ
    END-PERFORM
    CLOSE MST-FILE.
```

（続く）

```
OUTPUT-RTN SECTION.
OUTPUT1.
    OPEN OUTPUT CLS-FILE
    MOVE ZERO TO HANTEI
    RETURN SRT-FILE
        AT END MOVE 9 TO HANTEI
    END-RETURN
    PERFORM UNTIL HANTEI = 9
        MOVE SRT-TEN(1) TO MEI-EIGO
        MOVE SRT-TEN(2) TO MEI-SUGAKU
        MOVE SRT-TEN(3) TO MEI-KOKUGO
        MOVE SRT-TEN(4) TO MEI-RIKA
        MOVE SRT-TEN(5) TO MEI-SHAKAI
        MOVE SRT-GOKEI  TO MEI-GOKEI
        MOVE SRT-GAKBAN TO MEI-GAKBAN
        IF CNT-LINE = 20
            MOVE SPACE TO CLS-REC
            WRITE CLS-REC AFTER PAGE
            COMPUTE CNT-PAGE = CNT-PAGE + 1
            MOVE CNT-PAGE TO OOM-PAGE
            WRITE CLS-REC FROM OOMIDASI  AFTER 3
            WRITE CLS-REC FROM CHUMIDASI AFTER 2
            WRITE CLS-REC FROM KOMIDASI  AFTER 2
            MOVE SPACE TO CLS-REC
            WRITE CLS-REC AFTER 1
            MOVE ZERO TO CNT-LINE
        END-IF
        WRITE CLS-REC FROM MEISAI AFTER 2
        COMPUTE CNT-LINE = CNT-LINE + 1
        RETURN SRT-FILE
            AT END MOVE 9 TO HANTEI
        END-RETURN
    END-PERFORM
    CLOSE CLS-FILE.
```

９．６．３　金種計算

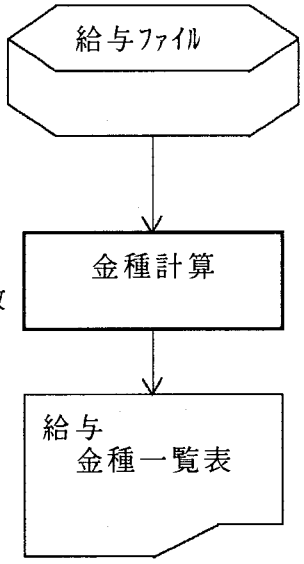
給与ファイルを入力して、個人の差引支給額を計算して、お金の種類と枚数を求めて給与金種一覧表を作成する。

①給与ファイルの形式

項目	部課ｺｰﾄﾞ	社員ｺｰﾄﾞ	氏　名	基本給	扶養手当	住宅手当	通勤手当	所得税	住民税	社会保険	雇用保険
桁	2	6	２０	6	5	5	5	5	5	5	5
属性	9	9	X	9	9	9	9	9	9	9	9

②処理内容

(1)差引支給額の計算
　差引支給額＝総支給額－総控除額
　総支給額＝基本給＋扶養手当＋住宅手当
　　＋通勤手当
　総控除額＝所得税＋住民税＋社会保険
　　＋雇用保険
(2)給与金種一覧表には個人ごとに次の項目を印刷する。
　部課ｺｰﾄﾞ、社員ｺｰﾄﾞ、差引支給額、金種別枚数
(3)部課ごとの合計の金種を印刷する。
(4)最後に総合計の金種を印刷する。
(5)改ﾍﾟｰｼﾞは考慮しない。



③給与金種一覧表の形式

```
                    ***** ｷｭｳﾖ　ｷﾝｼｭ　ｲﾁﾗﾝﾋｮｳ　*****      DATE  99/99/99

ﾌﾞｶ  ｼｬｲﾝｺｰﾄﾞ    ｷ ﾝ ｶﾞ ｸ  ¥10,000  ¥5,000 ¥1,000  ¥500  ¥100  ¥50  ¥10  ¥1

 99   999999   Z,ZZZ,ZZ9      ZZ9       9      9      9     9     9    9   9
 99   999999   Z,ZZZ,ZZ9      ZZ9       9      9      9     9     9    9   9
        :
 99   999999   Z,ZZZ,ZZ9      ZZ9       9      9      9     9     9    9   9

      BU-KEI  ZZZ,ZZZ,ZZ9    Z,ZZ9     ZZ9    ZZ9    ZZ9   ZZ9   ZZ9  ZZ9 ZZ9

 99   999999   Z,ZZZ,ZZ9      ZZ9       9      9      9     9     9    9   9
 99   999999   Z,ZZZ,ZZ9      ZZ9       9      9      9     9     9    9   9

      BU-KEI  ZZZ,ZZZ,ZZ9    Z,ZZ9     ZZ9    ZZ9    ZZ9   ZZ9   ZZ9  ZZ9 ZZ9

      SO-KEI  ZZZ,ZZZ,ZZ9    Z,ZZ9     ZZ9    ZZ9    ZZ9   ZZ9   ZZ9  ZZ9 ZZ9
```

「金種計算」のプログラム例

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 金種計算.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT KYU-FILE ASSIGN TO 作成者語.
    SELECT LST-FILE ASSIGN TO 作成者語.
DATA DIVISION.
FILE SECTION.
FD  KYU-FILE
    BLOCK CONTAINS 10 RECORDS.
01  KYU-REC.
    03  KYU-BUKA       PIC X(02).
    03  KYU-SYAIN      PIC X(06).
    03  KYU-NAME       PIC X(20).
    03  KYU-KIHON      PIC 9(06).
    03  KYU-FUYO       PIC 9(05).
    03  KYU-JYUTAKU    PIC 9(05).
    03  KYU-TUKIN      PIC 9(05).
    03  KYU-STAX       PIC 9(05).
    03  KYU-JTAX       PIC 9(05).
    03  KYU-SHOKEN     PIC 9(05).
    03  KYU-KHOKEN     PIC 9(05).
FD  LST-FILE
    BLOCK CONTAINS 10 RECORDS.
01  LST-REC.
    03                 PIC X(132).

WORKING-STORAGE SECTION.
01  OOMIDASI.
    03                 PIC X(41) VALUE SPACE.
    03                 PIC X(08) VALUE "*****".
    03                 PIC X(11) VALUE "キ ュ ウ ヨ".
    03                 PIC X(11) VALUE "キ ン シ ュ".
    03                 PIC X(16) VALUE "ｲﾁ ラ ン ヒ ョ ウ".
    03                 PIC X(27) VALUE "*****".
    03 HIZUKE          PIC 99/99/99.
01  KOMIDASI.
    03                 PIC X(05) VALUE SPACE.
    03                 PIC X(09) VALUE "ﾌﾞｶ".
    03                 PIC X(14) VALUE "ｼｬｲﾝｺｰﾄﾞ".
    03                 PIC X(12) VALUE "キ ン ｶﾞ ク".
    03                 PIC X(11) VALUE "¥10,000".
    03                 PIC X(10) VALUE "¥5,000".
    03                 PIC X(12) VALUE "¥1,000".
    03                 PIC X(10) VALUE "¥500".
    03                 PIC X(11) VALUE "¥100".
    03                 PIC X(10) VALUE "¥50".
    03                 PIC X(11) VALUE "¥10".
    03                 PIC X(10) VALUE "¥5".
    03                 PIC X(02) VALUE "¥1".
```

（続く）

```
01  MEISAI.
    03                  PIC X(06) VALUE SPACE.
    03 MEI-BUKA         PIC 9(02).
    03                  PIC X(07) VALUE SPACE.
    03 MEI-SYAIN        PIC 9(02).
    03                  PIC X(06) VALUE SPACE.
    03 MEI-KINGAKU      PIC Z,ZZZ,ZZ9.
    03                  PIC X(01) VALUE SPACE.
    03 MEI-MAISU        OCCURS 9 TIMES.
       05               PIC X(07) VALUE SPACE.
       05 MEI-MAI        PIC ZZ9.
01  KEI.
    03                  PIC X(15) VALUE SPACE.
    03 KEI-SYAIN        PIC X(08).
    03 KEI-KINGAKU      PIC Z,ZZZ,ZZZ,ZZ9.
    03                  PIC X(05) VALUE SPACE.
    03 KEI-MAISU        OCCURS 9 TIMES.
       05 KEI-MAI        PIC ZZ,ZZ9.
       05               PIC X(04).
01  BUKEI.
    03 BKW              PIC 9(09) VALUE ZERO.
    03 BKM              OCCURS 9 TIMES.
       05 BKMW           PIC 9(04) VALUE ZERO.
01  GOKEI.
    03 GKW              PIC 9(09) VALUE ZERO.
    03 GKM              OCCURS 9 TIMES.
       05 GKMW           PIC 9(04) VALUE ZERO.
01  KEISAN.
    03 KST              PIC 9(05) OCCURS 9 TIMES.
01  OLD-BUKA           PIC X(02).
01  WRK-MAI            PIC 9(03).
01  KOJYO              PIC 9(06).
01  HANTEI             PIC 9(01) VALUE ZERO.
01  I                  PIC 9(02) USAGE BINARY.
```

（続く）

次頁の代入文をデータ部で前もって設定する場合の例。

```
01  KEISAN-VALUE.
    03                  PIC 9(05) VALUE 10000.
    03                  PIC 9(05) VALUE  5000.
    03                  PIC 9(05) VALUE  1000.
    03                  PIC 9(05) VALUE   500.
    03                  PIC 9(05) VALUE   100.
    03                  PIC 9(05) VALUE    50.
    03                  PIC 9(05) VALUE    10.
    03                  PIC 9(05) VALUE     5.
    03                  PIC 9(05) VALUE     1.
01  KEISAN REDEFINES KEISAN-VALUE.
    03  KST            PIC 9(05) OCCURS 9 TIMES.
```

```
PROCEDURE DIVISION.
BEGIN SECTION.
BEGIN1.
    PERFORM INIT-RTN
    PERFORM MAIN-RTN
    PERFORM TERM-RTN
    STOP RUN.

INIT-RTN SECTION.
INIT1.
    OPEN INPUT  KYU-FILE
         OUTPUT LST-FILE
    ACCEPT HIZUKE FROM DATE
    MOVE 10000 TO KST(1)
    MOVE  5000 TO KST(2)
    MOVE  1000 TO KST(3)
    MOVE   500 TO KST(4)
    MOVE   100 TO KST(5)
    MOVE    50 TO KST(6)
    MOVE    10 TO KST(7)
    MOVE     5 TO KST(8)
    MOVE     1 TO KST(9)
    PERFORM READ-RTN.

MAIN-RTN SECTION.
MAIN1.
    MOVE SPACE TO LST-REC
    WRITE LST-REC AFTER PAGE
    WRITE LST-REC FROM OOMIDASI AFTER 3
    WRITE LST-REC FROM KOMIDASI AFTER 2
    PERFORM UNTIL KYU-BUKA = HIGH-VALUE
        MOVE KYU-BUKA TO OLD-BUKA
        PERFORM UNTIL KYU-BUKA NOT = OLD-BUKA
            PERFORM SUB1-RTN
        END-PERFORM
        PERFORM SUB1-RTN
    END-PERFORM.

READ-RTN SECTION.
READ1.
    READ KYU-FILE
        AT END MOVE HIGH-VALUE TO KYU-BUKA
    END-READ.
```

左の代入文を計算で行う場合

```
MOVE 10000 TO 分子
MOVE     2 TO 分母
PERFORM VARYING K FROM 1 BY 1
                    UNTIL K > 9
    MOVE 分子 TO KST(K)
    COMPUTE 分子 = 分子 / 分母
    COMPUTE 分母 =  7 - 分母
END-PERFORM
```

（続く）

```
SUB1-RTN SECTION.
SUB11.
    COMPUTE KOJYO = KYU-KIHON  + KYU-FUYO   + KYU-JYUTAKU
                  + KYU-TUKIN  - KYU-STAX   - KYU-JTAX
                  - KYU-SHOKEN - KYU-KHOKEN
    COMPUTE BKW   = BKW + KOJYO
    MOVE KOJYO TO MEI-KINGAKU
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 9 OR KOJYO = 0
        COMPUTE WRK-MAI = KOJYO / KST(I)
        COMPUTE KOJYO  = KOJYO - WRK-MAI * KST(I)
        MOVE WRK-MAI TO MEI-MAI
        COMPUTE BKMW(I) = BKMW(I) + WRK-MAI
    END-PERFORM
    MOVE KYU-BUKA TO MEI-BUKA
    MOVE KYU-SYAIN TO MEI-SYAIN
    WRITE LST-REC FROM MEISAI AFTER 2
    PERFORM VARYING I FROM I BY I UNTIL I > 9
        MOVE ZERO TO MEI-MAI(I)
    END-PERFORM
    PERFORM READ-RTN.

SUB2-RTN SECTION.
SUB21.
    MOVE "BU-KEI" TO KEI-SYAIN
    MOVE BKW      TO KEI-KINGAKU
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 9
        MOVE BKMW(I) TO KEI-MAI(I)
        COMPUTE GKMW(I) = GKMW(I) + BKMW(I)
    END-PERFORM
    COMPUTE GKW = GKW + BKW
    WRITE LST-REC FROM KEI AFTER 2
    INITIALIZE BUKEI.

TERM-RTN SECTION.
TERM1.
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 9
        MOVE GKMW(I) TO KEI-MAI(I)
    END-PERFORM
    MOVE "SO-KEI" TO KEI-SYAIN
    MOVE GKW      TO KEI-KINGAKU
    WRITE LST-REC FROM KEI AFTER 2
    CLOSE KYU-FILE
          LST-FILE.
```

指導上の留意点

◎字下げ（indentation）
プログラムの論理構造の範囲を明確にするために、制御構造の「選択」、「繰り返し」は、「順次」と区別できるように字下げを行うとよい。
また、制御が入れ子（ネスト）になる場合にも字下げを行う。なお、深い入れ子になる場合は、適当なレベルで止めてそれ以降の入れ子はサブルーチンにして外にだす。

◎注釈（comment）
データ名、手続き名など利用者が任意につけられるが、各種の付け方があり他人に分かり難いこともある。言語の注釈機能を利用して、データの意味や処理の概略などの説明を原始プログラム中に記述するように心がけるとよい。

◎データ名
ＣＯＢＯＬでは利用者名に３０文字までの名前が付けられるので、データ名や手続き名にその意味や機能等を表現するような名前にするとよい。
また、事務処理では複数のファイルを使う場合があり、どのファイルのデータ名なのか判別できる名前の付け方（接頭部、接尾部）もある。
大規模システムの開発では、ファイルのレコード項目に共通のデータ名を付け、そのファイルを使用するプログラムはＣＯＰＹ文やＲＥＰＬＡＣＥ文を使用して原始プログラムの中へ複写することでデータ名の統一を計っている。

ＦＯＲＴＲＡＮでは変数名が６文字までなので、変数名を省略することが多い。省略の変数名についても６文字で意味がわかるようにする。
なお、上位１桁でデータの属性（integer、real）も決まるためその注意も必要である。

◎モジュール化
処理単位を分割する場合の目安として、コンパイラのコンパイルリストの１頁（約60行）に納まるようにモジュール化するとよい。
各モジュールは「SECTION」毎にまとめるとPERFORMの実行範囲も明確になり、段落名の追加もPERFORMを意識しないで行える。

◎PICTURE句の揃え方
ＣＯＢＯＬではデータ名とその属性（タイプ、桁数等）を記述するがPICTURE句を前後の行で揃えると見やすさが向上する。
また、桁数についても前ゼロをつけて統一しておくと集団項目のサイズの計算などが楽になる。
編集項目は、繰り返しの括弧を使わないようにすると桁数や形式が理解しやすくなる。

◎ＧＯＴＯ文
構造化プログラミングの手法に従ってコーディングすれば、全くGOTO文を使用しないプログラムが作成できる。

◎終止符（ピリオド）
ＩＦ文など処理範囲を区切るためにピリオドの使用はできる限りやめて明示範囲符を使う。字下げと条件文の終了をピリオドで記述すると誤りが発生しやすい。

```
IF  条件
    MOVE  ﾃﾞｰﾀ名1  TO  ﾃﾞｰﾀ名2  .   ←ピリオドでＩＦの終了になるので
    COMPUTE  ﾃﾞｰﾀ名3  =  算術文        条件に関係なくCOMPUTEが実行される。
MOVE  定数  TO  ﾃﾞｰﾀ名4
```

◎ＰＥＲＦＯＲＭ文
「繰り返し」でＰＥＲＦＯＲＭ文を使用する場合は、明示的に「TEST　AFTER」、「TEST　BEFORE」を記述するとよい。

◎ファイルの定義
ＣＯＢＯＬでは、外部の物理ファイルの属性をＣＯＢＯＬの文法上で表現できないことがある。これらの物理ファイルの定義はコンパイラ作成者によって文法が拡張されている。
主に、ＳＥＬＥＣＴとＦＤ句が拡張されている。

```
SELECT  ﾌｧｲﾙ名  ASSIGN  TO  利用者語

FD  ﾌｧｲﾙ名

        ﾌｧｲﾙ情報の拡張記述

01  ﾚｺｰﾄﾞ名  ～
```

◎改ページ、改行
ＣＯＢＯＬのＷＲＩＴＥ文で改ページ（PAGE指定）や改行（LINES指定）の場合のレコード構成は、次のように処理するコンパイラもある。

```
01  ﾚｺｰﾄﾞ名.
    03  FILLER     PICTURE  X(1).
    03  ﾃﾞｰﾀ名1  ～
    03  ﾃﾞｰﾀ名2  ～
```

ﾚｺｰﾄﾞの最初のﾊﾞｲﾄが改行、改頁の制御ﾊﾞｲﾄに変換される。

```
WRITE  ﾚｺｰﾄﾞ名  AFTER  ADVANCING  PAGE
                                   (n  LINES)
```

これらは、ＦＯＲＴＲＡＮの改ページ、改行でも意識して記述することもある。

```
      WRITE(6,100)  変数名1  変数名2
100   FORMAT(1H1,～)
```

◎ファイル入力
ファイルの入力処理で見落とすことの中に、レコードが存在しない場合や１件のレコードだけのファイルの場合がある。
キーのマッチング処理ではファイルの終了時のキーの値をどうするのかも注意する必要がある。一般にはキーの値として「HIGH-VALUE」がよく使われるがキー項目が「数字項目」の場合には注意を要する。

◎配列の処理
汎用コンピュータのＣＯＢＯＬでは、配列の範囲外の添字を指定しても実行時エラーにならない場合もある。これは、プログラムの実行性能のために添字の範囲をﾁｪｯｸする処理を省略したためである。
添字には、バイナリ項目（USAGE BINARY）や指標名を使うと実行性能が向上する。