บทที่ 8: The EXTERNAL attribute

นอกจากชื่อ file ซึ่งปกติเป็น EXTERNAL แล้วในโปรแกรม compiler จะจัด INTERNAL attribute โดยอัตโนมัติให้กับทุกชื่อ ซึ่งไม่ได้ใช้ EXTERNAL attribute

INTERNAL หมายถึง ชื่อ ซึ่งทราบความหมายเฉพาะภายในบล๊อกที่มีการ declare และบล๊อกใด ๆ ซึ่งมีชื่อนั้นอยู่ วิธีที่จะทำให้การ declare ทราบนอก procedure ของตนคือให้ใช้ EXTERNAL attribute และ declare ชื่อนั้นต่างหากในแต่ละ external procedure ที่ต้องการใช้ วิธีนี้ EXTERNAL attribute จะทำให้การ declare ทราบภายนอก procedure ของตน

ให้พิจารณาโครงสร้างของโปรแกรมข้างล่างนี้ ซึ่งแสดงการ declare EXTERNAL ใน external procedure

```
PROGA: PROCEDURE OPTIONS (MAIN);
          DCL PROGB EXTERNAL ENTRY;
          DCL DIST DEC FIXED (7,2) EXTERNAL,
              TEMP DEC FIXED (5,2) EXTERNAL,
              PRESS BIN FIXED (15);
                    :
              CALL PROGB;
                    :
END PROGA;
PROGB: PROCEDURE;
   DCL DIST DEC FIXED (7,2) EXTERNAL,
          TEMP DEC FIXED (5,2) EXTERNAL,
          PRESS BIN FIXED (15);
                :
 END PROGB;
```

ใน main procedure PROGA เรียก external procedure PROGB ภายใต้ optimizing compiler, PROGA ต้อง declare PROGB เป็น ENTRY เพื่อที่ว่า Linkage editor จะสร้างตำแหน่ง

หน่วยความจำได้อย่างถูกต้อง สำหรับเรียก PROGB ระหว่างการ execute

ใน Procedure ทั้งสองมีคำสั่ง declare DIST, TEMP, และ PRESS เฉพาะ DIST และ TEMP เท่านั้นที่มี EXTERNAL attribute ดังนั้นการอ้างถึง DIST หรือ TEMP ในแต่ละ external procedure หมายถึงสิ่งเดียวกัน แต่สำหรับ PRESS โปรแกรม compiler จะจัด INTERNAL attribute ให้ ฉะนั้นจึงหมายถึงสอง fields ที่แตกต่างกัน และมีความหมายเฉพาะ (unique) ใน procedure ของมันเองเท่านั้น

การ declare external, DIST และ TEMP, แต่ละตัวเครื่องจะเตรียมเนื้อที่ไว้หนึ่ง field, ทั้งสอง field นี้โดยแท้จริงอยู่ในโปรแกรมเรียก PROGA, การใช้ EXTERNAL attribute ทำให้ตำแหน่งที่อยู่ของมันทราบได้ใน external procedure อื่น ถ้าต้องการกำหนดมูลค่าเริ่มต้น ให้ใช้ INITIAL attribute เฉพาะใน main procedure PROGA เท่านั้น โปรแกรม compiler จะไม่สนใจ INIT attribute ที่กำหนดให้กับ DIST และ TEMP ถ้าปรากฏใน procedure PROGB นอกจากนี้แล้วการ declare ตัวที่ตรงกันต้องเหมือนกันในทุก external procedure

โปรแกรม compiler จะ generate รหัสจำนวนหนึ่งสำหรับแต่ละตัวที่ใช้ EXTERNAL attribute ในโปรแกรมใหญ่ที่มีการ declare มาก ๆ จะมีประสิทธิภาพมากขึ้น ถ้าเราจะลดจำนวนการ declare EXTERNAL ให้มีน้อยที่สุด การปฏิบัติที่เป็นมาตรฐาน คือการกำหนดทั้งหมดในโครงสร้างเดียวให้มี EXTERNAL attribute แล้วแต่ละตัวจะมี EXTERNAL attribute แต่ compiler จะ generate external address ให้เฉพาะโครงสร้างรวมเท่านั้น

ตัวอย่าง

```
DCL 1 COMMON EXTERNAL,
      2 DIST DEC FIXED (7,2),
      2 TEMP DEC FIXED (5,2);
```

ตอนนี้ อีลิเม้นต์ทุกตัวในโครงสร้าง COMMON จะมี EXTERNAL attribute เพื่อให้แน่ใจว่าในแต่ละ procedure ถูกเรียก มีการ declare ด้วย attribute เหมือนกัน วิธีง่าย ๆ คือการ duplicate คำสั่ง declare แล้วใส่ลงในทุก procedure ที่ใช้ร่วมกัน นอกจาก INIT attribute แต่ละโครงสร้างในแต่ละ procedure ซึ่งชื่อและ attribute ตรงกัน ต้อง declare เหมือนกันและเรียงลำดับเหมือนกันด้วย ใน file เดียวกันที่ใช้ใน external procedure มากกว่า 1 ชุด ต้อง declare เหมือนกัน แต่ไม่จำเป็นต้องใช้ EXTERNAL

หมายเหตุ ถ้าเราใช้ PL/1 Optimizing compiler ให้ใส่คำสั่ง PROCESS ก่อนแต่ละ external procedure โดยมีรูปแบบดังนี้

*PROCESS

เครื่องหมาย * อยู่ในคอลัมน์ 1. PROCESS อาจจะมี option อื่น ๆ ได้อีก ซึ่งจะบรรยายในหัวข้อต่อไป

ตัวอย่าง โปรแกรม ที่มี external procedures

```
/* MAIN PROCEDURE */
*PROCA: PROCEDURE OPTIONS (MAIN);
        DCL SYSIN INPUT;
        DCL 1 COMMON EXTERNAL,
              2 HOURS DEC FIXED (5,1),
              2 RATE DEC FIXED (5,2),
              2 WAGE DEC FIXED (9,2);
        DCL PROCB EXTERNAL ENTRY;
          GET LIST (HOURS, RATE);
          CALL PROCB;
        END PROCA;
*PROCESS
/* EXTERNAL PROCEDURE : PROCB */
  PROCB: PROCEDURE;
          DCL 1 COMMON EXTERNAL,
                2 HOURS DEC FIXED (5,1),
                2 RATE DEC FIXED (5,2),
                2 WAGE DEC FIXED (9,2),
          DCL   PROCC EXTERNAL ENTRY;
                WAGE = HOURS * RATE;
                CALL PROCC;
              END PROCB;
*PROCESS
/* EXTERNAL PROCEDURE : PROCC */
    PROCC: PROCEDURE;
             DCL SYSPRINT OUTPUT;
```

```
        DCL 1 COMMON EXTERNAL,
              2 HOURS DEC FIXED (5,1),
              2 RATE DEC FIXED (5,2),
              2 WAGE DEC FIXED (9,2);
            PUT DATA (HOURS, RATE, WAGE);
          END PROCC;
```

พิมพ์ผลลัพธ์ดังนี้

```
COMMON.HOURS= 123.0  COMMON.RATE= 9.25  COMMON.WAGE= 1137.75;
```

Arguments and parameter

common data ในโปรแกรมหนึ่ง อาจเอาไปใช้ได้ใน external procedure มากกว่าหนึ่งชุด โดยใช้ EXTERNAL attribute หรือโดยการใช้ arguments และ parameters หรือโดยการใช้ทั้งสองวิธีปนกัน ตัวอย่างโปรแกรมต่อไปนี้แสดงการส่งมูลค่า arguments จาก procedure เรียก MAIN ไปยัง procedure ถูกเรียก SUB

```
MAIN: PROCEDURE OPTIONS (MAIN);
        DCL SUB EXTERNAL ENTRY;
        DCL DIST DEC FIXED (7,2),
            TEMP DEC FIXED (5,2) INIT (0),
            PRESSURE BIN FIXED (15);
                    .
                    .
            CALL SUB (DIST, TEMP); arguments
                    .
                    .
            END MAIN;
SUB: PROCEDURE (DISTANCE, TEMPER); parameters
      DCL DISTANCE DEC FIXED (7,2),
          TEMPER DEC FIXED (5,2),
          PRESSURE BIN FIXED (15);
                    .
                    .
       END SUB;
```

หมายเหตุ compiler สร้างความสัมพันธ์ระหว่าง arguments กับ พารามิเตอร์ จากซ้ายไปขวา และมีข้อจำกัดดังนี้

1. arguments และ พารามิเตอร์ ที่ตรงกันต้องอยู่ในลำดับเดียวกัน
2. จำนวน arguments และจำนวน พารามิเตอร์ ต้องเท่ากัน
3. argument และ พารามิเตอร์ แต่ละคู่ที่ตรงกันได้ จะต้องมี attribute เหมือนกัน อย่างไรก็ตาม compiler จะไม่สนใจการ declare INITIAL attribute ของ พารามิเตอร์ ใน procedure ถูกเรียก ดังนั้นโปรแกรมเมอร์จึงไม่ควรกำหนด attribute นี้ ให้กับ พารามิเตอร์ นอกจากนี้แล้ว พารามิเตอร์ จะต้องไม่มี attributes AUTOMATIC, STATIC, BASED, DEFINED หรือ EXTERNAL
4. ถ้าจำเป็นต้องมีการส่งผ่านโครงสร้าง ต้องแน่ใจว่าทั้ง argument และ พารามิเตอร์ เป็น level−1

ในโปรแกรมตัวอย่างที่แสดงข้างต้น DISTANCE และ TEMPER defined เหมือนกับ DIST และ TEMP ตามลำดับ ใน procedures ทั้งสองมีการ defined PRESSURE แต่ไม่ได้ใช้ EXTERNAL attribute และไม่มีการส่งมูลค่าของ PRESSURE ผ่านทาง argument ด้วย ดังนั้น การใช้ PRESSURE ในแต่ละ procedure จึงเป็นเนื้อที่คนละแห่งกันในหน่วยความจำหลัก

แบบฝึกหัด

ให้โปรแกรมหนึ่งประกอบด้วย main procedure 1 ชุด และ external procedure อีก 1 ชุด ใน main procedure มีข้อมูลชื่อ DISTANCE, TIME และ SPEED เป็นตัวเลขไม่เกิน 7 หลัก (เลข 3 ตัวหลังเป็นจุดทศนิยม) ข้อมูลนี้ต้องการเอาไปใช้ใน external procedure ด้วย โดยใช้ชื่อเหมือนกัน เมื่อหมายถึงสิ่งเดียวกัน

จงเขียน procedure ทั้ง 2 ชุดข้างต้นใน main procedure อ่านมูลค่าของ DISTANCE และ TIME แล้วเรียก external procedure จากนั้นพิมพ์ค่า DISTANCE, TIME และ SPEED ใน external procedure คำนวณหา SPEED จากสูตร

$$\text{SPEED} = \frac{\text{DISTANCE}}{\text{TIME}}$$