# Section F : Statements

**DO**

คำสั่ง DO ใช้เป็นหัวเรื่องของ do-group หนึ่ง ๆ และยังใช้สำหรับบอกให้เครื่อง execute คำสั่งทั้งหมดภายในกลุ่มตามจำนวนครั้งที่ระบุไว้

**รูปแบบทั่วไป**

แบ่งออกเป็น 3 ชนิดดังนี้

ชนิดที่ 1 DO;

ชนิดที่ 2 DO { WHILE (element-expression) [UNTIL (element-expression)] / UNTIL (element-expression) [WHILE (element-expression)] };

ชนิดที่ 3 DO { pseudovariable / variable } = specification [,specification]...;

เมื่อ "specification" มีรูปแบบดังนี้

expression 1 [ TO expression 2 [BY expression 3] / BY expression 3 [TO expression 2] / REPEAT expression 6 ] [WHILE (expression 4)] [UNTIL (expression 5)];

เมื่อ WHILE และ UNTIL จะเรียงลำดับอันไหนก่อนหรือหลังก็ได้

**กฎเกณฑ์เกี่ยวกับ** syntax

1. คำสั่ง DO ทั้ง 3 ชนิดนี้ ต้องใช้คู่กับคำสั่ง END เพื่อบอกขอบเขตจำกัดของ do-group หนึ่ง ๆ คำสั่ง DO ชนิดแรกเท่านั้นที่เครื่องจะ execute คำสั่งทั้งหมดภายใน group เพียงครั้งเดียว

2. คำสั่ง DO ชนิดที่ 3, variable หรือ pseudovariable ต้องเป็น a single element อาจจะมี subscripted และ/หรือ qualified ก็ได้ (ถ้าใช้ optimizing compiler, pseudovariable ในที่นี้จะเป็น COMPLETION, COMPLEX, PRIORITY, STRING ไม่ได้)

**หมายเหตุ**

ถ้า "variable" ในที่นี้เป็น program control variable ใน specification ต้องไม่มี option BY และ option TO

3. expression แต่ละอันที่อยู่ใน specification ต้องเป็น an element expression

4. ถ้าใน specification หนึ่ง มีเฉพาะ "TO expression 2" แต่ไม่มี "BY expression 3" หมายความว่า ในที่นี้ expression 3 มีมูลค่าเท่ากับ 1

5. ถ้าใน specification หนึ่ง ๆ มีเฉพาะ "BY expression 3" แต่ไม่มี "TO expression 2" เครื่องจะ execute คำสั่งใน do–group นั้นซ้ำ ๆ กัน จนกระทั่งจบด้วย WHILE clause หรือ UNTIL clause หรือบางคำสั่งซึ่งจะส่ง control ออกไปนอก do–group

6. ถ้าใน specification นั้น มี "REPEAT expression 6" เครื่องจะ execute ซ้ำ ๆ กัน จนกระทั่งจบด้วย WHILE หรือ UNTIL clause หรือบางคำสั่ง ซึ่งจะส่ง control ออกไปนอก do–group

7. ถ้าใน specification นั้นไม่มี "TO expression 2" ไม่มี "BY expression 3" และไม่มี "REPEAT expression 6" หมายความว่า เครื่องจะ execute คำสั่งทั้งหมดใน do–group เพียงครั้งเดียว เฉพาะ control variable ที่มีมูลค่าเท่ากับ expression 1 ซึ่งในกรณีนี้ UNTIL clause เครื่องจะไม่สนใจ แต่ถ้ามี "WHILE expression 4" อยู่ด้วย เครื่องจะ execute ก็ต่อเมื่อ expression 4 เป็นจริงเท่านั้น

**กฎเกณฑ์ทั่วไป**

1. คำสั่ง DO ชนิดที่ 1, ใช้บอกขอบเขตจำกัดการเริ่มต้นของ do–group หนึ่ง ๆ เท่านั้น, เครื่องจะ execute คำสั่งทั้งหมดภายใน do–groupt เพียงครั้งเดียว

**ตัวอย่าง**

```
DO;
 TEMP  = A;
 A     = B;
 B     = TEMP;
END;
```

2. คำสั่ง DO ชนิดที่ 2, ใช้บอกขอบเขตจำกัด การเริ่มต้นของ do–group หนึ่ง ๆ และเครื่องจะ execute คำสั่งทั้งหมด ซ้ำ ๆ กัน ทั้งนี้ขึ้นอยู่กับ defined ข้างล่างนี้

```
LABEL : DO WHILE (expression 1) UNTIL (expression 2);
            Statement−1
              :
              :
            Statement−n
        END;
NEXT:statement/*STATEMENT FOLLOWING THE DO GROUP*/
```

คำสั่งทั้งหมดข้างต้นมีความหมายเหมือนกับคำสั่งข้างล่างนี้

```
LABEL : IF (expression 1) THEN; ELSE GO TO NEXT;
            Statement − 1
              :
              :
            Statement − n
LABEL 2 : IF (expression 2) THEN; ELSE GO TO LABEL;
NEXT : Statement/* STATEMENT FOLLOWING THE DO GROUP */
```

ถ้าไม่มี option UNTIL คำสั่ง IF ที่ label LABEL 2 ให้แทนดด้วย GO TO LABEL, ถ้าไม่มี option WHILE คำสั่ง IF ที่ label LABEL ให้แทนด้วย null statement

**หมายเหตุ**

ถ้าไม่มี optionn WHILE เครื่องจะ execute statement−1 ถึง statement−n อย่างน้อยที่สุดหนึ่งครั้ง

**ตัวอย่าง**

**โปรแกรมคำนวณหาตัวเลขที่มีมูลค่าน้อยที่สุดจากข้อมูลที่เป็นตัวเลขชุดหนึ่ง**

```
/* READ  BALANCE  & COMPUTE THE MINIMUM*/
MINBAL : PROCEDURE OPTIONS (MAIN);
        GET LIST (OLD_BALANCE, BALANCE);
        DO WHILE (BALANCE > = 0);
            IF BALANCE < OLD_BALANCE
              THEN OLD_BALANCE = BALANCE;
            GET LIST (BALANCE);
        END;
```

```
        PUT SKIP LIST ('MINIMUM BALANCE =', OLD_ BALANCE);
    END MINBAL;
```

**ลักษณะบัตรข้อมูล**

```
5  12  7  10  3  -9
```

เครื่องจะพิมพ์ผลลัพธ์ดังนี้

```
MINIMUM BALANCE = Δ 3.00000E+00
```

**ตัวอย่าง** top–down illustration

```
ABC : PROCEDURE OPTIONS (MAIN);
      DCL EOF BIT (1) INIT ('0'B);
      ON ENDFILE (SYSIN) EOF = '1'B;
      GET LIST (A,B,C);

      DO WHILE (¬EOF);
         X = A+B;
         Y = A–B;
         Z = A* B;
      PUT SKIP LIST (X,Y,Z);
      GET LIST (A,B,C);
      END;

      END ABC;
```

3. คำสั่ง DO ชนิดที่ 3, ใช้บอกขอบเขตจำกัดการเริ่มต้นของ do–group หนึ่ง และเป็นการควบคุมให้เครื่อง execute คำสั่งทั้งหมดใน do–group ซ้ำ ๆ กัน ถ้าใน specification มีทั้ง option–TO และ BY ดังนี้

```
LABEL : DO variable = expression 1
                      TO expression 2
                      BY expression 3
                      WHILE (expression 4)
                      UNTIL (expression 5);
                Statement – 1
                      :
                      :
                Statement – m
LABEL 1   : END;
NEXT      : statement
```

# ถ้า variable นั้นไม่ใช่ pseudovariable, คำสั่งทั้งหมดข้างต้นจะมีความหมายเหมือนกับข้างล่างนี้

```
LABEL : P = ADDR (variable);
        e1 = expression 1;
        e2 = expression 2;
        e3 = expression 3;
         v = e1;

LABEL 2 :IF (e3 > = 0) & (v > e2) |(e3 < 0) & (v < e2)
                    THEN GO TO NEXT;
                    IF (expression 4) THEN;
                    ELSE GO TO NEXT;
                Statement − 1
                        :
                        :
                Statement − m

LABEL 1 : IF (expression 5) THEN GO TO NEXT;
LABEL 3    : v = v + e3;
             GO TO LABEL 2;
NEXT       : statement
```

ถ้า specification นั้นมี option REPEAT อยู่ด้วยตามรูปแบบข้างล่างนี้

```
LABEL : DO variable = expression 1
                      REPEAT expression 6
                      WHILE (expression 4)
                      UNTIL (expression 5);
        statement − 1
                :
                :
        statement − m
LABEL 1    : END;
NEXT       : statement
```

ถ้าในที่นี้ variable ไม่ใช่ pseudovariable คำสั่งทั้งหมดข้างต้น จะมีความหมายเหมือนกับข้างล่างนี้

```
LABEL : p = ADDR (variable);
        el = expression 1;
         v = el;
LABEL 2 :
          IF (expression 4) THEN;
             ELSE GO TO NEXT;
             statement - 1


             statement - m
LABEL 1   : IF (expression 5) THEN GO TO NEXT;
LABEL 3   : v = expression 6;
            GO TO LABEL 2;
NEXT      : statement
```

**ตัวอย่าง**

โปรแกรมหาค่า factorial N (N!)

```
/*THIS PROGRAM COMPUTES FACTORIAL N FOR N = 0*/
FACTOR : PROCEDURE OPTIONS (MAIN);
         GET LIST (N);
         PUT SKIP LIST (N);
         IF N > = 0 THEN FACT = 1;
                    ELSE FACT = 0;
         DO K=NTO1BY - 1;
              FACT = FACT*K;
         END;
         PUT SKIP LIST (FACT);
END FACTOR;
```

4. WHILE clause ในคำสั่ง DO ชนิดที่ 2 และชนิดที่ 3 เป็นการกำหนดว่า ก่อนที่เครื่องจะ execute คำสั่ง ซ้ำกันในแต่ละครั้งนั้น เครื่องจะประเมินผล element expression และถ้าจำเป็นก็อาจจะเปลี่ยนรูปให้เป็น bit string ถ้าบิทใด ๆ ใน string ผลลัพธ์มีค่าเท่ากับ 1 คำสั่งทั้งหมดใน do-group จะถูก execute สำหรับคำสั่ง DO ชนิดที่ 2 ถ้า บิท ทั้งหมดมีค่าเท่ากับ 0 เครื่องมันจะหยุด execute คำสั่งใน do-group ในขณะที่คำสั่ง DO ชนิดที่ 3 การหยุด execute จะขึ้นอยู่กับ specification ที่มี WHILE clause เท่านั้น มิฉะนั้นแล้วมันจะ execute ซ้ำอีก กับ specification ชุดถัดไป

**ตัวอย่าง**

```
DO N = 1 BY 1 WHILE (TERM > 0.0001);
```

ตัวอย่างที่ตราบใดที่ TERM ยังมีค่ามากกว่า 0.0001 เครื่องจะ execute คำสั่งใน do–group ซ้ำ ๆ กันและทุกครั้ง N จะมีค่าเพิ่มขึ้นทีละ 1 เสมอ เมื่อใดที่ TERM มีมูลค่าน้อยกว่าหรือเท่ากับ 0.0001 เครื่องจะหยุด execute do–group แล้วส่ง control ออกไปนอก loop

5. UNTIL clause ที่อยู่ในคำสั่ง DO ชนิดที่ 2 และชนิดที่ 3 เป็นตัวกำหนดว่า**หลังจาก**ที่มีการ execute คำสั่งซ้ำกันในแต่ละครั้งนั้น เครื่องจะประเมินผล element expression และถ้าจำเป็นก็อาจจะเปลี่ยนรูปให้เป็น bit string ถ้าทุกบิทใน string ผลลัพธ์มีค่าเป็น 0 คำสั่งใน do–group จะถูก execute แต่ถ้ามีบิทใดก็ตามเป็น 1 สำหรับคำสั่ง DO ชนิดที่ 2 เครื่องจะหยุด execute คำสั่งใน do–group ในขณะที่คำสั่ง DO ชนิดที่ 3 การหยุด execute จะขึ้นอยู่กับ "specification" ที่มี UNTIL clause เท่านั้นแล้ว เครื่องจะทำการ execute ซ้ำกับ specification ชุดถัดไป

6. ใน "specification" หนึ่ง ๆ ซึ่งมีทั้ง option TO และ BY expression 1 หมายถึง มูลค่าแรกของ control variable (ซึ่งอาจจะเป็น variable หรือ pseudovariable ก็ได้)
expression 3 หมายถึง มูลค่าเพิ่มที่จะเอาไปบวกกับมูลค่าของ control variable ภายหลังจากที่มีการ execute คำสั่งทั้งหมดใน group
expression 2 หมายถึง มูลค่าสุดท้ายของ control variable

การหยุด execute คำสั่งใน do–group สำหรับ "specification" หนึ่ง ๆ นั้น เกิดขึ้นเมื่อมูลค่าของตัวแปรควบคุม ซึ่งได้รับการตรวจสอบ เมื่อจบการ execute คำสั่งใน loop มีมูลค่าไม่ได้อยู่ในขอบเขตจำกัด (range) ซึ่งกำหนดโดย expression 1 และ expression 2 เมื่อเครื่องหยุด execute สำหรับ specification สุดท้ายแล้ว มันจะส่ง control ให้ไป execute คำสั่งซึ่งอยู่ถัดจาก do–group ต่อไป

7. ใน specification ที่มี option REPEAT มูลค่าแรกของตัวแปรควบคุม จะมีค่าเท่ากับ expression 1 หลังจากที่ execute คำสั่งทั้งหมดใน group แล้ว เครื่องจะประเมินผล expression 6 แล้วกำหนดให้เป็นมูลค่าของตัวแปรควบคุมตัวถัดไป

8. การย้าย control จากภายนอก do–group เข้าไปใน do–group หนึ่ง จะกระทำได้ก็ต่อเมื่อใน do–group นั้นถูกกำหนดขอบเขตด้วย คำสั่ง DO ชนิดที่ 1 นั่นหมายความว่า จะไม่มีการ execute คำสั่งซ้ำ และใน do–group ที่มีการ execute ซ้ำ ๆ กันนั้นต้องไม่มีคำสั่ง ENTRY

9. generation ของตัวแปรควบคุมแต่ละตัว อาจจะเป็น pointer–qualified หรือ controlled ซึ่งสร้างมาจากนอก loop ก็ได้ ก่อนที่มูลค่าแรกของ expression (expression 1) จะได้รับการประเมินผล ถ้าภายใน loop control variable generation เปลี่ยนไปโดยอาจจะเป็นการเปลี่ยน pointer หรือ

โดยการ allocating ก็ตาม, loop นั้นก็ยังคงดำเนินต่อไปด้วย control variable ที่ได้จาก generation ก่อนนั้น อย่างไรก็ตาม reference ใด ๆ ดัง control variable ที่อยู่ใน loop เป็น reference generation ต่อไป และถือว่าเป็นข้อผิดพลาด (error) ในการให้อิสระแก่ generation

10. สำหรับ optimizing compiler อาจจะให้มี loop ซ้อนกันได้มากที่สุด 49 ชั้น แต่ถ้าเป็น checkout compiler ไม่มีข้อจำกัด

## แบบฝึกหัด

1. จงบอกที่ผิดในคำสั่ง DO ต่อไปนี้

   a) DO K = 1 TO 15,

   b) DO K = 31 TO 1 BY − 1.

   c) DO WHILE K = 0;

2. ถามว่า loop ซึ่งกำหนดด้วยคำสั่ง DO ข้างล่างนี้จะได้รับการ execute กี่ครั้ง

   a) DO I = 1 TO 50;

   b) DO J = 5 TO 80 BY 5;

   c) DO K = L TO M; เมื่อ L = 10 และ M = 6

   d) DO L = 10 TO 15 BY 9;

   e) DO M = 3 TO −4 BY −1;

3. loop ที่กำหนดในโปรแกรม segment ข้างล่างนี้จะได้รับการ execute กี่ครั้ง และแต่ละครั้งตัวแปรควบคุม จะมีมูลค่าเท่ากับเท่าไร?
   (กำหนดให้ J = 2. K = 3. L = 3 และ M = 5)

   a) DO I = 1 TO M + 4;

   b) DO I = 10 TO − 5 BY L;

   c) DO I = j − I TO K*M BY K + 2;

   d) DO I = KTOL;

   e) DO I = I BY I WHILE (M>O);
      M = M − 1; PUT LIST (M);
      END;

## END

คำสั่ง END ใช้สำหรับปิด, หรือจำกัดขอบเขต blocks, do–groups และ select–groups รูปแบบทั่วไป

```
END        [identifier];
```

### กฎเกณฑ์เกี่ยวกับsyntax

ในที่นี้ identifier เป็น a label หรือ entry constant
และมี subscripted ไม่ได้

### กฎเกณฑ์ทั่วไป

1. ถ้ามี label ตามหลัง END ใช้เป็นคำสั่งปิด unclosed do-group select-group หรือ block ซึ่งคำสั่งหัวเรื่องเป็นคำสั่ง DO. SELECT, BEGIN หรือ PROCEDURE ซึ่งมี label นั้น และยังใช้เป็นคำสั่งปิด unclosed do-groups, select groups หรือ blocks ภายใน group หรือ block อื่น ๆ ได้อีกด้วย

2. ถ้าไม่มี label ตามหลัง END ใช้เป็นคำสั่งปิด do-group select-group หรือ บล๊อก ที่มีคำสั่งหัวเรื่องเป็นคำสั่ง DO, SELECT, BEGIN หรือ PROCEDURE ซึ่งไม่ได้ corresponding กับคำสั่ง END

3. เมื่อ control มาถึงคำสั่ง END ของ PROCEDURE หนึ่ง ๆ มันจะปฏิบัติเหมือนกับว่าเป็นคำสั่ง RETURN

### ตัวอย่าง

```
RIVER : PROCEDURE OPTIONS (MAIN);



END RIVER;
```

### ตัวอย่าง

```
DO MIN = 1 TO 60;
    GET LIST (NO-CUST);
    PUT LIST (MIN, NO-CUST);
'END;
```

ตัวอย่าง

```
IF X > Y THEN DO;
                A = A+B;
                B = B+C;
                C = C+D;
                END;
X = A+B+C;
```

## แบบฝึกหัด

จากโปรแกรมข้างล่างนี้ จงแสดงมูลค่าของ ตัวแปร I, J และ K ที่เครื่องคอมพิวเตอร์จะพิมพ์ออกมาให้

```
EXAM : PROOCEDURE OPTIONS (MAIN);
       DCL (I,J,K) FIXED DECIMAL (3, 1);
       DO I = 5 TO 10 BY 5;
          DO J = 1 TO 3;
             DO K = O TO .5 BY .2;
                PUT LIST (I,J,K);
                END;
             END;
          END;
       END EXAM;
```

IF

คำสั่ง IF ใช้ตรวจสอบมูลค่าของ expression ซึ่งกำหนดมาให้ และควบคุมทิศทางการ execute ทั้งนี้ขึ้นอยู่กับผลลัพธ์ของการตรวจสอบนั้น

**รูปแบบทั่วไป**

| IF element−expression THEN unit−1 [ELSE unit−2] |
|---|

**กฎเกณฑ์ของ** syntaxs

1. แต่ละ unit อาจจะเป็น a single statement อะไรก็ได้ (ยกเว้น DO, SELECT, END, PROCEDURE, BEGIN, DECLARE, DEFAULT, FORMAT, หรือ ENTRY) หรือ a do−group, a select−group หรือ a begin block ก็ได้

2. คำสั่ง IF โดยตัวมันเองจะไม่จบด้วยเครื่องหมาย semi colon (;) อย่างไรก็ตามแต่ละ unit ที่กำหนดให้ ต้องจบด้วยเครื่องหมาย ; หนึ่งตัว

3. แต่ละ unit อาจจะมี label และอาจจะมีเงื่อนไข prefixes ก็ได้

**กฎเกณฑ์ทั่วไป**

1. element expression ซึ่งได้รับการประเมินผล ถ้าจำเป็นจะต้องถูกเปลี่ยนรูปให้เป็น bit string ถ้ามี ELSE clause (ส่วนที่เป็น ELSE และตามด้วย unit) เมื่อเกิดเหตุการณ์ต่อไปนี้

ถ้ามีบิทใดบิทหนึ่งใน string เป็น 1 หรือมีมูลค่าเป็นจริง เครื่องจะ execute unit−1 แล้วส่ง control ไปยังคำสั่งที่อยู่ถัดจากคำสั่ง IF

ถ้าบิททุกตัวใน string นั้นมีค่าเป็น 0 หรือมีมูลค่าเป็นเท็จ เครื่องจะไม่ execute unit−1 แต่จะ execute unit−2 หลังจากนั้นจะส่ง control ไปยังคำสั่งถัดไป

**ถ้าไม่มีส่วนที่เป็น** ELSE clause เมื่อเกิดเหตุการณ์ต่อไปนี้

ถ้ามีบิทใดบิทหนึ่งใน string เป็น 1 เครื่องจะ execute unit−1 และส่ง control ไปยังคำสั่งที่อยู่ถัดจากคำสั่ง IF, ถ้าบิททุกตัวเป็น 0 เครื่องจะไม่ execute unit−1 และส่ง control ไปยังคำสั่งถัดไป

แต่ละ unit อาจจะประกอบด้วยคำสั่งซึ่งใช้สำหรับเคลื่อนย้าย control ก็ได้ (เช่นคำสั่ง GO TO) ในกรณีนี้ลำดับการ execute ปกติของคำสั่ง IF อาจจะใช้ไม่ได้

**ตัวอย่าง**

```
IF A = 0 THEN GO TO NEXT_PARAGRAPH; ELSE C = 1;
        :
        :
```

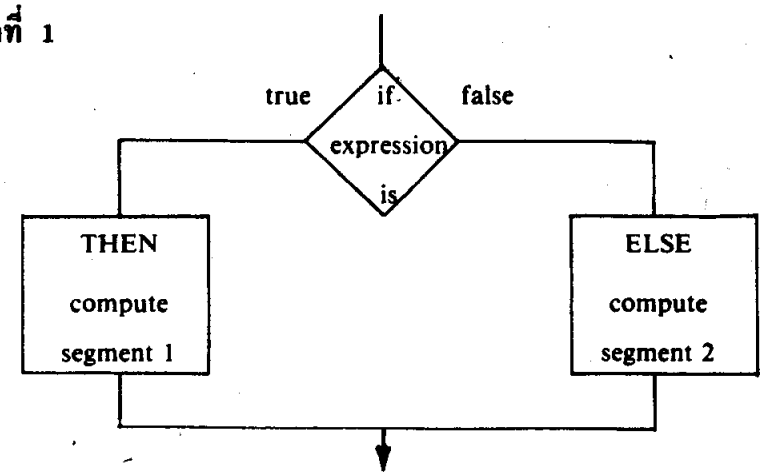
NEXT_PARAGRAPH :

:

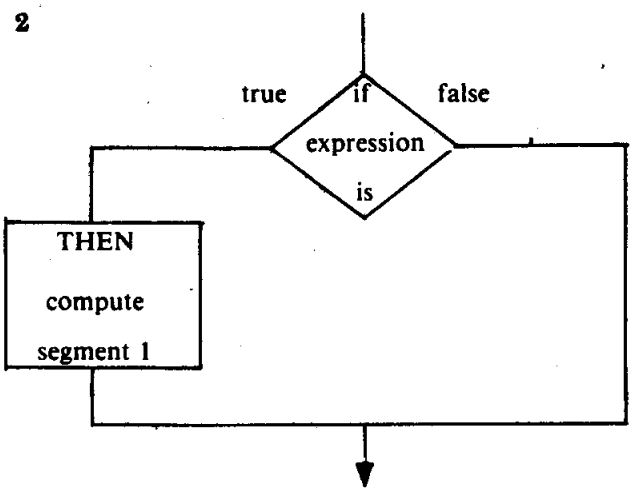
An array หรือ structure variable อาจจะปรากฏใน element expression เฉพาะในรูปของ an argument ของฟังก์ชั่น ซึ่งให้มูลค่าของอีลิเม้นท์หนึ่งจำนวน

ตามกฎเกณฑ์ข้อนี้เราเขียนเป็น flowchart ได้สองรูปดังนี้

**ชนิดที่ 1**




**ชนิดที่ 2**

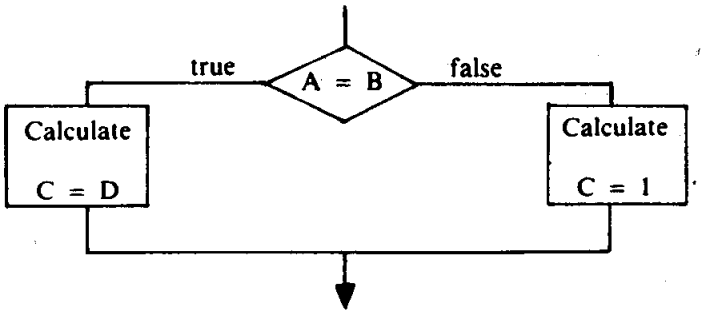


**ตัวอย่าง**

IF A = B THEN C = D; ELSE C = 1;

คำสั่งข้างต้นมีความหมายดังนี้ ถ้ามูลค่าของตัวแปร A เท่ากับมูลค่าของตัวแปร B ให้ทำคำสั่ง C = D แล้วทำส่วนที่เหลือต่อไปโดยไม่ต้องทำส่วนที่เป็น ELSE แต่ถ้ามูลค่าของ A ไม่เท่ากับ B เครื่องจะข้ามส่วนที่เป็น THEN แล้วทำส่วนที่เป็น ELSE คือกำหนดให้ตัวแปร C มีค่าเท่ากับ 1

เขียน flowchart ดังนี้




**ตัวอย่าง**

```
IF CONSUMPTION < = 400
 THEN AMOUNT = CONSUMPTION*0.0039;
 ELSE AMOUNT =400*0.0039 + (CONSUMPTION−400)*0.0025;
```

**ตัวอย่าง**

จงเขียน flowchart จากโปรแกรม ข้างล่างนี้

```
RIVER : PROCEDURE OPTIONS (MAIN);
MORE : GET LIST (LENGTH);/* INPUT A RIVER LENGTH*/
       IF LENGTH > 1200 THEN PUT SKIP LIST (LENGTH);
       GO TO MORE;
       END RIVER;
```

## Compound comparisons

**ตัวอย่าง**

```
R = HEIGHT/WEIGHT;
IF (R > = 2.7) & (R < = 3.0)
   THEN GOODRATIO = GOODRATIO + 1;
```

แต่เขียน expression 2.7 < = R < = 3.0 ใช้ไม่ได้

**ตัวอย่าง**

```
GET LIST (TELNO, PREVIOUS, CURRENT);
IF (CURRENT = PREVIOUS) | (CURRENT−PREVIOUS > 1000)
THEN PUT SKIP LIST ('SERVICE FOR',TELNO);
```

จากตัวอย่างทั้งสองข้างต้น เครื่องหมายวงเล็บที่ล้อมรอบ comparisons โปรแกรมเมอร์จะไม่เขียนก็ได้ เพราะว่า logical and (&) และ logical or (|) operation มีลำดับตามสำคัญ (priority) น้อยกว่า comparison operation เครื่องจะเลือกทำ comparison ก่อนเสมอ

2. คำสั่ง IF อาจจะมีซ้อนกัน (nested) นั่นคือ "unit" ใด unit หนึ่ง หรือทั้ง 2 units โดยตัวมันเองอาจจะเป็นคำสั่ง IF ก็ได้ เนื่องจาก ELSE clause แต่ละชุด ต้องใช้คู่กับคำสั่ง IF ในชุดที่อยู่ใน block หรือ do−group เดียวกัน ELSE ที่ตามด้วย null statement อาจจะใช้สำหรับกำหนดลำดับที่ของ control สำหรับ optimizing compiler จำนวน IF ที่ซ้อนกัน กำหนดได้มากที่สุด 49 ชั้น ถ้าเป็น checkout compiler ไม่มีข้อจำกัดในเรื่องนี้

**ใน nested's IF เขียนเป็น format ง่าย ๆ ได้ดังนี้**

```
IF (expression 1) THEN statement − 1; ELSE
IF (expression 2) THEN statement −2; ELSE statement − 3;
   next statement;
```

**ขั้นตอนการทำงาน**

1. ถ้า expression 1 เป็นจริงให้คำนวณ statement−1 แล้วตามด้วย next statement

2. ถ้า expression 1 ไม่เป็นจริงให้ตรวจสอบ expression 2 ถ้า expression 2 เป็นจริงให้คำนวณ statement-2 แล้วทำ next statement

3. ถ้า expression 1 ไม่จริง และ expression 2 ไม่จริงด้วย ให้คำนวณ statement–3 แล้วทำ next statement

สำหรับรูปแบบทั่วไปเขียนดังนี้

```
IF (expression 1) THEN statement–1;ELSE
IF (expression 2) THEN statement–2;ELSE
                  :
                  :
IF (expression n) THEN statement–n; ELSE statement–m;
   next statement
```

**SELECT**

คำสั่งนี้ใช้เป็นหัวเรื่องใน select–group หนึ่ง ๆ โดยมีรูปแบบทั่วไป ดังนี้

```
SELECT [( expression–1 )];
[[(condition–prefix [, condition–prefix ]...): ]
WHEN (expression–2 [, expression–2 ]...) unit ]...
[{ OTHERWISE|OTHER }                    unit ]
END [ identifier ];
```

## กฎเกณฑ์ของ syntax

1. แต่ละ 'unit' อาจจะเป็น a single statement ก็ได้ (ยกเว้น DO,SELECT, END,PROCEDURE,BEGIN,DECLARE,DEFAULT,FORMAT, หรือ ENTRY) หรือ a do–group, a select–group, หรือ a begin block ก็ได้

2. แต่ละ unit อาจจะมี label และอาจจะมีเงื่อนไข prefix ก็ได้

3. ใน select–group หนึ่ง ๆ ต้องจบด้วยคำสั่ง END

## กฎเกณฑ์ทั่วไป

1. expression ในคำสั่ง SELECT เมื่อประเมินผลแล้วเครื่องจะเก็บมูลค่านั้นเอาไว้ ส่วน expressions ใน WHEN clause เมื่อประเมินผลแล้วผลลัพธ์ที่ได้ก็จะถูกนำไปเปรียบเทียบกับมูลค่าที่เก็บไว้ ถ้าปรากฏว่า expression ชุดใดชุดหนึ่งเท่ากับมูลค่าที่เก็บไว้ เครื่องก็จะหยุดประเมินผล expressions ใน WHEN clause แล้ว execute 'unit' ตามหลัง WHEN clause นั้น ถ้าไม่มี expression ใด ๆ ใน WHEN clause มีมูลค่าเท่ากับมูลค่าที่เก็บไว้ เครื่องจะข้ามไป execute unit ซึ่งตามหลัง OTHERWISE clause แทน

2. ถ้าไม่มี expression–1, expression–2, แต่ละชุดจะถูกประเมินผล ถ้าจำเป็น จะเปลี่ยนรูปเป็น bit string ถ้าบิทใดบิทหนึ่งใน string นั้นเป็น '1'B เครื่องจะ execute 'unit' ที่ตามหลัง WHEN clause แต่ถ้าไม่มีบิทใดเป็น '1'B เครื่องจะข้าม unit หลัง WHEN clause และไป execute unit หลัง OTHERWISE clause แทน

3. หลังจากที่เครื่อง execute "unit" หลัง WHEN หรือ OTHERWISE clause แล้ว control จะถูกส่งไปยังคำสั่ง executable แรกที่อยู่ถัดจากไป select–group ยกเว้นไว้แต่ในกรณีที่ทิศทางของ control เปลี่ยนไปภายใน 'unit'

4. ถ้ามี expression–1, expression–2, แต่ละชุดจะถูกนำมาเปรียบเทียบดังนี้ ((expression–1) = (expression–2))

แล้วให้ผลลัพธ์เป็น a scalar bit value

5. ถ้าไม่มี OTHERWISE clause การ execute ใน select–group จะไม่เกิดผลลัพธ์ในการเลือก "unit" จึงเกิดเงื่อนไข error ขึ้น

6. เงื่อนไข prefix หนึ่ง ๆ ในคำสั่ง SELECT ใช้เฉพาะในการประเมินผลของ expression ในคำสั่ง SELECT เท่านั้น ไม่ได้ใช้กับคำสั่งอื่น ๆ ใน group, เงื่อนไข prefix จะมีผลกระทบต่อ WHEN clause ก็เฉพาะในการประเมินผล expression ใน WHEN clause เท่านั้น ไม่ใช่กับ 'unit' ที่ตามหลัง WHEN clause

7. ถ้าเป็น optimizing compiler จำนวน select–groups สามารถกำหนดให้ซ้อนกันได้มากถึง 49 ชุด แต่ถ้าเป็น checkout compiler ไม่มีข้อจำกัด

## STOP

คำสั่งนี้เครื่องจะหยุดทันทีใน major task และ subtasks ทั้งหมด โดยมีรูปแบบทั่วไปดังนี้

```
STOP;
```

### กฎเกณฑ์ทั่วไป

การหยุด execute ใด ๆ ก่อนหน้าคำสั่งนี้ ในงานชิ้นหนึ่ง เมื่อเครื่อง execute คำสั่ง STOP จะเกิดเงื่อนไข FINISH ขึ้น แต่ในการ return ตามปกติ เครื่องจะหยุดงานทั้งหมดในโปรแกรมนั้น

### ตัวอย่างโปรแกรม

1. หน่วยงานสถิติของโรงเรียนประถมศึกษาแห่งหนึ่ง ได้บันทึกความสูงของนักเรียนทั้งหมดที่มีอายุ 8 ปี ได้ตัวเลขตามตารางข้างล่างนี้

| ความสูง (เซ็นติเมตร) | จำนวนนักเรียน (คน) |
|---|---|
| 130 | 5 |
| 131 | 4 |
| 132 | 11 |
| 133 | 20 |
| 134 | 18 |
| 135 | 17 |
| 136 | 10 |
| 137 | 0 |
| 138 | 8 |
| 139 | 7 |
| 140 | 1 |

จงเขียน flowchart และโปรแกรม เพื่อให้เครื่องคอมพิวเตอร์อ่านข้อมูลข้างต้น จากบัตรแล้วพิมพ์แผนภูมิแท่ง (Histogram) ออกมาดังนี้

```
*****
****
***********
********************
******************
**********
********
*******
*
```

SOURCE LISTING

```
STMT LEV NT

                    /*PROGRAM WRITE HISTOGRAM OF THE LENGTH*/

1    0 GRAPH: PROC OPTIONS(MAIN);
2  1 0        DCL SYSIN INPUT, SYSPRINT OUTPUT;
3  1 0        DCL D FIXED DEC(3),B FIXED DEC(2),A CHAR(25) INIT((25)'*');
4  1 0        DCL C CHAR (25);
5  1 0        DCL SUBSTR BUILTING;
6  1 0        PUT PAGE EDIT ('LENGHT (C.M.)','TOTAL OF PERSONS')
                                (A,X(10),A);
7  1 0        GET LIST (D,B);  /*GET FIRST CARD*/
8  1 0        DO WHILE (D¬= 000);
9  1 1        C = SUBSTR (A,1,B);
10 1 1        PUT EDIT (D,C) (SKIP(2),X(5),F(3),X(15),A);
11 1 1        GET LIST (D,B);  /*GET NEXT CARD*/
12 1 1        END;
              /*D = 000 AT THIS POINT*/
13 1 0        PUT LIST ('* = A PERSON');
14 1 0        END GRAPH;
```

| LENGHT (C.M) | TOTAL OF PERSONS | |
|---|---|---|
| 130 | ***** | |
| 131 | **** | |
| 132 | ********** | |
| 133 | ********** | |
| 134 | ********** | |
| 135 | ********** | |
| 136 | ********** | |
| 137 | | |
| 138 | ******** | |
| 139 | ******* | |
| 140 | * | * = A PERSON |

2. จงเขียน flowchart และ โปรแกรมให้ออกเสียงตัวเลข 1 จำนวนโดยการอ่านเลขจำนวนเต็มหนึ่งตัว เข้าไปในเครื่อง (เลขจำนวนเต็มอะไรก็ได้ระหว่าง 1 ถึง 999) แล้วออกเสียงเป็นภาษาอังกฤษ

**ตัวอย่างเช่น**

อ่านเลข 283 ออกเสียงเป็น TWO HUNDREDS AND EIGHTY THREE

SOURCE LISTING

```
STMT LEV NT
 1    0 ZOMBY:   PROC OPTIONS (MAIN);
 2    1    0     DCL NUM PIC '999',
                      SP (CHAR (80) VAR,
                       1 X DEF NUM,
                         2 H PIC '9',
                         2 T PIC '9',
                         2 U PIC '9',
                      A(0:9) CHAR(5) VAR INIT ('','ONE','TWO','THREE','FOUR',
                                               'FIVE','SIX','SEVEN','EIGHT'
                                               'NINE'),
                      B(0:9) CHAR(9) VAR INIT ('TEN','ELEVEN','TWELVE',
                                              'THIRTEEN','FOURTEEN',
                                               'FIFTEEN',
                                              'SIXTEEN','SEVENTEEN',
                                              'EIGHTEEN','NINETEEN'),
                      C(0:9) CHAR(7) VAR INIT ('','','TWENTY','THIRTY',
                                              'FOURTY','FIFTY,'SIXTY',
                                              'SEVENTY','EIGHTY',
                                              'NINETY'),
                      SYSIN INPUT,SYSPRINT OUTPUT;
 3    1    0        ON ENDFILE(SYSIN) STOP;
 4    1    0 JUMP:  GET EDIT(NUM)(F(3));
 5    1    0        SELECT;
```

```
 6  1  1          WHEN(H=0&T=0&U=0)      SP = 'ZERO';
 7  1  1          WHEN(H=0&T=0&U¬=0)     SP = A(U);
 8  1  1          WHEN(H&0&T>=0&T= 1)    SP = B(U);
 9  1  1          WHEN(H=0&T>=2&U=0)     SP = C(T);
10  1  1          WHEN(H=0&T>=2&U¬=0)    SP=C(T)||'-'||A(U);
11  1  1          WHEN(H¬=0&T=0&U=0)     SP = A(H)||'HUNDRED';
12  1  1          WHEN(H¬=0&T=0&U¬=0)    SP = A(H)|| 'HUNDRED AND'
                                              ||A(U);
13  1  1          WHEN(H¬=0&T¬=0&T= 1)   SP = A(H)|| 'HUNDRED AND'
                                              ||B(U);
14  1  1          WHEN(H¬=0&T>=2&U=0)    SP = A(H)||'HUNDRED AND'
                                              ||C(T);
15  1  1          WHEN(H¬=0&T>=2&U¬=0)   SP=A(H)||'HUNDRED AND'
                                              ||C(T)||'-'||A(U);
16  1  1          OTHERWISE              SP = 'NOT NUMERICAL SET';

17  1  1       END;
18  1  0       PUT SKIP(3) EDIT('THE SPELLING OF ',NUM,' IS ● ',SP,' ● )
                          (COL(30),A,F(3),A,A,A);
19  1  0       GO TO JUMP;
20  1  0 END ZOMBY;
```

```
THE SPELLING OF   0 IS  *ZERO*
THE SPELLING OF   5 IS *FIVE*
THE SPELLING OF  10 IS *TEN*
THE SPELLING OF  12 IS *TWELVE*
THE SPELLING OF  18 IS *EIGHTEEN*
THE SPELLING OF  20 IS *TWENTY*
THE SPELLING OF  76 IS *SEVENTY-SIX*
THE SPELLING OF 105 IS *ONE HUNDRED AND FIVE*
THE SPELLING OF 143 IS *ONE HUNDRED AND FOURTY-THREE*
THE SPELLING OF 949 IS *NINE HUNDRED AND FOURTY-NINE*
THE SPELLING OF 888 IS *EIGHT HUNDRED AND EIGHTY-EIGHT*
THE SPELLING OF 987 IS *NINE HUNDRED AND EIGHTY-SEVEN*
THE SPELLING OF 453 IS *FOUR HUNDRED AND FIFTY-THREE*
THE SPELLING OF   0 IS *ZERO*
THE SPELLING OF   0 IS *ZERO*
THE SPELLING OF   0 IS *ZERO*
THE SPELLING OF   0 IS *ZERO*
THE SPELLING OF   0 IS *ZERO*
THE SPELLING OF   0 IS *ZERO*
THE SPELLING OF   0 IS *ZERO*
```

## แบบฝึกหัด

1. คำสั่ง IF ข้างล่างนี้ถูกต้องหรือไม่? ถ้ามีที่ผิดจงแก้ไขให้ถูกต้อง

   a) IF A = B THEN GO TO SOUTH;

   b) IF A = B THEN C = D ELSE C = E;

   c) IF WEATHER = CLOUDLY THEN PUT LIST ('COOL DAY EXPECTED');

   d) IF (EYES = BLUE & (HEIGHT>70) THEN N = N+1, ELSE M = M + 1;

2. จงเขียน flowchart และโปรแกรม segment คำนวณหาค่า y ตามหลักเกณฑ์ดังนี้

   ถ้า $x \leq 1$, $y = ax + b$

   ถ้า $1 < x \leq 2$, $y = cx + d$

   ถ้า $2 < x \leq 3$, $y = ex + f$

   และถ้า $x > 3$, $y = gx + H$

3. จากโปรแกรม segment ข้างล่างนี้ จงเขียนเสียใหม่โดยไม่ใช้ nested IF'S แต่มูลค่าที่กำหนดให้ X ต้องเหมือนเดิม

```
IF A>0 THEN IF B>0 THEN IF C > 0 THEN X = 4;
                    ELSE X = 5;
               ELSE X = 1;
          ELSE X = -1;
```

4. กำหนดให้มูลค่าของตัวแปร C ขึ้นอยู่กับเครื่องหมายของตัวแปร A และตัวแปร B ดังที่แสดงไว้ในตารางข้างล่างนี้ จงเขียนส่วนหนึ่งของโปรแกรมกำหนดมูลค่าของ C ให้ถูกต้องสำหรับทุก combination ของมูลค่า A และ B

| | A = 0 | A > 0 | A < 0 |
|---|---|---|---|
| B = 0 | C = 1 | C = 2 | C = 3 |
| B>0 | C = 2 | C = 1 | C = 4 |
| B<0 | C = 3 | C = 4 | C = 1 |

5. จงเขียน flowchart และคำสั่ง IF ตรวจสอบมูลค่าของ A ดังนี้

ถ้า A มีค่าน้อยกว่าหรือ

เท่ากับ 0 ให้คำนวณ x = 2A, y = A + X

ถ้า A มีค่ามากกว่า 0 ให้คำนวณ X = 3A, Y = B + X

หลังจากตรวจสอบทั้งสองกรณีแล้ว ให้คำนวณ Z = X + Y

6. กำหนดให้ A, B, C และ D เป็นตัวแปร ส่วน $S_1$, $S_2$, $S_3$ และ $S_4$ เป็นคำสั่งอย่างใดอย่างหนึ่ง

a) จงเขียน flowchart ของคำสั่ง nested IF ข้างล่างนี้

IF A > B THEN IF B ¬> C THEN IF C ¬= D THEN S1; ELSE S2; ELSE S3; ELSE S4;

b) จงเปลี่ยนคำสั่ง nested IF ในข้อ a) ให้เป็น simple IF หลาย ๆ คำสั่งโดยใช้ compound conditions

7. คำสั่งข้างล่างนี้ตรวจสอบความสัมพันธ์ทางด้านขนาดของเลขจำนวนเต็ม x, y และ z สมมติว่าทั้ง 3 ค่านี้ไม่เท่ากันเลย

```
IF X < Z THEN IF X < Y THEN IF Y < Z THEN C = 1;ELSE C = 2; ELSE
IF Y < Z THEN C = 3;ELSE C = 4;ELSE IF X < Y THEN
IF X < Z THEN C = 5;ELSE C = 6;ELSE IF Y < Z THEN C = 7; ELSE
IF Z < X THEN IF Z < Y THEN C = 8;ELSE C = 9;ELSE C = 10;
```

a) จงเขียน flowchart และคำสั่งนี้ใหม่ให้อยู่ในรูปแบบที่อ่านง่ายขึ้น

b) เนื่องจากลำดับที่เป็นไปได้มีเพียง 6 อย่างเท่านั้น สำหรับการตรวจสอบเลข 3 ค่า จงหาว่าผลลัพธ์อีก 4 อย่างที่ไม่เกิดขึ้นแน่นอนมีอะไรบ้าง ให้เอาออก (**คำตอบ** C = 4, 5, 9, 10)

c) จงเขียนคำสั่งที่สั้นและง่ายกว่า เพื่อให้ทำงานแล้วได้ผลลัพธ์อย่างเดียวกันนี้

d) จงเขียน flowchart ของข้อ c)

8. นางสาววันเพ็ญ เป็นโปรแกรมเมอร์ฝึกงานของบริษัทบางกอกคอมพิวเตอร์ จำกัด เธอมีความวิตกกังวลเกี่ยวกับงานเขียนโปรแกรมมาก โปรแกรมแรกที่เธอเขียนนี้มีข้อผิดพลาด 1 แห่ง, โปรแกรมที่สองมีผิดพลาด 2 แห่ง, โปรแกรมที่สามมีผิดพลาด 4 แห่ง และเป็นเช่นนี้ตลอดมา นั่นคือในการเขียนโปรแกรมครั้งต่อ ๆ มาเธอจะทำผิดพลาดเพิ่มขึ้นเป็น 2 เท่าของครั้งก่อนหน้านี้ ในช่วงเวลาฝึกงาน 13 สัปดาห์ ในแต่ละสัปดาห์เธอจะมีงานเขียนโปรแกรม 2 ชิ้น

จงเขียนโปรแกรมคำนวณว่า งานเขียนโปรแกรมชิ้นสุดท้ายของนางสาววันเพ็ญจะมีที่ผิดกี่แห่ง (**คำตอบ** $2^{26-1}$)

9. บริษัทไทยโทรทัศน์ จำกัด ได้กำหนดหลักเกณฑ์การพิจารณาเงินโบนัสประจำปีของพนักงานทุกคนไว้ 2 อย่างคือ จำนวนชั่วโมงทำงานล่วงเวลาและจำนวนชั่วโมงที่พนักงานคนนั้นลางาน ข้อมูลของพนักงานแต่ละคนบันทึกในบัตร 80 คอลัมน์ มีหมายเลขของพนักงาน, ชื่อ นามสกุล, จำนวนชั่วโมงทำงานล่วงเวลา และจำนวนชั่วโมงที่ลางาน ตารางคำนวณเงินโบนัสมีดังนี้

| จำนวนล่วงเวลา $-\frac{2}{3}$* จำนวนที่ลางาน | โบนัส (บาท) |
|---|---|
| > 40 ชั่วโมง | 1,000 |
| > 30 แต่ ⩽ 40 ชั่วโมง | 800 |
| > 20 แต่ ⩽ 30 ชั่วโมง | 600 |
| > 10 แต่ ⩽ 20 ชั่วโมง | 400 |
| < 10 ชั่วโมง | 200 |

ให้นักศึกษากำหนดรูปแบบของ input และ output เอง จากนั้นจงเขียน flowchart และโปรแกรมอ่านบัตรข้อมูลของพนักงานในบริษัทนี้แล้วคำนวณและพิมพ์เงินโบนัสประจำปีของพนักงานแต่ละคน

10. บริษัทไฟฟ้านครหลวง จำกัด คิดค่ากระแสไฟฟ้าจากลูกค้า 3 อัตรา โดยขึ้นอยู่กับปริมาณไฟฟ้าที่ใช้ดังนี้

90 กิโลวัตต์แรก (Kwh)เสียเงินในอัตรา 11.74 cents per Kwh
450 กิโลวัตต์ถัดไป เสียเงินในอัตรา 3.74 cents per Kwh
และส่วนที่เกินจากนี้ เสียเงินในอัตรา 3.04 cents per Kwh

จงเขียน flowchart และ โปรแกรม segment อ่านข้อมูลซึ่งเป็นปริมาณไฟฟ้าที่ลูกค้าแต่ละรายใช้ มีหน่วยเป็น Kwh แล้วคำนวณและพิมพ์จำนวนเงินที่ลูกค้าต้องจ่ายให้กับบริษัท (มีหน่วยเป็น Dollar)

11. จงเขียน flowchart และโปรแกรมอ่านตัวเลข 2 จำนวน แล้วพิมพ์เลข 2 จำนวนนี้ จากน้อยไปหามาก

12. จงเขียน flowchart และ โปรแกรมอ่านตัวเลข 3 จำนวน แล้วพิมพ์เลข 3 จำนวนนี้จากน้อยไปหามาก

13. จงเขียน flowchart และโปรแกรมคำนวณหาค่าเกรดเฉลี่ยจากผลการสอบไล่ของนักศึกษาจำนวนหนึ่ง โดยมีหลักเกณฑ์การคิดดังนี้

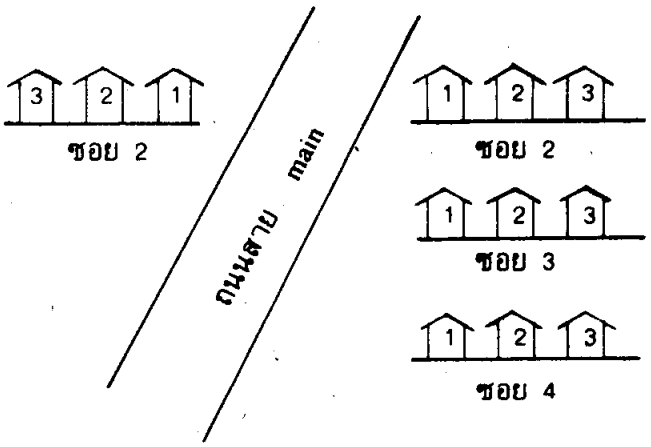
| คะแนนสอบ | เกรดเฉลี่ย |
|---|---|
| 90–100 | 4 |
| 80–89 | 3 |
| 72–79 | 2 |
| 65–71 | 1 |
| น้อยกว่า 65 | 0 |

ข้อมูลของนักศึกษาแต่ละคนบันทึกในบัตร 80 คอลัมน์หนึ่งใบ มีรหัสประจำตัว ชื่อ นามสกุล คะแนนสอบ สำหรับบัตรสุดท้ายให้เจาะหมายเลข 99999999 แล้วพิมพ์คะแนนเฉลี่ยของนักศึกษาแต่ละคนออกมาทาง line printer

**ตัวอย่าง** output

| คะแนนสอบ | เกรด |
|---|---|
| 95 | 4 |
| 64 | 0 |
| 65 | 1 |
| 71 | 1 |
| 72 | 2 |
| 79 | 2 |
| 80 | 3 |
| 90 | 4 |
| 89 | 3 |
| 100 | 4 |

14. หมู่บ้านจัดสรรเล็ก ๆ แห่งหนึ่งมีบ้านอยู่ทั้งหมด 12 หลัง ปรากฏตามแผนที่ข้างล่างนี้




บริษัทตั้งราคาบ้านไว้ดังนี้

| | ( หลังที่ 1 | หลังที่ 2 | หลังที่ 3 ) |
|---|---|---|---|
| ซอย 1 | 7,000 | 10,500 | 11,250 |
| ซอย 2 | 12,500 | 12,000 | 8,500 |
| ซอย 3 | 7,800 | 13,750 | 8,900 |
| ซอย 4 | 10,750 | 11,500 | 10,200 |

จงเขียน flowchart และโปรแกรม อ่านข้อมูล 4 ใบ

บัตรใบที่ 1 บันทึกราคาบ้าน 3 หลัง ในซอยที่ 1
บัตรใบที่ 2 บันทึกราคาบ้าน 3 หลัง ในซอยที่ 2
บัตรใบที่ 3 บันทึกราคาบ้าน 3 หลัง ในซอยที่ 3
บัตรใบที่ 4 บันทึกราคาบ้าน 3 หลัง ในซอยที่ 4

แล้วคำนวณราคาเฉลี่ยของบ้านในแต่ละซอย จากนั้นให้พิมพ์ผลลัพธ์ออกมาในลักษณะดังนี้

AVERAGE HOUSE VALUES

THE AVERAGE HOUSE VALUE FOR CROSS-STREET 1 IS 9583

THE AVERAGE HOUSE VALUE FOR CROSS-STREET 2 IS 11000

THE AVERAGE HOUSE VALUE FOR CROSS-STREET 3 IS 10150

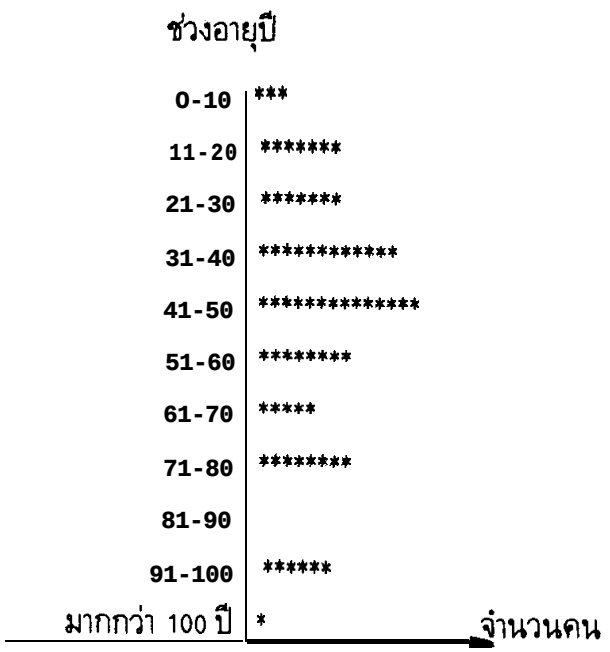
THE AVERAGE HOUSE VALUE FOR CROSS-STREET 4 IS 10816

15. จงเขียน *flowchart* และโปรแกรมทำตามขั้นตอนต่อไปนี้

a) อ่านข้อมูลจากบัตรจำนวน 500 ใบ ในบัตรแต่ละใบที่คอลัมน์ 1 - 3 เป็นอายุของคน ให้เก็บข้อมูลทั้งหมดนี้ใน array A 1 มิติ

b) นับจำนวนคนที่อยู่ในช่วงอายุ 0 - 10 ปี, 11 - 20, 21 - 30, 31 - 40, 41 - 50 ....., 91 - 100 ปี และกลุ่มที่มากกว่า 100 ปีขึ้นไป ข้อมูลนี้เก็บใน array B

c) พิมพ์ตารางแจกแจงความถี่แยกตามกลุ่มอายุคน ดังตัวอย่างข้างล่างนี้




**หมายเหตุ** * หมายถึง 1 คน

**เฉลยข้อ 10**

PL/1 OPTIMIZING COMPILER WATER: PROCEDURE OPTIONS (MAIN);

SCURCE LISTING

```
STMT LEV NT

1    0 WATER: PROCEDURE OPTIONS (MAIN);
2  1  0       DECLARE SYSIN INPUT, SYSPRINT OUTPUT;
3  1  0       DECLARE NAME CHARACTER (32);
4  1  0       DECLARE KWH FIXED DECIMAL (4);
5  1  0       DECLARE K FIXED DECIMAL (8,2);
6  1  0       DECLARE N FIXED DECIMAL (6,2);
7  1  0       ON ENDFILE (SYSIN) STOP;
              /* PRINT HEADER */
8  1  0       PUT PAGE;
9  1  0       PUT SKIP (4) EDIT
              ('NAME','USED(KWH)','AMOUNT(CENT)','PAY(DOLLAR)'
              (X(26),A,X(26),A,X,(12),A,X(12),A);
10 1  0 QUANT: GET LIST (NAME,KWH);
11 1  0       IF KWH<=90 THEN K=KWH*.11.75; ELSE
12 1  0       IF (KWH<90) & (KWN<=540) THEN K=90*11.75+(KWH-90)*3.74; ELSE
13 1  0       K=90*11.75+450*3.74+(KWH-540)*3.04;
14 1  0       N=K/100;
15 1  0       PUT SKIP(2) EDIT (NAME,KWH,K,N)
                        (X(12),A,X(12),F(4),X(17),F(8,2),X(16),F(6.2));
16 1  0       GO TO QUANT;
17 1  0       END WATER;
```

| NAME | USED(KWH) | AMOUNT(CENT) | PAY(DOLLAR) |
|---|---|---|---|
| LIVER BROTHER (THAILAND) LTD. | 1000 | 4138.90 | 41.38 |
| SAHAVIRIYA PANICH CO.,LTD. | 800 | 3530.90 | 35.30 |
| SAHAUNION CO.,LTD. | 540 | 2740.50 | 27.40 |
| THAI HINO MOTERSALES CO.,LTD. | 190 | 1431.50 | 14.31 |
| TOYOTA MOTER(THAILAND) CO.,LTD. | 75 | 881.25 | 8.81 |
| TERDMIT(THAILAND) CO.,LTD. | 10 | 117.50 | 1.17 |

**เฉลยข้อ 13**

```
PL/I OPTIMIZING COMPILER          GRADE:PROCEDURE OPTIONS(MAIN);

                    SOURCE  LISTING

SIMT LEV NT

    1     0 GRACE:  PROCEDURE OPTIONS (MAIN);
    2  1  0          DECLARE SYSIN INPUT, SYSPRINT OUTPUT;
    3  1  0          DECLARE NAME CHARACTER (29);
    4  1  0          DECLARE NO FIXED DECIMAL (8);
    5  1  0          DECLARE MARK FIXED BINARY;
    6  1  0          DECLARE FIXED BINARY;
                      /* PRINT HEADER */
    7  1  0          PUT PAGE;
    8  1  0          PUT SKIP(4) EDIT ('STUDENT NO.','NAME','EXAM MARK',
                      'GRADE')(X(12),A,X,(14),A,X,(21),A,X,(12),A);
    9  1  0 BEGIN:    GET LIST (NO,NAME,MARK);
   10  1  0          IF NO=99999999 THEN STOP;
   11  1  0          IF MARK>=90 THEN J=4; ELSE
   12  1  0          IF (MARK>=80) & (MARK<=89) THEN J=3; ELSE
   13  1  0          IF (MARK>=72) & (MARK<=79) THEN J=2; ELSE
   14  1  0          IF (MARK>=65) & (MARK<=71) THEN J=1; ELSE
   15  1  0          J=0;
   16  1  0          PUT SKIP (2) EDIT (NO,NAME,MARK,J)
                                 (X(13),F(8),X(9),A,X(5),F(3),X(19),F(1));
   17  1  0          GO TO BEGIN;
   18  1  0          END GRADE;
```

| STUDENT NO. | NAME | EXAM MARK | GRADE |
|---|---|---|---|
| 19701100 | APISIT NAŞKUL | 95 | 4 |
| 19701327 | DANAI SUGARASORN | 64 | 0 |
| 19701611 | JUREE TONGSAVAT | 65 | 1 |
| 19701696 | VICHET PIRIYAKUL | 71 | 1 |
| 19702141 | PRONTIP SUSEVA | 72 | 2 |
| 19702999 | ORAPIN DUMRONGSIRI | 79 | 2 |
| 19704041 | SOMCHAI PURAYA | 80 | 3 |
| 19704027 | RASAMEE KONGTONG | 90 | 4 |
| 19704991 | KSISANA LEKAUDOM | 89 | 3 |
| 19705001 | PRAMOTE SEANGMANEE | 100 | 4 |

L8II 'ONCODE'=0070 'ENDFILE' CONDITION RAISED ('ONFILE'= SYSIN)
IN STATEMENT 9 AT OFFSET +000100 IN PROCEDURE WITH ENTRY GRADE

**เฉลยข้อ 15**

```
PROG3:PROCEDURE OPTIONS (MAIN);
      DCL A(100) DEC FIXED(3), B(11) DEC FIXED(3) INIT((11)0),
                 (N,CHECK) DEC FIXED(3) INIT(0);
      DCL (I,J,K) FIXED BINARY;
      DCL C(11) CHAR(6) INIT('0-10','11-20','21-30','31-40,'41-50',
                             '51-60','61-70','71-80','81-90',
                             '91-100','>100');
   DCL STAR CHAR(50) VARYING INIT((50)'*'), OUT CHAR(50) VARYING,
   DCL SUBSTR BUILTIN;
   DCL SYSIN INPUT, SYSPRINT OUTPUT;
      GET LIST((A(I) DO I=1 TO 100)); /* GET VALUE OF ARRAY A */
      DO J=1 TO 100;
        DO K=10 TO 100 BY 10;
          N=K/10;
            IF A(J)<=K THEN DO;
                              B(N)=B(N)+1; GO TO AB;
                            END;
       END;
           B(11)=B(11)+1;
      AB:  CHECK=CHECK+1;
      END;
```

```
    PUT EDIT ('CREATING HISTROGRAM OF SAMPLING GROUP')
                (SKIP(5),COL(46),A);
       PUT EDIT ((40)'=') (SKIP(l), COL(44),A);
       PUT EDIT ('SAMPLING GROUP(AGE)') (SKIP(3;,COL(56),A);
    PUT EDIT ((A(1) DO I=1 TO 100)) (SKIP(3),COL(46),(10)(F(3),X(2)));
    PUT EDIT ('CHART OF HISTROGRAM')(SKIP(5),COL(56),A,SKIP (3));
       DO I=11 TO 1 BY -1;
         OUT=SUBSTR (STAR,1,B(I));
    PUT EDIT (C(I),'|',OUT)
          (SKIP(1),COL(46),A,COL(54),A,COL(56),A);
       END;
    PUT EDIT ((50)'_')(SKIP(1),COL(46),A,);
    PUT EDIT ('TOTAL PERSONS=',CHECK)(SKIP(3),COL(63),A,F,(3));
    PUT EDIT ('NOTE *= A PERSON')(SKIP(3),COL(70),A);
END PROG3;
```

CREATING HISTROGRAM OF SAMPING GROUP

SAMPLING GROUP(AGE)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 42 | 56 | 79 | 58 | 42 | 34 | 48 | 35 | 43 | 20 |
| 56 | 78 | 52 | 18 | 95 | 83 | 29 | 65 | 58 | 79 |
| 42 | 38 | 19 | 25 | 28 | 25 | 15 | 10 | 40 | 80 |
| 120 | 20 | 90 | 18 | 35 | 52 | 46 | 35 | 73 | 17 |
| 50 | 69 | 85 | 20 | 28 | 35 | 38 | 21 | 32 | 34 |
| 3 | 9 | 59 | 18 | 17 | 28 | 35 | 46 | 63 | 32 |
| 17 | 33 | 80 | 35 | 80 | 120 | 103 | 18 | 28 | 36 |
| 20 | 23 | 24 | 45 | 40 | 17 | 16 | 35 | 42 | 29 |
| 48 | 35 | 32 | 36 | 30 | 28 | 39 | 19 | 13 | 8 |
| 19 | 107 | 84 | 5 | 79 | 18 | 46 | 38 | 45 | 90 |

CHART OF HISTROGRAM

| | |
|---|---|
| >100 | **** |
| 91–100 | * |
| 81-90 | ***** |
| 71-80 | ******** |
| 61-70 | *** |
| 51-60 | ******* |
| 41-50 | ********** |
| 31-40 | ********** |
| 21-30 | ********** |
| 11-20 | ********** |
| O-10 | ***** |

TOTAL PERSONS=100

NOTE * = A PERSON