第4章
条件による判定

1 IF文

例題4　人口密度の最大値

県ファイルを読んで，都道府県ごとの人口密度を求め，人口密度の最大値と最大の都道府県名を求める一覧表を作成しなさい。

▼入力形式

県コード	県名	面積	人口
9 (02)	X (08)	9 (05)	9 (05)

▼処理条件

人口密度の小数点以下は四捨五入する。

▼出力形式

```
                *** トドウフケン ヘイキン イチランヒョウ ***
コードバン   ケ ン メ イ          メンセキ        ジンコウ        ミツドバン
   01        ホツカイドウ        83,520          5,646             68
   02        アオモリ             9,619          1,529            159
   03        イワテ              15,277          1,433             94
   04        ミヤギ               7,292          2,210            303
   05        アキタ              11,613          1,245            107
    ｜         ｜                  ｜              ｜               ｜
   46        カゴシマ             9,167          1,812            198
   47        オキナワ             2,255          1,230            545

                                                              5,381
                                                              トウキヨウ
             サイタマイ
             ケンメイ
```

問題の分析と考え方

最大値は次の手順で求める。

たとえば，データが右のような3桁の数であったとする。

① 最大値に初期値として，データとしてはありえない小さい値をセットする。

②から④までをデータ終了まで繰り返す。

② データを読む。

③ 最大値と「数」を比較する。

数＞最大値

であれば，④にいく。

④ 「数」を最大値として記憶する。

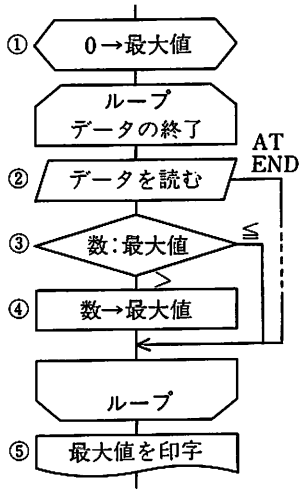
⑤ 最大値を印字する。

COBOLでは，上記の手順③のように，大小関係を判断するさいにはIF文を用いる。

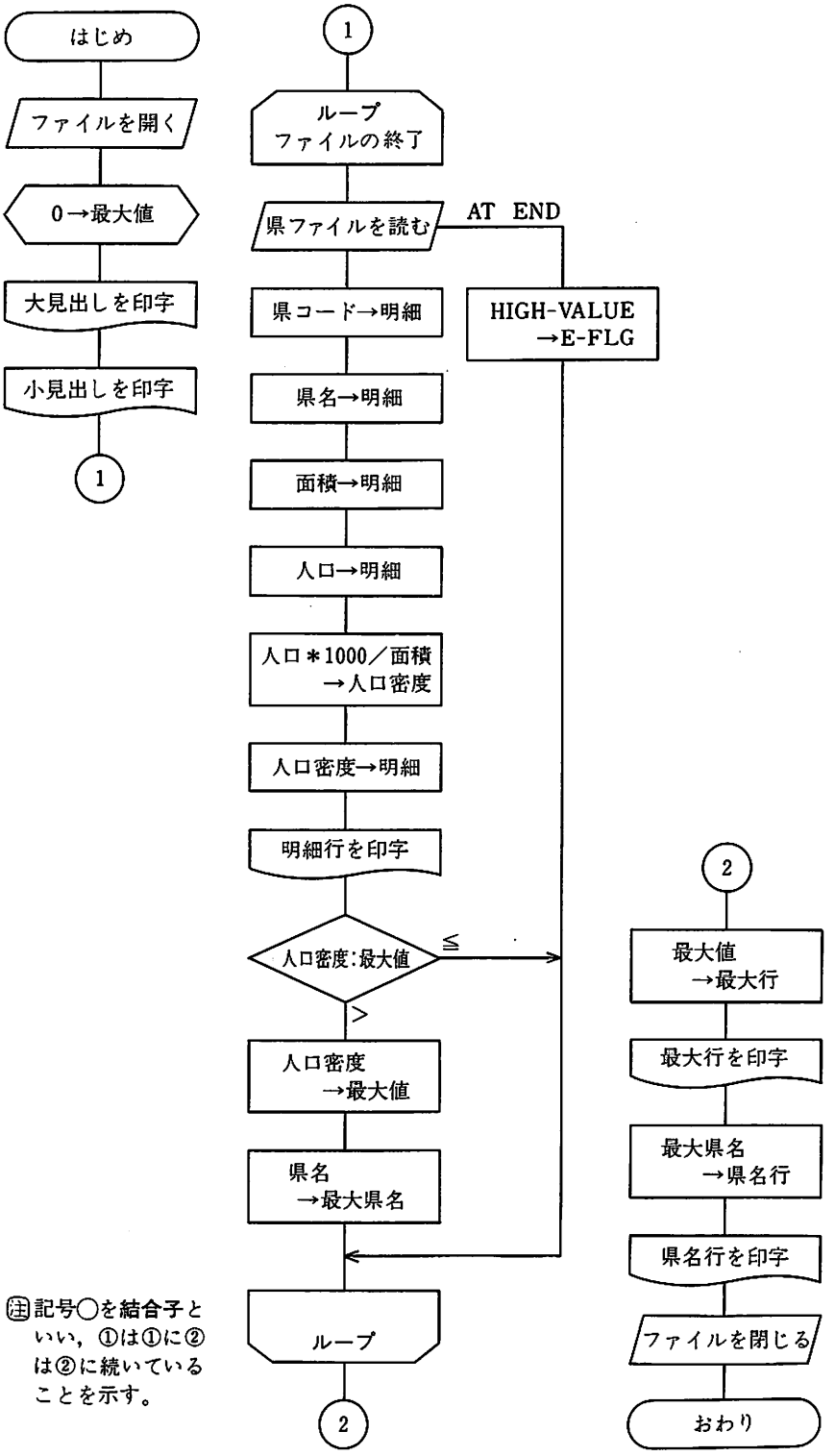
入力データ

数
68
159
94

	数	最大値	
①		0	⇐0をセット
②	68	0	
③	68	0	⇐68＞0
④	68	68	
②	159	68	
③	159	68	⇐159＞68
④	159	159	
②	94	159	
③	94	159	⇐94≦159
⑤	94	159	⇨159を出力



●流れ図●



㊟記号○を結合子といい，①は①に②は②に続いていることを示す。

◉プログラム◉

```
DATA                    DIVISION.
FILE                    SECTION.
FD   KEN-FILE.
01   KEN-REC.
     05 KENCODE         PIC 9(02).
     05 KENMEI          PIC X(08).
     05 MENSEKI         PIC 9(05).
     05 JINKO           PIC 9(05).
FD   ITIRAN-FILE.
01   ITIRAN-REC         PIC X(132).
WORKING-STORAGE SECTION.
01   E-FLG              PIC X(01)  VALUE  LOW-VALUE.
01   MITUDO             PIC 9(05).
01   SAIDAI             PIC 9(05).
01   MAXKEN             PIC X(08).
01   OOMIDASI-GYO       PIC X(47)  VALUE
     "                       *** ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ ***".
01   KOMIDASI-GYO       PIC X(67)  VALUE
     "        ｺｰﾄﾞ    ｹ ﾝ ﾒ ｲ          ﾒﾝｾｷ        ｼﾞﾝｺｳ           ﾐｯﾄﾞ".
01   MEISAI-GYO.
     05                  PIC X(09)  VALUE  SPACE.
     05 M-KENCODE       PIC 9(02).
     05                  PIC X(05)  VALUE  SPACE.
     05 M-KENMEI        PIC X(08).
     05                  PIC X(05)  VALUE  SPACE.
     05 M-MENSEKI       PIC ZZ,ZZ9.
     05                  PIC X(05)  VALUE  SPACE.
     05 M-JINKO         PIC ZZ,ZZ9.
     05                  PIC X(05)  VALUE  SPACE.
     05 M-MITUDO        PIC ZZ,ZZ9.
01   MAX-GYO.
     05                  PIC X(16)  VALUE  SPACE.
     05                  PIC X(05)  VALUE  "ｻｲﾀﾞｲ".
     05                  PIC X(30)  VALUE  SPACE.
     05 MAX-MITUDO      PIC ZZ,ZZ9.
01   NAME-GYO.
     05                  PIC X(16)  VALUE  SPACE.
     05                  PIC X(04)  VALUE  "ｹﾝﾒｲ".
     05                  PIC X(31)  VALUE  SPACE.
     05 NAME-MAXKEN      PIC X(08).
*
 PROCEDURE DIVISION.
 SYORI.
     OPEN      INPUT KEN-FILE     OUTPUT ITIRAN-FILE
     MOVE      0             TO   SAIDAI
     WRITE     ITIRAN-REC  FROM   OOMIDASI-GYO  AFTER  PAGE
     WRITE     ITIRAN-REC  FROM   KOMIDASI-GYO  AFTER  1
     PERFORM   UNTIL  E-FLG  =    HIGH-VALUE
       READ  KEN-FILE
         AT END
           MOVE  HIGH-VALUE TO  E-FLG
         NOT AT END
           MOVE  KENCODE     TO  M-KENCODE
           MOVE  KENMEI      TO  M-KENMEI
```

```
        MOVE   MENSEKI      TO   M-MENSEKI
        MOVE   JINKO        TO   M-JINKO
        COMPUTE MITUDO      ROUNDED  =
                            JINKO  * 1000 / MENSEKI
        MOVE   MITUDO       TO   M-MITUDO
        WRITE  ITIRAN-REC   FROM  MEISAI-GYO  AFTER 1
        IF     MITUDO  >    SAIDAI
          THEN
            MOVE   MITUDO   TO   SAIDAI
            MOVE   KENMEI   TO   MAXKEN
          ELSE
            CONTINUE
        END-IF
      END-READ
    END-PERFORM
    MOVE    SAIDAI        TO     MAX-MITUDO
    WRITE   ITIRAN-REC    FROM   MAX-GYO   AFTER 2
    MOVE    MAXKEN        TO     NAME-MAXKEN
    WRITE   ITIRAN-REC    FROM   NAME-GYO AFTER 1
    CLOSE   KEN-FILE      ITIRAN-FILE
    STOP    RUN.
```

プログラムの解説

四捨五入

ROUNDED 句

```
COMPUTE   データ名   ROUNDED  =  算術式
```

COMPUTE 文において，ROUNDED 句を指定することによって，受取り側のデータ名の内容が自動的に四捨五入される。

例 A＝5，B＝2 のとき

```
COMPUTE C ROUNDED = A / B
```

この COMPUTE 文によって，C には 3 が記憶される。

IF 文

IF文

```
IF    条件    THEN    文1
                      .....
              ELSE    文2
                      .....
END-IF
```

条件を満足した場合には，文 1 以下を実行し，条件を満足しない場合は文 2 以下を実行する。

文 2 以下がない場合は，ELSE 以下は省略できる。

文 2 以下が存在しない場合も次に示す CONTINUE 文を用いるとわかりやすいプログラムになる。

IF の条件は，次のいずれかの形で書かれる。

{ データ名 1 / 定数 1 / 算術式 1 } 比較演算子 { データ名 2 / 定数 2 / 算術式 2 }

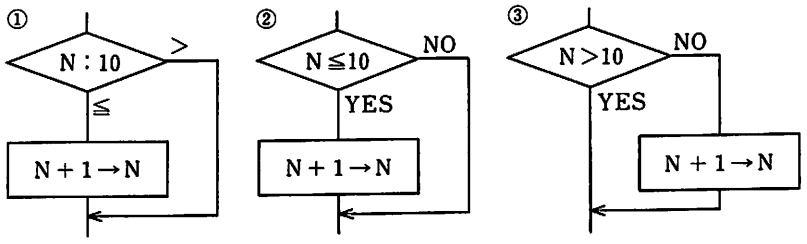
大小判定をする条件を書くために用いられる文字を**比較演算子**といい，右のものがある。

比較演算子	意　味
=	等しい (=)
NOT =	等しくない (≠)
>	大きい (>)
<	小さい (<)
>= または NOT <	大きいか等しい (≧)
<= または NOT >	小さいか等しい (≦)

▲比較演算子

IF 文は，流れ図記号では**判断**を用いる。

例 次の 3 つの流れ図は同じ処理を表している。



CONTINUE 文

CONTINUE 文

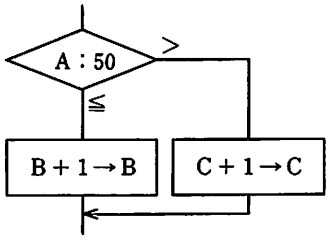
```
CONTINUE
```

プログラムの実行には何も影響を与えない文。手続き部のどこにでも書くことができ，プログラムを見やすくするために用いる。

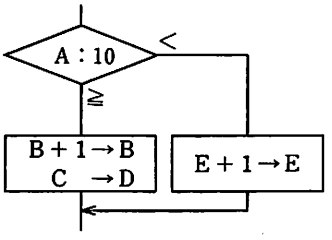
いろいろな IF 文の例

IF 文を用いた例を，流れ図とともに示す。書き方はいろいろ許されるが，誰が見てもわかりやすい書き方をすることが大切である。

```
[1] IF  A  <=  50
        THEN
          COMPUTE  B  =  B  +  1
        ELSE
          COMPUTE  C  =  C  +  1
    END-IF
```

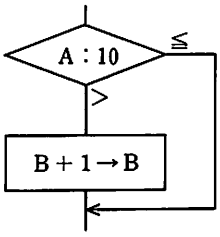



```
[2] IF  A  >=  10
        THEN
          COMPUTE  B  =  B  +  1
          MOVE  C  TO  D
        ELSE
          COMPUTE  E  =  E  +  1
    END-IF
```



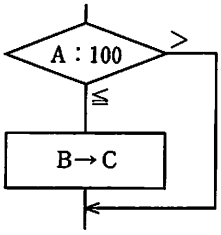
[3] 条件を満足しない場合の処理がないときは，ELSE以降が省略できるが，処理がないということを明確にするため，CONTINUE文を使ってもよい。

```
IF A > 10                    IF A > 10
   THEN                         THEN
     COMPUTE B = B + 1            COMPUTE B = B + 1
   ELSE                       END-IF
     CONTINUE
END-IF
```



[4] 「NOT ＞」「NOT ＜」を使うことができるが，「＜＝」「＞＝」のほうがわかりやすい。

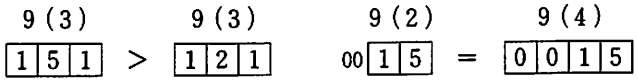
```
IF A <= 100                  IF A NOT > 100
   THEN                         THEN
     MOVE B TO C                  MOVE B TO C
   ELSE                         ELSE
     CONTINUE                     CONTINUE
END-IF                       END-IF
```



数字比較と文字比較

条件式の右辺と左辺は同じ性質どうしの比較が原則である。数字項目と英数字項目，数字項目と数字編集された項目という間での比較はできない。

数字項目どうしのときは，数字の大小関係で，条件の満足不満足が決まる。このことを**数字比較**という。桁数が等しくない場合は，小さい桁数のデータの左側に0があるものとして比較される。



▲数字比較

英数字項目どうしの場合は，**文字比較**といい，次の規則で大小関係が決まる。

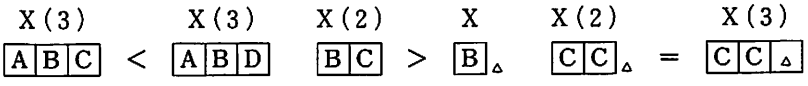
桁数が同じ場合は，左端の文字から順番に比較していき，等しくない文字が現れたときに大小関係が決まる。文字の大小関係は，その文字のビット構成によって，次のように大小関係が決まっている。*

空白＜A＜B＜……＜Z＜0＜1＜……＜9

▲文字の大小関係

*機種により，数字より英字のほうが大きい場合もある。

桁数が異なる場合は，小さい桁数のデータの右側に空白があるものとして，文字比較がなされる。



▲文字比較

基本項目が数字項目であっても，集団項目として比較した場合は文字比較になる。

```
01  A.                    IF A = "12345" THEN ～  ←Aを条件として使った場合は文字比較。
  02  B  PIC  99.                                  文字定数なので「"」で囲む。
  02  C  PIC  999.       IF B = 12      THEN ～  ←Bを条件として使った場合は数字比較。
```

HIGH－VALUE，LOW－VALUE

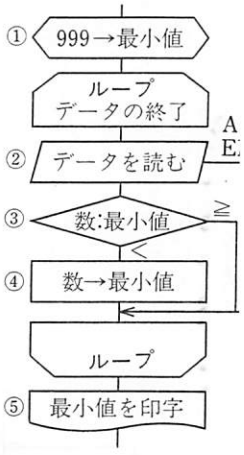
表意定数の一種。文字列の大小順序で，最大のものがHIGH－VALUE，最小のものがLOW－VALUEである。計算機で扱う1文字は1バイト（8ビット）から構成されており，8個の2進数の組み合わせからなる。HIGH－VALUEは8ビットがすべて1のもの，LOW－VALUEはすべて0のものをいい，通常で扱うアルファベットや数字は，すべて1や0のものは存在しない。そのため，文字比較をした場合に，最大や最小になる。

11111111
▲HIGH－VALUE

00000000
▲LOW－VALUE

この例題では，HIGH－VALUE，LOW－VALUEを終了フラグとして使っている。終了フラグは2つの異なる値を取ればよいので，何を用いてもよい。

●最小値の求め方●



最小値は次の手順で求める。

① 最小値に初期値として，データとしてはありえない大きな値をセットする。

②から④までをデータ終了まで繰り返す。

② データを読む。

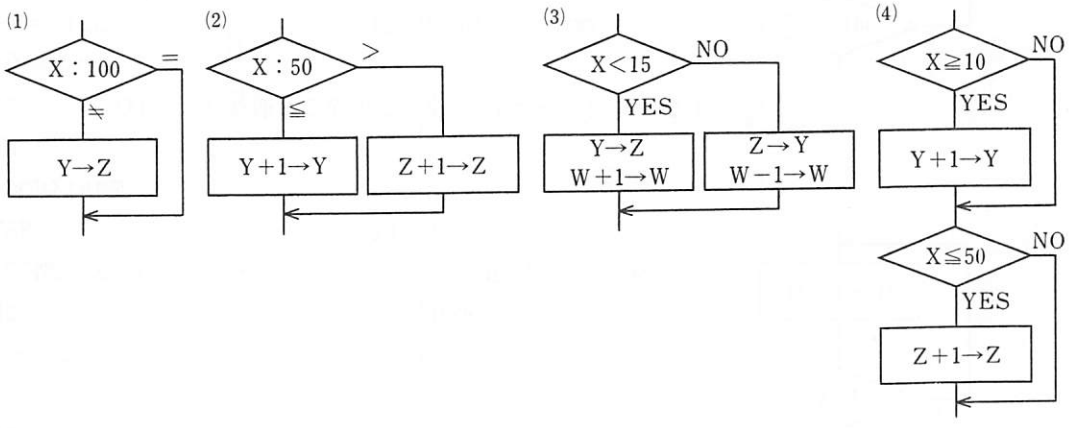
③ 数＜最小値 であれば④にいく。

④ 「数」を最小値として記憶する。

⑤ 最小値を印字する。

●練習問題●

1 次の流れ図をIF文を用いて書きなさい。



2　次の条件で比較したとき，AとBの大小関係を答えなさい。

	A		B	
	PICTURE 句	記憶内容	PICTURE 句	記憶内容
(1)	X (03)	PQR	X (03)	PQY
(2)	X (03)	PQR	X (02)	PQ
(3)	9 (01)	5	9 (03)	005
(4)	X (01)	5	X (03)	005
(5)	X (01)	4	X (03)	4△△

◀実習問題▶

1　健康診断ファイルを読んで，健康診断一覧表を作成しなさい。

▼入力形式

社員コード	社員名	身長	体重
9 (05)	X (10)	9 (03)	9 (03)

▼処理条件

(1)　ローレル指数＝体重＊10^7／身長^3

(2)　身長と体重の最大最小の値を最後に出力する。

▼出力形式

(社員コード)	(社員名)	(身長)	(体重)	(ローレル指数)
99999	XXXXXXXXXX	ZZ9	ZZ9	ZZ9
99999	XXXXXXXXXX	ZZ9	ZZ9	ZZ9
⋮	⋮	⋮	⋮	⋮
	ｻｲﾀﾞｲ	ZZ9	ZZ9	
	ｻｲｼｮｳ	ZZ9	ZZ9	

2　走り幅とび記録ファイルを読んで，走り幅とび成績一覧表を作成しなさい。

▼入力形式

選手コード	選手名	第1回記録	第2回記録
9 (04)	X (08)	9 (03)	9 (03)

▼処理条件

(1)　記録は cm 単位

(2)　第１回記録と第２回記録を比較して大きい方を選手の記録とする。

(3)　最後に最高記録と，その記録を出した選手名を出力する。

▼出力形式

(選手コード)	(選手名)	(記録)
9999	XXXXXXXX	ZZ9
9999	XXXXXXXX	ZZ9
⋮	⋮	⋮
	ｻｲｺｳｷﾛｸ	ZZ9
	ｾﾝｼｭﾒｲ	XXXXXXXX

2　複合条件

例題5　人口増加率が一定範囲の都道府県

入力形式のようなデータ(県別人口ファイル)を読んで，人口増加率一覧表を作成しなさい。

▼入力形式

県コード	県　名	現在の人口	5年前の人口
9 (02)	X (08)	9 (05)	9 (05)

▼処理条件

(1)　人口増加数
　＝現在の人口－5年前の人口

(2)　人口増加率(%)
　＝人口増加数＊100／5年前の人口

(3)　人口増加率が1%以上か－1%以下の都道府県に＊印をつける。

(4)　最後に全国の人口，人口増加率を出力する。

▼出力形式

```
                  *** ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ ***
ｺｰﾄﾞ   ｹ ﾝ ﾒ ｲ      ｼﾞﾝｺｳ   5ﾈﾝﾏｴｼﾞﾝｺｳ      ｿﾞｳｶｽｳ      ｿﾞｳｶﾘﾂ   ﾏｰｸ
 01    ﾎｯｶｲﾄﾞｳ      5,646        5,663         -17      - 0.30
 02    ｱｵﾓﾘ         1,529        1,551         -22      - 1.42    *
 03    ｲﾜﾃ          1,433        1,447         -14      - 0.97
 ｜     ｜            ｜            ｜            ｜          ｜
 46    ｶｺﾞｼﾏ        1,812        1,819          -7      - 0.38
 47    ｵｷﾅﾜ         1,230        1,184          46        3.89    *

       ｾﾞﾝｺｸｹｲ    122,335      120,004       2,331        1.94
```

●問題の分析と考え方●

この例題では，都道府県ごとに人口増加率を求め，

　　人口増加率が　1%以上　または　－1%以下

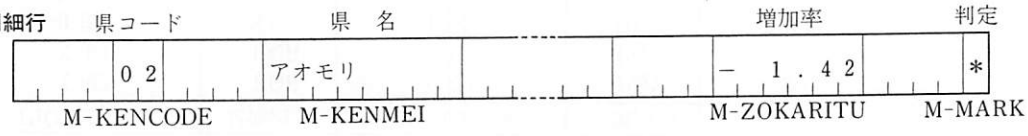
という条件について，特定の処理をしている。

このように，いろいろな比較判断をする場合，1つの判断処理ではなく，2つ以上の判断処理が複合されることが多い。こういう場合に**複合条件**を利用する。複合条件を使用すると，2つのIF文を1つにすることができ，プログラムが簡潔になる。

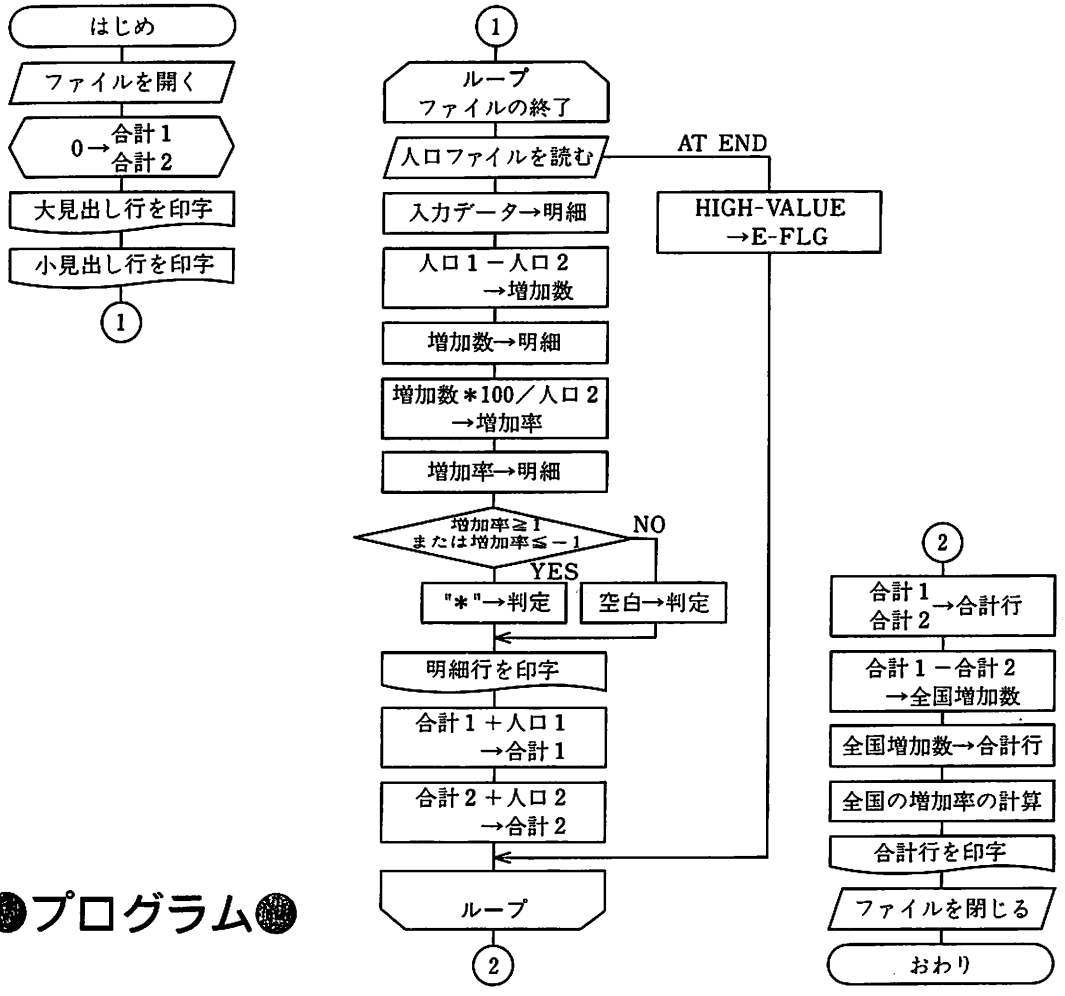
また，人口増加率が一定比率の都道府県には「＊」印をつける処理は，明細行に判定欄を設け，

　　一定比率の場合は「＊」を，そうでない場合は「空白」を

判定欄に転記すればよい。



●流れ図●



●プログラム●

```
DATA                    DIVISION.
FILE                    SECTION.
FD   KEN-FILE.
01   KEN-REC.
     05 KENCODE         PIC  9(02).
     05 KENMEI          PIC  X(08).
     05 JINKO1          PIC  9(05).
     05 JINKO2          PIC  9(05).
FD   ITIRAN-FILE.
01   ITIRAN-REC         PIC  X(132).
WORKING-STORAGE SECTION.
01   E-FLG              PIC  X(01)   VALUE   LOW-VALUE.
01   ZOKASU             PIC S9(05).
01   ZOKARITU           PIC S99V99.
01   ZEN-ZOKASU         PIC S9(05).
01   KEI1               PIC  9(06).
01   KEI2               PIC  9(06).
01   OOMIDASI-GYO       PIC  X(45)   VALUE
     "                    *** ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ ***".
01   KOMIDASI-GYO       PIC  X(67)   VALUE
     "   ｺｰﾄﾞ    ｹ ﾝ ﾒ ｲ       ｼﾞﾝｺｳ  5ﾈﾝﾏｴｼﾞﾝｺｳ      ｿﾞｳｶｽｳ      ｿﾞｳｶﾘ
-    "ﾂ   ﾏｰｸ".
```

```
01  MEISAI-GYO.
    05                  PIC  X(04)   VALUE  SPACE.
    05 M-KENCODE        PIC  9(02).
    05                  PIC  X(04)   VALUE  SPACE.
    05 M-KENMEI         PIC  X(08).
    05                  PIC  X(04)   VALUE  SPACE.
    05 M-JINK01         PIC  ZZ,ZZ9.
    05                  PIC  X(06)   VALUE  SPACE.
    05 M-JINK02         PIC  ZZ,ZZ9.
    05                  PIC  X(04)   VALUE  SPACE.
    05 M-ZOKASU         PIC  ---,--9.
    05                  PIC  X(04)   VALUE  SPACE.
    05 M-ZOKARITU       PIC  -Z9.99.
    05                  PIC  X(04)   VALUE  SPACE.
    05 M-MARK           PIC  X(01).
01  ASIGAKI-GYO.
    05                  PIC  X(10)   VALUE  SPACE.
    05                  PIC  X(11)   VALUE  "ｾﾞﾝｺｸｹｲ".
    05 A-KEI1           PIC  ZZZ,ZZ9.
    05                  PIC  X(05)   VALUE  SPACE.
    05 A-KEI2           PIC  ZZZ,ZZ9.
    05                  PIC  X(04)   VALUE  SPACE.
    05 A-ZOKASU         PIC  ---,--9.
    05                  PIC  X(04)   VALUE  SPACE.
    05 A-ZOKARITU       PIC  -Z9.99.
*
 PROCEDURE DIVISION.
 SYORI.
     OPEN    INPUT KEN-FILE    OUTPUT ITIRAN-FILE
     INITIALIZE    KEI1 KEI2
     WRITE   ITIRAN-REC FROM   OOMIDASI-GYO AFTER    PAGE
     WRITE   ITIRAN-REC FROM   KOMIDASI-GYO AFTER    1
     PERFORM    UNTIL  E-FLG   = HIGH-VALUE
       READ KEN-FILE
         AT END
           MOVE  HIGH-VALUE TO  E-FLG
         NOT AT END
           MOVE  KENCODE     TO  M-KENCODE
           MOVE  KENMEI      TO  M-KENMEI
           MOVE  JINK01      TO  M-JINK01
           MOVE  JINK02      TO  M-JINK02
           COMPUTE ZOKASU    =   JINK01 - JINK02
           MOVE  ZOKASU      TO  M-ZOKASU
           COMPUTE ZOKARITU ROUNDED =  ZOKASU * 100 / JINK02
           MOVE  ZOKARITU    TO  M-ZOKARITU
           IF    ZOKARITU    >= 1  OR  ZOKARITU  <= -1
             THEN  MOVE  "*"     TO     M-MARK
             ELSE  MOVE  SPACE   TO     M-MARK
           END-IF
           WRITE    ITIRAN-REC FROM  MEISAI-GYO  AFTER  1
           COMPUTE KEI1    =   KEI1  +  JINK01
           COMPUTE KEI2    =   KEI2  +  JINK02
       END-READ
     END-PERFORM
     MOVE      KEI1     TO     A-KEI1
     MOVE      KEI2     TO     A-KEI2
     COMPUTE   ZEN-ZOKASU     =      KEI1   -  KEI2
     MOVE      ZEN-ZOKASU     TO     A-ZOKASU
     COMPUTE   A-ZOKARITU     ROUNDED   = ZEN-ZOKASU * 100 / KEI2
     WRITE     ITIRAN-REC     FROM   ASIGAKI-GYO   AFTER  2
     CLOSE     KEN-FILE       ITIRAN-FILE
     STOP      RUN.
```

プログラムの解説

想定小数点

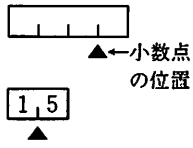
入力領域や作業領域で，小数点以下の数字を利用したい場合は，小数点の位置に「V」を入れる。この「V」のことを**想定小数点**という。ただし，「V」は桁数には含まれず，指定した位置に小数点が確保される。

例

```
01 WORK1 PIC  999V9.
01 WORK2 PIC  9V9  VALUE  1.5.
```

全体の桁数が4桁，小数点以下1桁の領域を確保する。

全体の桁数が2桁，小数点以下1桁の領域を確保し，初期値として1.5を入れる。



▲想定小数点の例

負数の取り扱い

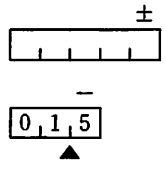
正の数ばかりでなく，負の数も扱う場合は，数値項目のPICTURE句で文字列の先頭に「S」をつける。「S」はデータの桁数には含まれない。実際の正負記号はデータの中に確保される。

例

```
01 RIEKI1 PIC  S9(05).
01 RIEKI2 PIC S99V9 VALUE -1.5.
```

符号つきの5桁の領域を確保する。

符号つき3桁，小数点以下1桁の領域を確保し，初期値を−1.5とする。



▲符号つきデータの例

なお，上記のWORK1，WORK2，RIEKI1，RIEKI2などは，編集された項目ではないので，計算式の右辺に使用できる。

前の行からのプログラムの継続

標識領域である7桁目に「−」を使った場合は，前の行からプログラムが継続していることを表す。データ名も標識領域を使用して継続することができるが，プログラムがわかりにくくなるので使用しない方がよい。継続として利用される場合は，文字定数が多い。

```
7          12                                72
           "....................................
−          "..................".
           "....................................
−          " ".
           "....................................."
           .
```

文字定数が前の行から続く場合

文字定数が前の行でちょうど終わった場合は，継続の行は「"」を2つ書く。

72桁目が「"」でちょうど終わった場合は，次の行の標識領域は空白で，B領域の最初に「.」を1つつける。

▲標識領域を使ったプログラムの継続

いろいろな編集用文字

この例題では，はじめて編集用文字「－」を使用している。ここでいろいろな編集方法と編集用文字を一括して説明する。

編集方法				編集用文字
挿入	特定の文字を指定の場所に挿入する	単純挿入	指定の場所	「，」,「B」,「0」,「／」
		特殊挿入	小数点の位置	「．」
		固定挿入	左端か右端	「¥」,「＋」,「－」
		浮動挿入	有効数字の直前	「¥」,「＋」,「－」
ゼロ抑制	有効数字の前の0を指定の文字に変える。	空白によるゼロ抑制		「Z」
		星印によるゼロ抑制		「＊」

▲編集用文字の種類

なお，「B」「0」「／」は「，」と同様に，それぞれ指定した場合に「空白」「0」「／」を挿入する。

次にいろいろな編集転記例を示す。

送出し側		受取り側		備考
PICTURE	内容	PICTURE	内容	
999V99	00123	ZZ9.99	△△1.23	想定小数点と「．」で桁あわせする
999V99	00000	ZZZ.99	△△△.00	整数部分は空白になる
999V99	00000	ZZZ.ZZ	△△△△△△	すべて空白になる
999V99	12345	ZZ,ZZ9	△△△123	整数部分のみ転記
999V99	12345	Z9.999	23.450	小数点以下が多い部分は0になる
S9 (04)	0034⁻	－Z,ZZ9	－△△△34	送出し側が負の場合は「－」
S9 (04)	0034⁺	－Z,ZZ9	△△△△34	送出し側が正の場合は空白
S9 (04)	0034⁻	＋Z,ZZ9	－△△△34	送出し側が負の場合は「－」
S9 (04)	0034⁺	＋Z,ZZ9	＋△△△34	送出し側が正の場合は「＋」
S9 (05)	00345⁻	－－－－－－	△△－345	－符号が浮動挿入される
S9 (05)	00345⁺	－－－－－－	△△△345	送出し側が正の場合は符号部は空白
S9 (05)	00345⁻	＋＋＋，＋＋＋	△△△－345	－符号が浮動挿入される
S9 (05)	00345⁺	＋＋＋，＋＋＋	△△△＋345	＋符号が浮動挿入される
S9 (05)	00345⁺	**,**9	***345	有効数字の前の0が「＊」になる
S9 (05)	00000	**,***	******	すべて「＊」になる

▲いろいろな編集転記の例

INITIALIZE 文

INITIALIZE 文

```
INITIALIZE　データ名1　（データ名2 ……）
```

数字項目，数字編集項目には0を，英数字項目には空白を設定する文である。データ名が集団項目の場合は，基本項目の内容にあわせて，空白または0が設定される。

例
```
01 A.                                  INITIALIZE  B      ←Bに0を設定する。
   02  B   PIC  9(03).                 INITIALIZE  C      ←Cに空白を設定する。
   02  C   PIC  X(02).                 INITIALIZE  B C    ←Bに0,Cに空白を設定する。
   02  D   PIC  9(02).                 INITIALIZE  A      ←BとDに0,Cに空白を設定する。
```

NOT・AND・OR

複合条件には，次の3つの**論理演算子**を使用する。

論理演算子	演算子の意味	例	IF 文の例
AND	2つの条件がともに満足するとき	Aが10以上100以下	IF A >= 10 AND A <= 100
OR	2つの条件のうち，どちらか一方を満足するとき	Aが10未満または100以上	IF A < 10 OR A >= 100
NOT	条件式の否定をするとき	Aが10以上でない	IF NOT A >= 10

▲論理演算子

論理演算子を複数用いた場合の，優先順位は

①NOT　②AND　③OR

の順であり，順位の変更は算術式と同じように（ ）を用いる。

例
```
IF A > 10  AND  A < 20      ←Aが10より大きくかつ20より小さい場合
IF B < 10  OR   B > 20      ←Bが10より小さいかまたは20より大きい場合
IF NOT C >= 10              ←Cが10以上でない場合
IF D < 10  OR  D > 20  AND  E > 0
        ↑Dが20より大きくかつEが0より大きいか，Dが10より小さい場合
         (ANDを優先する)
IF (D < 10  OR  D > 20)  AND  E > 0
        ↑Dが10より小さいか20より大きく，かつEが0より大きい場合
         (ORを優先する場合はカッコを用いる)
```

●練習問題●

3　次の送出し側から受取り側へ転記された結果はどうなるか答えなさい。

送出し側		受取り側		送出し側		受取り側	
PICTURE	内　容	PICTURE	内　容	PICTURE	内　容	PICTURE	内　容
9(06)	123456	ZZ,ZZZ	ア	S9(05)	12345⁻	------	ケ
999V99	12121	ZZ9.9	イ	S9(05)	12345⁺	++++++	コ
999V99	12345	ZZZ9.999	ウ	S9(05)	00345⁺	−¥¥¥¥9	サ
999V99	01234	ZZ.999	エ	S9(05)	00345⁻	+¥¥¥¥9	シ
V999	987	ZZ.999	オ	S9(05)	00345⁺	−¥¥¥¥9	ス
999V999	123456	ZZZZ.9	カ	S9(05)	00345⁺	**,***	セ
99V9	987	Z,ZZ9	キ	S9(05)	00345⁺	+++,++9	ソ
9999V9	98765	****.*	ク	S9(05)	01345⁻	---,--9	タ

◀実習問題▶

3　健康診断ファイルを読んで，健康診断一覧表を作成しなさい。

▼入力形式

社員コード	社員名	身長	体重
9 (05)	X (10)	999V9	999V9

▼処理条件

(1)　ローレル指数＝体重＊10^7／身長^3

(2)　ローレル指数が114～144の場合は備考欄に「フツウ」と印字する。

▼出力形式

```
                ***　ケンコウシンタﾞン　イチランヒヨウ　***

ｺｰﾄﾞ　　ｼﾔｲﾝﾒｲ　　ｼﾝﾁﾖｳ　　ﾀｲｼﾞｭｳ　ﾛｰﾚﾙｼｽｳ　ﾋﾞｺｳ
99999　　XXXXXXXXXX　　ZZ9.9　　ZZ9.9　　ZZ9
99999　　XXXXXXXXXX　　ZZ9.9　　ZZ9.9　　ZZ9　　ﾌﾂｳ
99999　　XXXXXXXXXX　　ZZ9.9　　ZZ9.9　　ZZ9
  ｛　　　　｛　　　　｛　　　｛　　　｛
          ﾍｲｷﾝ　　ZZ9.9　　ZZ9.9　　ZZ9
```

4　ある都市の8月の気温ファイルを読んで，気温一覧表を作成しなさい。

▼入力形式

日付	最高気温	最低気温
9 (04)	S99V9	S99V9

▼処理条件

(1)　温度差＝最高気温－最低気温

(2)　真夏日（最高気温が30°以上）でかつ熱帯夜（最低気温が25°以上）の場合備考欄に＊をつける。

(3)　最後に真夏日，熱帯夜，真夏日でかつ熱帯夜の日数および温度差の一番小さい日を出力する。

▼出力形式

```
                ***　キオン　イチランヒヨウ　***

ﾋﾂﾞｹ　　ｻｲｺｳｷｵﾝ　　ｻｲﾃｲｷｵﾝ　　ｵﾝﾄﾞｻ　ﾋﾞｺｳ
99／99　　Z9.9　　Z9.9　　Z9.9
99／99　　Z9.9　　Z9.9　　Z9.9　　＊
99／99　　Z9.9　　Z9.9　　Z9.9
  ｛　　　｛　　　｛　　　｛
  ﾏﾅﾂﾋﾞ　　Z9　ﾆﾁ
  ﾈﾂﾀｲﾔ　　Z9　ﾆﾁ
  ﾏﾅﾂﾋﾞﾃﾞﾈﾂﾀｲﾔ　　Z9　ﾆﾁ
  ｵﾝﾄﾞｻﾉｲﾁﾊﾞﾝﾁｲｻｲﾋ　　99／99
```

3　条件名条件

例題6　一定の条件をもつ都道府県

県ファイルを読んで，出力形式のような都道府県別一覧表を作成しなさい。

▼入力形式

県コード	県　名	面　積	人　口
9 (02)	X (08)	9 (05)	9 (05)

▼処理条件

(1)　人口が500万人以上でかつ人口密度が1000人以上の都道府県に星印をつける（人口の単位は千人）。

(2)　最後に全国の面積，人口，人口密度を出力する。

▼出力形式

```
                    ***  ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ  ***
ｺｰﾄﾞ    ｹ ﾝ ﾒ ｲ        ﾒﾝｾｷ        ｼﾞﾝｺｳ        ﾐｯﾄﾞ     ﾏｰｸ
 01     ﾎｯｶｲﾄﾞｳ      83,520       5,646         68
 02     ｱｵﾓﾘ          9,619        1,529        159
 03     ｲﾜﾃ          15,277        1,433         94
 ｜       ｜            ｜           ｜           ｜
 11     ｻｲﾀﾏ          3,799        6,194      1,630      *
 12     ﾁﾊﾞ           5,151        5,415      1,051      *
 ｜       ｜            ｜           ｜           ｜
 46     ｶｺﾞｼﾏ         9,167        1,812        198
 47     ｵｷﾅﾜ          2,255        1,230        545

        ｾﾞﾝｺｸｹｲ     377,682      122,335        324
```

●問題の分析と考え方●

この例題の条件判定は，今まで学習してきた内容で記述すると，

```
IF  JINKO >= 5000 AND MITUDO >= 1000
              THEN  MOVE  "*"  TO  M-HOSI
              ELSE  ~
  END-IF
```

このような条件判定の記述は，最初にデータ部において条件名で記述すると，簡潔になる。

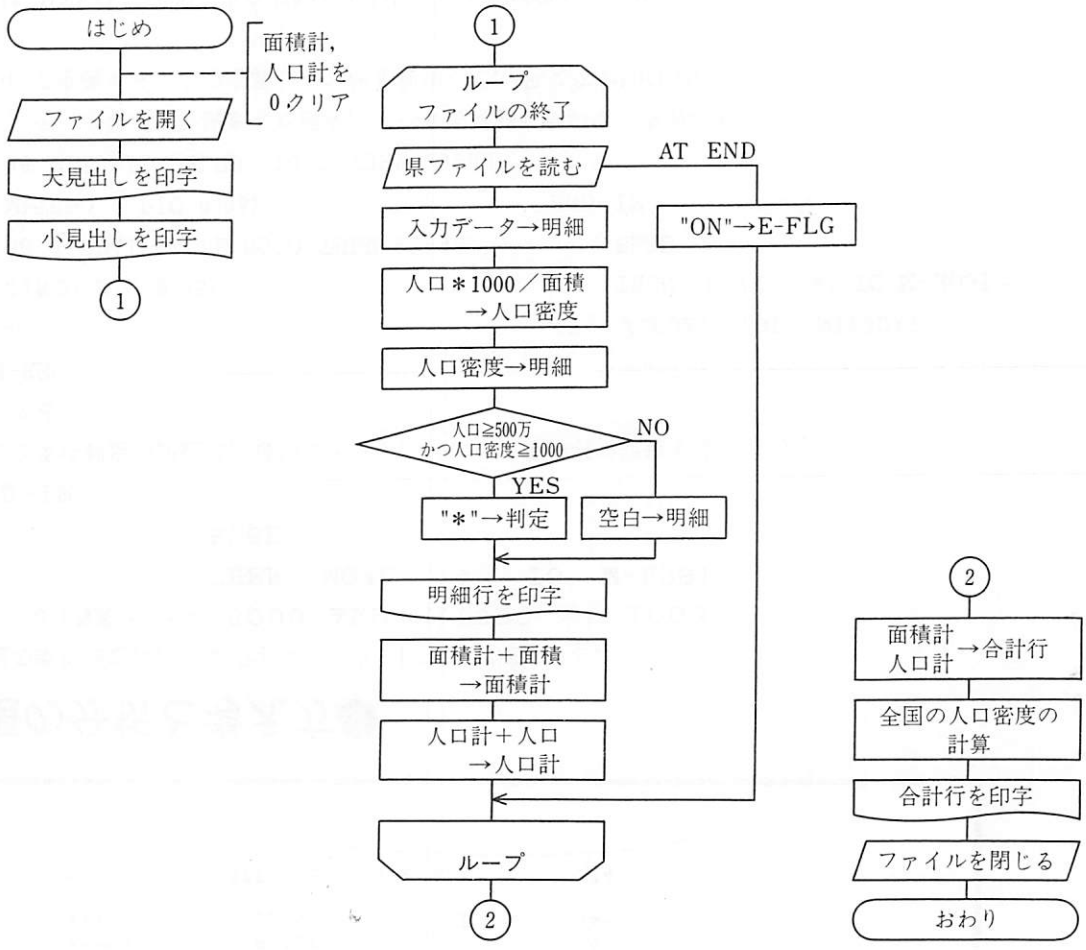
```
01 KEN-REC.
     ..............
   02 JINKO PIC 9(05).
      88 JINDAI VALUE 5000 THRU 99999.
   02 MITUDO   PIC 9(05).
      88 MITUDAI VALUE  1000 THRU 99999.
```

```
     ..............
IF   JINDAI  AND  MITUDAI
     THEN  MOVE  "*" TO M-HOSI
     ELSE  ~
END-IF
```

条件名は，データ名が取る決まった値を，レベル番号88を用いて，事前にある名前をつけて定義する。その定義した名前を利用して，IF 文や PERFORM 文を書く。

上記の例は，JINKO が5000以上を JINDAI, MITUDO が1000以上を MITUDAI という条件名をつけて，条件名条件として IF 文を利用している。

●流れ図●



●プログラム●

```
DATA                    DIVISION.
FILE                    SECTION.
FD   KEN-FILE.
01   KEN-REC.
     05 KENCODE         PIC 9(02).
     05 KENMEI          PIC X(08).
     05 MENSEKI         PIC 9(05).
     05 JINKO           PIC 9(05).
        88 JINDAI       VALUE   5000  THRU  99999.
FD   ITIRAN-FILE.
01   ITIRAN-REC         PIC X(132).
WORKING-STORAGE SECTION.
01   E-FLG              PIC X(03)    VALUE  "OFF".
     88   EOF                        VALUE  "ON".
01   MITUDO             PIC 9(05).
     88   MITUDAI                    VALUE  1000  THRU  99999.
01   MENSEKIKEI         PIC 9(06)    VALUE  0.
01   JINKOKEI           PIC 9(06)    VALUE  0.
01   OOMIDASI-GYO       PIC X(50)    VALUE
     "                          *** ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ ***".
```

```
 01  KOMIDASI-GYO     PIC X(64)   VALUE
     "        ｺｰﾄﾞ     ｹ ﾝ ﾒ ｲ        ﾒﾝｾｷ       ｼﾞﾝｺｳ          ﾐｯﾄﾞ
-    " ﾏｰｸ".
 01  MEISAI-GYO.
     05                PIC X(09)   VALUE  SPACE.
     05 M-KENCODE     PIC 9(02).
     05                PIC X(05)   VALUE  SPACE.
     05 M-KENMEI      PIC X(08).
     05                PIC X(05)   VALUE  SPACE.
     05 M-MENSEKI     PIC ZZ,ZZ9.
     05                PIC X(05)   VALUE  SPACE.
     05 M-JINKO       PIC ZZ,ZZ9.
     05                PIC X(05)   VALUE  SPACE.
     05 M-MITUDO      PIC ZZ,ZZ9.
     05                PIC X(05)   VALUE  SPACE.
     05 MARK-M        PIC X(01).
 01  GOKEI-GYO.
     05                PIC X(16)   VALUE  SPACE.
     05                PIC X(07)   VALUE  "ｾﾞﾝｺｸｹｲ".
     05                PIC X(05)   VALUE  SPACE.
     05 G-MENSEKI     PIC ZZZ,ZZ9.
     05                PIC X(04)   VALUE  SPACE.
     05 G-JINKO       PIC ZZZ,ZZ9.
     05                PIC X(05)   VALUE  SPACE.
     05 G-MITUDO      PIC ZZ,ZZ9.
*
 PROCEDURE DIVISION.
 SYORI.
     OPEN    INPUT KEN-FILE    OUTPUT ITIRAN-FILE
     WRITE   ITIRAN-REC FROM  OOMIDASI-GYO  AFTER  PAGE
     WRITE   ITIRAN-REC FROM  KOMIDASI-GYO  AFTER  1
     PERFORM    UNTIL   EOF
       READ     KEN-FILE
         AT END
           MOVE  "ON"     TO  E-FLG
         NOT AT END
           MOVE  KENCODE  TO  M-KENCODE
           MOVE  KENMEI   TO  M-KENMEI
           MOVE  MENSEKI  TO  M-MENSEKI
           MOVE  JINKO    TO  M-JINKO
           COMPUTE MITUDO     ROUNDED  =
                   JINKO  *  1000  /  MENSEKI
           MOVE  MITUDO   TO  M-MITUDO
           IF    JINDAI   AND MITUDAI
             THEN
               MOVE  "*"      TO  MARK-M
             ELSE
               MOVE  SPACE    TO  MARK-M
           END-IF
           WRITE ITIRAN-REC   FROM  MEISAI-GYO  AFTER  1
           ADD   MENSEKI      TO    MENSEKIKEI
           ADD   JINKO        TO    JINKOKEI
       END-READ
     END-PERFORM
     MOVE      MENSEKIKEI     TO       G-MENSEKI
     MOVE      JINKOKEI       TO       G-JINKO
     COMPUTE   G-MITUDO       ROUNDED =
               JINKOKEI   *   1000  /  MENSEKIKEI
     WRITE     ITIRAN-REC     FROM     GOKEI-GYO  AFTER 2
     CLOSE     KEN-FILE       ITIRAN-FILE
     STOP      RUN.
```

●プログラムの解説●

条件名条件*

基本項目の定義に続けて，レベル番号88を用いた条件名を定義すると，基本項目に対して，特定の条件のときに名前をつけたことになり，IF文やPERFORM文において

```
IF          条件名
PERFORM UNTIL    条件名
```

と書くだけですむ。

* IF文においていろいろな条件が記述される。p.42のように，比較演算子を用いた条件を正式には比較条件というのに対し，条件名を用いる場合を条件名条件という。

条件名条件(1)

```
88    条件名    VALUE    定数.                                    ……①
88    条件名    VALUE    定数1  { THRU / THROUGH }  定数2.        ……②
```

①は条件が1つだけの場合，②は条件が範囲（定数1＜定数2）をとる場合である。

例 データ名SEIBETUは，1が男性，2が女性である。

〔条件名を利用した場合〕

```
01   SEIBETU   PIC 9.
     88   DANSEI   VALUE   1.
     88   JOSEI    VALUE   2.
IF DANSEI
   THEN
     男性の処理
   ELSE
     女性の処理
END-IF
```

〔条件名を利用しない場合〕

```
IF SEIBETU = 1
   THEN
     男性の処理
   ELSE
     女性の処理
END-IF
```

THRU句を利用した条件名は，複合条件のANDと同じ働きをもつ。

例

```
02   NENREI   PIC 9(2).
     88 SHOUGAKU VALUE 6 THRU 11.
IF   SHOUGAKU      ← IF NENREI >= 6 AND NENREI <= 11と同じ。
     THEN  ……
```

条件名条件(2)

```
88    条件名    VALUE    定数1  定数2  …….
```

データ記述項が連続する値でなく，いくつかの値をとる場合は，VALUE句にそのとる定数を並べる。この条件名は複合条件のORと同じ働きをもつ。

例

```
02   TEN       PIC 9(2).
     88 POINTO   VALUE 20   30.
IF   POINTO          ← IF TEN = 20 OR TEN = 30 と同じ。
```

終了フラグに条件名を用いる場合

終了フラグに条件名を用いると，PERFORM 文は UNTIL 条件名だけでよい。

例
```
01   E-FLG   PIC  X(03)  VALUE   "OFF".      ←初期値として"OFF"を入れておく。
     88   EOF              VALUE   "ON".      ←条件名として"ON"の入った名前を定義する。
          ..............
     PERFORM     UNTIL   EOF                 ←PERFORM 文の UNTIL 句は条件名だけでよい。
       READ KEN-FILE
            AT END    MOVE  "ON"  TO  E-FLG  ←ファイルが終わったならば"ON"をE-FLGに入れる。
```

COMPUTE 文以外の算術文*

今までは，四則演算は COMPUTE 文を用いてきたが，COBOL には，単独で加減乗除が計算できる文も用意されている。ただし，2 つの演算（たとえば，加算と除算，減算と乗算など）を同時に行う場合は，COMPUTE 文を利用しなければならない。

* COMPUTE 文とここで説明する 4 つの文をあわせて**算術文**という。

加算　ADD 文　　減算　SUBTRACT 文
乗算　MULTIPLY 文　　除算　DIVIDE 文

	算術文の例	COMPUTE 文に直した場合
ADD	ADD B TO A	COMPUTE A = A + B
	ADD 5 C TO A	COMPUTE A = A + 5 + C
	ADD 1 TO A GIVING G	COMPUTE G = A + 1
	ADD B C D TO A GIVING G	COMPUTE G = A + B + C + D
SUBTRACT	SUBTRACT B FROM A	COMPUTE A = A − B
	SUBTRACT 5 C FROM A	COMPUTE A = A − (5 + C)
	SUBTRACT 1 FROM A GIVING G	COMPUTE G = A − 1
	SUBTRACT B C D FROM A GIVING G	COMPUTE G = A − (B + C + D)
MULTIPLY	MULTIPLY 5 BY A	COMPUTE A = A * 5
	MULTIPLY B BY A GIVING G	COMPUTE G = A * B
DIVIDE	DIVIDE 5 INTO A	COMPUTE A = A / 5
	DIVIDE B INTO A GIVING G	COMPUTE G = A / B
	DIVIDE B BY A GIVING G	COMPUTE G = B / A

●練習問題●

4　次の条件で条件名の定義をしなさい。

(1)　項目名　YAKUSYOKU　英数字項目 1 桁
　　条件名　BUCHO　1　KACHO　2　KAKARICHO　3

(2)　項目名　NENREI　数字項目 2 桁
　　条件名　SYONEN　15～19　SEINEN　20～39
　　CHUNEN　40～49　JITUNEN　50～69
　　RONEN　70～

◀実習問題▶

5 健康診断ファイルを読んで，健康診断一覧表を作成しなさい。

▼入力形式

社員番号	性別	社員名	身長	体重
9 (05)	9 (01)	X (10)	999V9	999V9

▼処理条件

(1) ローレル指数
＝体重＊10^7／身長^3

(2) 備考欄 ローレル指数
145以上 ＋印

(3) 性別(男＝1)の場合に，性別欄に星印をつける。

(4) 男女別に平均を算出する。

(5) 条件名条件を使用する。

▼出力形式

```
                    ***  ｹﾝｺｳｼﾝﾀﾞﾝ ｲﾁﾗﾝﾋｮｳ  ***

ｺｰﾄﾞ  ｾｲﾍﾞﾂ      ｼﾔｲﾝﾒｲ      ｼﾝﾁﾖｳ   ﾀｲｼﾞｭｳ ﾛｰﾚﾙｼｽｳ ﾋﾞｺｳ
99999   *       XXXXXXXXXX  ZZ9.9   ZZ9.9   ZZ9
99999           XXXXXXXXXX  ZZ9.9   ZZ9.9   ZZ9     +
99999   *       XXXXXXXXXX  ZZ9.9   ZZ9.9   ZZ9     +
  ｝               ｝         ｝       ｝       ｝
          ﾀﾞﾝｼ ﾍｲｷﾝ          ZZ9.9   ZZ9.9   ZZ9
          ｼﾞｮｼ ﾍｲｷﾝ          ZZ9.9   ZZ9.9   ZZ9
```

6 給与ファイルを読んで，給与一覧表を作成しなさい。

▼入力形式

職員コード	名前	役職コード	本給	加給	控除額
9 (04)	X (08)	X (01)	9 (07)	9 (06)	9 (06)

▼処理条件 (条件名条件を用いること)

(1) 役職コードが1の場合に加給を¥5,000ふやし，役職コード欄に星印をつける。

(2) 総支給額＝本給＋加給

(3) 現金支給額＝総支給額－控除額

(4) 最後に総支給額，控除額，現金支給額の合計を印字する。

▼出力形式

```
                      ***  ｷﾕｳﾖ ｲﾁﾗﾝﾋｮｳ  ***

ｺｰﾄﾞ    ﾅ ﾏ ｴ      ﾔｸｼｮｸ  ﾎﾝｷﾕｳ       ｶｷﾕｳ       ｺｳｼﾞｮ      ｼｷﾕｳｶﾞｸ
9999    XXXXXXXX     *   Z,ZZZ,ZZ9   ZZZ,ZZ9   ZZZ,ZZ9   ¥¥,¥¥¥,¥¥9
9999    XXXXXXXX         Z,ZZZ,ZZ9   ZZZ,ZZ9   ZZZ,ZZ9   ¥¥,¥¥¥,¥¥9
9999    XXXXXXXX     *   Z,ZZZ,ZZ9   ZZZ,ZZ9   ZZZ,ZZ9   ¥¥,¥¥¥,¥¥9
  ｝       ｝        ｝       ｝         ｝         ｝          ｝
                     ｺﾞｳｹｲ Z,ZZZ,ZZ9   ZZZ,ZZ9   ¥¥¥,¥¥¥,¥¥9
```

4　EVALUATE 文

例題7　複雑な条件による区分

県ファイルを読んで，出力形式のような都道府県一覧表を作成しなさい。

▼入力形式

県コード	県名	面積	人口
9 (02)	X (08)	9 (05)	9 (05)

▼処理条件

人口密度によって，区分欄に次のマークをつける。

1000人以上	＋＋	150～249	－
500～999	＋	149人以下	－－
250～499	空白		

▼出力形式

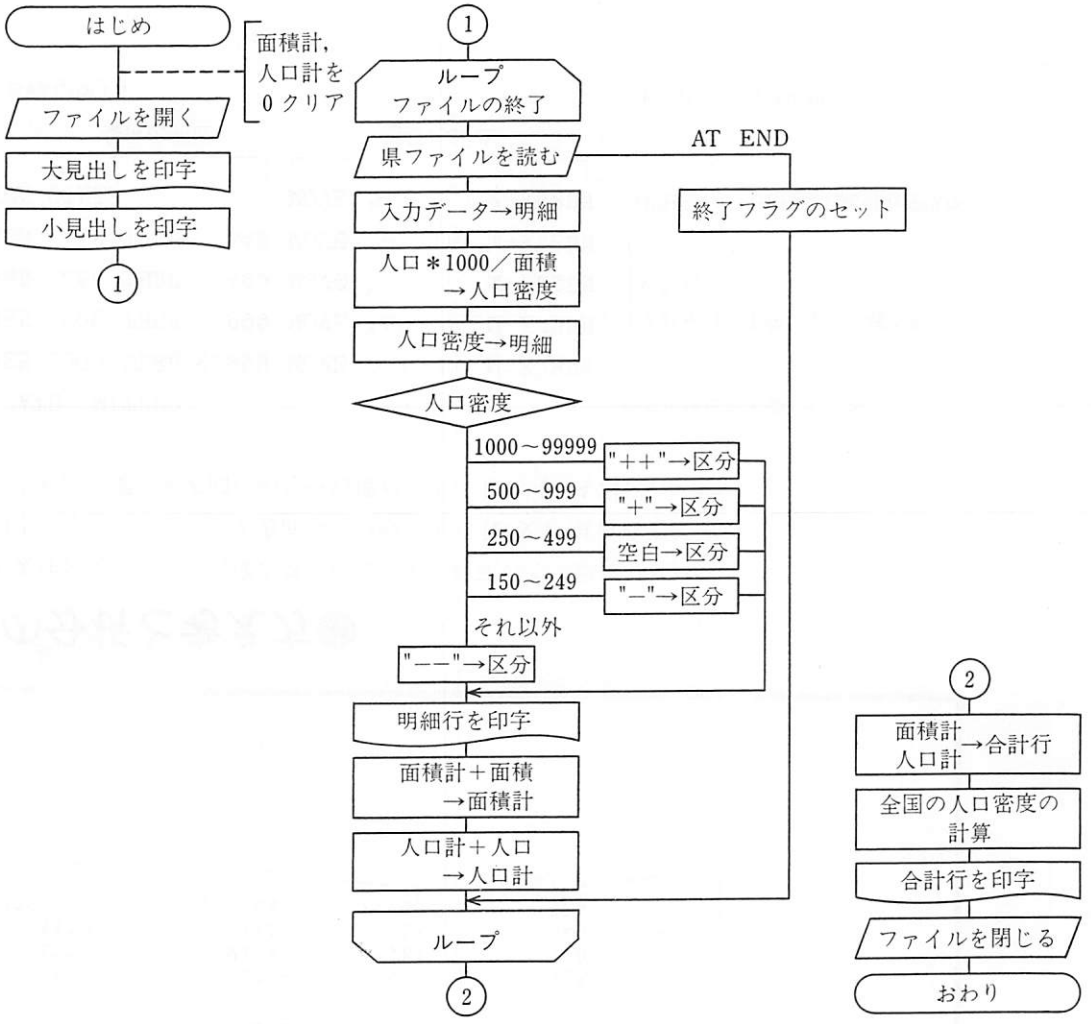
```
                    *** ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ ***
ｺｰﾄﾞ     ｹ ﾝ ﾒ ｲ        ﾒﾝｾｷ        ｼﾞﾝｺｳ          ﾐｯﾄﾞ     ｸﾌﾞﾝ
 01      ﾎｯｶｲﾄﾞｳ      83,520       5,646          68      --
 02      ｱｵﾓﾘ          9,619       1,529         159      -
 03      ｲﾜﾃ          15,277       1,433          94      --
 04      ﾐﾔｷﾞ          7,292       2,210         303
 05      ｱｷﾀ          11,613       1,245         107      --
  ⌇       ⌇             ⌇           ⌇             ⌇
 45      ﾐﾔｻﾞｷ         7,735       1,184         153      -
 46      ｶｺﾞｼﾏ         9,167       1,812         198      -
 47      ｵｷﾅﾜ          2,255       1,230         545      +
ｾﾞﾝｺｸｹｲ             377,682     122,335         324
```

●問題の分析と考え方●

この例題の条件判定は，人口密度の数字によって処理がいくつかのパターンになる。このような条件判定に，2方向への分岐であるIF文を用いると処理が複雑になるため，一度に多方向への分岐処理ができるEVALUATE文を用いる。

```
EVALUATE  MITUDO                                    ⇦条件判定をするデータ名
  WHEN 1000 THRU 99999 MOVE "++" TO M-KUBUN  ┐
  WHEN  500 THRU   999 MOVE "+"  TO M-KUBUN  ├ それぞれの条件による処理内容
  WHEN  250 THRU   499 MOVE " "  TO M-KUBUN  ├ の記述
  WHEN  150 THRU   249 MOVE "-"  TO M-KUBUN  ┘
  WHEN OTHER           MOVE "--" TO M-KUBUN   ⇦WHEN OTHERは上記の条件以外
       └──条件の記述──┘ └──処理の記述──┘
END-EVALUATE                                        ⇦EVALUATE文の範囲
```

●流れ図●



●プログラム●

```
DATA                    DIVISION.
FILE                    SECTION.
FD   KEN-FILE.
01   KEN-REC.
     05 KENCODE         PIC 9(02).
     05 KENMEI          PIC X(08).
     05 MENSEKI         PIC 9(05).
     05 JINKO           PIC 9(05).
FD   ITIRAN-FILE.
01   ITIRAN-REC         PIC X(132).
WORKING-STORAGE SECTION.
01   E-FLG              PIC X(01)    VALUE   LOW-VALUE.
     88   EOF                        VALUE   HIGH-VALUE.
01   MITUDO             PIC 9(05).
01   MENSEKIKEI         PIC 9(06)    VALUE   0.
01   JINKOKEI           PIC 9(06)    VALUE   0.
```

```
 01  OOMIDASI-GYO     PIC X(50)    VALUE
     "                        *** ﾄﾄﾞｳﾌｹﾝ ﾍﾞﾂ ｲﾁﾗﾝﾋｮｳ ***".
 01  KOMIDASI-GYO     PIC X(68)    VALUE
     "         ｺｰﾄﾞ     ｹ ﾝ ﾒ ｲ          ﾒﾝｾｷ        ｼﾞﾝｺｳ          ﾐｯﾄﾞ
-    " ｸﾌﾞﾝ".
 01  MEISAI-GYO.
     05                PIC X(09)    VALUE   SPACE.
     05 M-KENCODE     PIC 9(02).
     05                PIC X(05)    VALUE   SPACE.
     05 M-KENMEI      PIC X(08).
     05                PIC X(05)    VALUE   SPACE.
     05 M-MENSEKI     PIC ZZ,ZZ9.
     05                PIC X(05)    VALUE   SPACE.
     05 M-JINKO       PIC ZZ,ZZ9.
     05                PIC X(05)    VALUE   SPACE.
     05 M-MITUDO      PIC ZZ,ZZ9.
     05                PIC X(05)    VALUE   SPACE.
     05 M-KUBUN       PIC X(02).
 01  GOKEI-GYO.
     05                PIC X(09)    VALUE  SPACE.
     05                PIC X(08)    VALUE  "ｾﾞﾝｺｸｹｲ".
     05                PIC X(11)    VALUE  SPACE.
     05 G-MENSEKI     PIC ZZZ,ZZ9.
     05                PIC X(04)    VALUE  SPACE.
     05 G-JINKO       PIC ZZZ,ZZ9.
     05                PIC X(05)    VALUE  SPACE.
     05 G-MITUDO      PIC ZZ,ZZ9.
*
 PROCEDURE DIVISION.
 SYORI.
     OPEN   INPUT KEN-FILE    OUTPUT ITIRAN-FILE
     WRITE  ITIRAN-REC  FROM  OOMIDASI-GYO  AFTER PAGE
     WRITE  ITIRAN-REC  FROM  KOMIDASI-GYO  AFTER 1
     PERFORM  UNTIL   EOF
       READ   KEN-FILE
         AT END
           SET     EOF       TO     TRUE
         NOT AT END
           MOVE    KENCODE   TO     M-KENCODE
           MOVE    KENMEI    TO     M-KENMEI
           MOVE    MENSEKI   TO     M-MENSEKI
           MOVE    JINKO     TO     M-JINKO
           COMPUTE MITUDO    ROUNDED  =
                   JINKO  * 1000 / MENSEKI
           MOVE    MITUDO    TO     M-MITUDO
           EVALUATE  MITUDO
             WHEN 1000  THRU 99999   MOVE "++" TO M-KUBUN
             WHEN  500  THRU   999   MOVE "+"  TO M-KUBUN
             WHEN  250  THRU   499   MOVE " "  TO M-KUBUN
             WHEN  150  THRU   249   MOVE "-"  TO M-KUBUN
             WHEN OTHER              MOVE "--" TO M-KUBUN
           END-EVALUATE
           WRITE   ITIRAN-REC   FROM  MEISAI-GYO  AFTER 1
           ADD     MENSEKI      TO    MENSEKIKEI
           ADD     JINKO        TO    JINKOKEI
       END-READ
     END-PERFORM
     MOVE      MENSEKIKEI  TO    G-MENSEKI
     MOVE      JINKOKEI    TO    G-JINKO
     COMPUTE   G-MITUDO    ROUNDED  =
               JINKOKEI   *  1000  /  MENSEKIKEI
     WRITE     ITIRAN-REC FROM  GOKEI-GYO  AFTER  1
     CLOSE     KEN-FILE    ITIRAN-FILE
     STOP      RUN.
```

●プログラムの解説●

SET 文

SET 文

```
SET    条件名    TO    TRUE
```

レベル番号88で定義した条件名条件を，VALUE 句で定義した値に設定する。今までMOVE 文で行っていた処理を，SET 文で行う。条件名条件で定義した値を自動的に設定するため，手続き部において送出し側の内容を指示する必要がない。

例

```
01   E-FLG   PIC X(03)   VALUE  "OFF".     ←終了フラグに初期値のセット
     88  EOF              VALUE  "ON".      ←条件名条件の定義
          ..............
     READ  KEN-FILE   AT  END
                SET   EOF   TO   TRUE      ←このSET文でE-FLGが"ON"になる
```

EVALUATE 文

EVALUATE 文(1)

（書き方1）

```
EVALUATE  データ名
   WHEN    条件1        文1
   WHEN    条件2        文2
      ..............
   WHEN    OTHER        文n
END-EVALUATE
```

（書き方2）

```
EVALUATE  TRUE
   WHEN  条件式1(条件名1)    文1
   WHEN  条件式2(条件名2)    文2
      ..............
   WHEN  OTHER               文n
END-EVALUATE
```

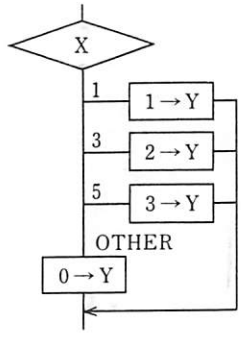
IF 文では2つの方向への分岐だけであったが，多方向への分岐の場合は，EVALUATE 文を用いるとプログラムがわかりやすい。

例　（書き方1）

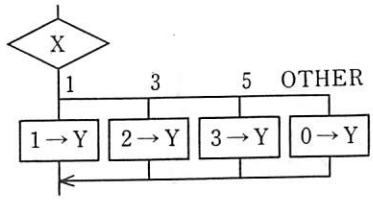
```
EVALUATE X
    WHEN  1     MOVE 1 TO Y
    WHEN  3     MOVE 2 TO Y
    WHEN  5     MOVE 3 TO Y
    WHEN OTHER  MOVE 0 TO Y
END-EVALUATE
```

（書き方2）

```
EVALUATE TRUE
    WHEN X = 1  MOVE 1 TO Y
    WHEN X = 3  MOVE 2 TO Y
    WHEN X = 5  MOVE 3 TO Y
    WHEN OTHER  MOVE 0 TO Y
END-EVALUATE
```



また，EVALUATE 文の流れ図は右のどちらの形で書いてもよい。



EVALUATE の次に書く記述を**選択主体**，WHEN の次に書く記述を選択対象という。

書き方1において，選択対象の条件が単独である場合は，その値を記述すればよいが，範囲を取る場合は，THRU 句を使う。書き方2においては，THRU 句は使えない。

また，選択対象にも条件名条件を使うことができる。例題を条件名条件を用いると

```
01   MITUDO    PIC   9 (05).
     88   RANK1     VALUE   1000   THRU   99999.
     88   RANK2     VALUE    500   THRU     999.
     88   RANK3     VALUE    250   THRU     499.
     88   RANK4     VALUE    150   THRU     249.
                        ..............
     EVALUATE   TRUE
          WHEN   RANK1        MOVE   "++"   TO   M-KUBUN
          WHEN   RANK2        MOVE   "+"    TO   M-KUBUN
          WHEN   RANK3        MOVE   "  "   TO   M-KUBUN
          WHEN   RANK4        MOVE   "-"    TO   M-KUBUN
          WHEN   OTHER        MOVE   "--"   TO   M-KUBUN
     END-EVALUATE
```

EVALUATE 文(2)

```
EVALUATE  データ名1    ALSO  データ名2    ……
     WHEN    条件11    ALSO  条件12       ……     文1
     WHEN    条件21    ALSO  条件22       ……     文2
        ……              ……                     ……
     WHEN  OTHER                                  文n
END-EVALUATE
```

選択主体のデータが2つ以上ある多分岐の処理には，EVALUATE 文に ALSO 句をつけると可能になる。

例　筆記試験　HIKKI　英数字項目1桁
　　面接試験　MENSETU　英数字項目1桁
　　合否結果　GOHI　英数字項目7桁

筆記試験	面接試験	合否
A	A	採　用
A	B	補欠1
B	A	採　用
B	B	補欠2
この条件以外		不採用

```
EVALUATE  HIKKI  ALSO  MENSETU
     WHEN "A" ALSO "A" MOVE "SAIYO"    TO GOHI
     WHEN "A" ALSO "B" MOVE "HOKETU1"  TO GOHI
     WHEN "B" ALSO "A" MOVE "SAIYO"    TO GOHI
     WHEN "B" ALSO "B" MOVE "HOKETU2"  TO GOHI
     WHEN OTHER        MOVE "HUSAIYO"  TO GOHI
END-EVALUATE
```

選択主体のデータ名をALSO句で結び，選択対象においては，選択主体のデータ名の順番に条件をALSO句で結んで定義することで，データ名と条件が対比される。

例　役職コード　YAKUSYOKU　英数字項目1桁
　　配偶者コード　HAIGUSYA　英数字項目1桁
　　手当　TEATE　数字項目5桁

役職コード	配偶者コード	手当
0	関係なし	5,000
1	1	10,000
2～3	0	15,000
2～3	1	20,000
4	関係なし	50,000

```
EVALUATE YAKUSYOKU ALSO HAIGUSYA
    WHEN "0" ALSO ANY  MOVE  5000 TO TEATE
    WHEN "1" ALSO "1"  MOVE 10000 TO TEATE
    WHEN "2" THRU "3"
             ALSO "0"  MOVE 15000 TO TEATE
    WHEN "2" THRU "3"
             ALSO "1"  MOVE 20000 TO TEATE
    WHEN "4" ALSO ANY  MOVE 50000 TO TEATE
    WHEN OTHER         CONTINUE
END-EVALUATE
```

選択対象がすべての所は，ANY句をつける。処理が何もない所はIF文と同じように，CONTINUE文を書く。

EVALUATE文(3)

```
EVALUATE  条件式
  WHEN  TRUE    文1
  WHEN  FALSE   文2
END-EVALUATE
```

選択主体を条件式にして，選択対象に，TRUE句とFALSE句を用いる方法もある。TRUEは「真」すなわち条件を満足することで，FALSEは「偽」すなわち条件を満足しないということである。

例　X＞Yならば，Z＝X－Y，そうでなければ，Z＝Y－Xの計算をする。

```
EVALUATE X > Y
    WHEN  TRUE  COMPUTE  Z = X - Y
    WHEN  FALSE COMPUTE  Z = Y - X
END-EVALUATE
```

この例は次のIF文と同じことである。

```
IF  X  >  Y
    THEN     COMPUTE  Z = X - Y
    ELSE     COMPUTE  Z = Y - X
END-IF
```

すなわち，IF文の表記はすべてEVALUATE文で表現できる。

IF 文の入れ子

IF 文の中に IF 文を書くことを**入れ子**または**ネスト**という。COBOL 85では，EVALUATE 文が追加されたのであまり使われなくなった。

例題を EVALUATE 文と IF 文のネスト構造を対比させる。THEN, ELSE, END−IF の使い方に注意すること。

[EVALUATE 文]

```
EVALUATE   MITUDO
  WHEN 1000 THRU 99999
           MOVE "++" TO M-KUBUN
  WHEN  500 THRU   999
           MOVE "+"  TO M-KUBUN
  WHEN  250 THRU   499
           MOVE "  "  TO M-KUBUN
  WHEN  150 THRU   249
           MOVE "-"  TO M-KUBUN
  WHEN OTHER
           MOVE "--" TO M-KUBUN
END-EVALUATE
```

[IF 文]

```
IF MITUDO >= 1000
  THEN
      MOVE "++" TO M-KUBUN
  ELSE  IF  MITUDO >= 500
          THEN
              MOVE "+"  TO M-KUBUN
          ELSE  IF  MITUDO >= 250
                  THEN
                      MOVE "  "  TO M-KUBUN
                  ELSE  IF  MITUDO >= 150
                          THEN
                              MOVE "-"  TO M-KUBUN
                          ELSE
                              MOVE "--" TO M-KUBUN
                  END-IF
          END-IF
  END-IF
END-IF
```

必ずIF と END−IF が組になる

EVALUATE 文と同じ処理を IF 文で表現しようとすると，非常に複雑になる。EVALUATE 文の方がわかりやすいことが一目瞭然である。

●練習問題●

5　次の処理条件をコーディングした。空欄をうめなさい。

▼処理条件

職種コードは数字項目1桁，データ名はSYOKUSYU

残業手当は数字項目5桁，データ名はTEATE

残業時間は数字項目3桁，データ名はJIKAN

職種コード＝1　　残業手当＝残業時間＊500

職種コード＝2　　残業手当＝残業時間＊700

職種コード＝3　　残業手当＝残業時間＊1000

職種コードが1，2，3以外は　残業手当＝残業時間＊200

```
EVALUATE  (ア)
    WHEN 1   COMPUTE  TEATE = JIKAN * 500
    WHEN 2   COMPUTE  (イ)
    WHEN 3   (ウ)
    WHEN     (エ)
(オ)
```

6　次の条件をEVALUATE文で書きなさい。

(1)　A＞B　のとき，AAに1を加える。

A＜B　のとき，BBに1を加える。

A＝B　のとき，何もしない。

(2)

扶養コード	FUYO	英数字項目1桁
扶養手当	TEATE	数字項目5桁
子供	KAZU	数字項目

〔処理〕

扶養コード	扶養手当
0	TEATE＝0
1	TEATE＝10000
2	TEATE＝50000
3	TEATE＝50000＋KAZU＊5000

(3)

性別	SEIBETU	数字項目1桁
面接結果	RANK	英数字項目1桁
組分け	KUMI	数字項目1桁

〔処理〕

性別	面接結果	組分け
0	A	2
0	B	1
1	A	1
1	B	2

条件　　結果