บทที่ 7 :
โปรแกรมย่อย (Subprograms)

เนื้อหาในบทนี้จะครอบคลุมการ linkage และ execution ของโปรแกรมหลัก และโปรแกรมย่อยที่แยกคอมไพล์ต่างหากจากกัน

7.1 ข้อแนะนำเบื้องต้น

สิ่งที่นักศึกษาต้องฝึกหัดคือ ลิงค์ (link) ระหว่างโปรแกรมต่าง ๆ ที่แยกคอมไพล์ต่างหากจากกัน โปรแกรมหนึ่งจะมีชื่อว่าโปรแกรมเรียก (calling program) และโปรแกรมที่ถูกเรียกจะมีชื่อว่า โปรแกรมย่อย (subprogram) ทั้งสองโปรแกรมนี้จะคอมไพล์เป็นอิสระจากกัน แต่ execute พร้อมกันเพื่อให้ได้ผลลัพธ์ตามที่เราต้องการ

ทำไมจึงต้องลิงค์โปรแกรมชุดหนึ่งกับโปรแกรมย่อยอื่น ๆ เหตุผล

(1) โปรแกรมย่อยเหล่านั้นได้เขียนและทดสอบมาแล้ว นอกจากนี้เป็นการง่ายต่อการคอมไพล์และแคตตาลอก (cataloged)

(2) โปรแกรมย่อยเหล่านั้นสามารถนำไปใช้ได้โดยโปรแกรมใด ๆ ก็ได้ในระบบของคอมพิวเตอร์เครื่องนั้น

(3) โครงการที่ทำเป็นทีม (team project) การแบ่งโปรแกรมหนึ่งโปรแกรมให้เป็นโปรแกรมเล็ก ๆ หลายโปรแกรม จะมีความยืดหยุ่นมากขึ้น

(4) โปรแกรมย่อยเหล่านั้นจะเขียนด้วยภาษาใดก็ได้ และโปรแกรมที่เรียกโปรแกรมย่อยไปใช้อาจจะเขียนด้วยอีกภาษาหนึ่งที่ไม่เหมือนกัน ตัวอย่างเช่น เราเรียกโปรแกรมย่อยที่เป็นภาษา assembler เพื่อใช้รับ/ส่งข้อมูล แต่โปรแกรมหลักเขียนด้วยภาษาโคบอลเพื่อการประมวลผลพิเศษกับข้อมูลนั้น

เมื่อนักศึกษาออกแบบ logic ของโปรแกรมย่อยแล้ว มีสิ่งที่ต้องเพิ่มเติมเพียง 2 แห่งเท่านั้น และทั้งสองสิ่งเขียนง่ายมากในภาษาโคบอล

สิ่งแรก การลิงค์ระหว่างโปรแกรมเรียก (calling program) กับโปรแกรมถูกเรียก (called program)

สิ่งที่สอง การเตรียมการรับ/ส่งข้อมูล ในทั้งสองโปรแกรมนั้นต้องทำได้ปกติ

7.2 การลิงค์โปรแกรม (Program linkage)

ในโปรแกรมเรียกต้องการเพียงคำสั่ง CALL เท่านั้น เพื่อลิงค์กับโปรแกรมย่อยโดยมีรูปแบบทั่วไปดังนี้

CALL 'entry-name' [**USING** data-name-1, ...]

คำสั่งนี้ control จะถูกส่งออกจากโปรแกรมหลักไปยังโปรแกรมถูกเรียก, entry-name ปกติคือชื่อของโปรแกรมย่อย (ชื่อในพารากราฟ PROGRAM-ID ของโปรแกรมย่อย) แต่บางครั้งอาจจะเป็นชื่อของ entry point ที่อยู่ภายใน procedure division ของโปรแกรมย่อยก็ได้

Entry point ของโปรแกรมถูกเรียกขึ้นอยู่กับคำสั่ง CALL นั้นระบุชื่อในพารากราฟ PROGRAM-ID หรือว่าชื่อ entry อื่น, ถ้าระบุชื่อในพารากราฟ PROGRAM-ID, การ execute โปรแกรมย่อยจะเริ่มตรงจุดเริ่มต้นของ procedure division, ถ้าระบุชื่อ entry อื่น จะต้องมีคำสั่ง ENTRY พิเศษภายใน procedure division ก่อนชื่อของ procedure ถูกเรียกเพื่อบอกตำแหน่งของ entry point

คำสั่งในโปรแกรมย่อยที่จะส่ง control กลับไปยังโปรแกรมเรียกมีรูปแบบดังนี้

para-name **EXIT PROGRAM**.

ตำแหน่งที่กลับมาคือคำสั่งซึ่งอยู่ถัดจากคำสั่ง CALL ในโปรแกรมเรียก EXIT PROGRAM ต้องเป็นคำสั่งเดียวในพารากราฟนั้น ในโปรแกรมย่อยชุดหนึ่งอาจจะเรียกโปรแกรมย่อยอีกชุดหนึ่งได้ แต่จะใช้คำสั่ง CALL เรียกโปรแกรมตัวมันเองไม่ได้

ถ้าคอมพิวเตอร์เครื่องที่เราใช้เป็นยี่ห้อ IBM จะมีอีกคำสั่งหนึ่งคือ GOBACK ซึ่งใช้แทนคำสั่ง EXIT PROGRAM นอกจากนี้แล้วไม่จำเป็นต้องมีชื่อพารากราฟนำหน้าคำสั่ง GOBACK แต่มีข้อแนะนำว่าถ้าเราใช้ ANS COBOL เขียนโปรแกรม การใช้คำสั่ง EXIT PROGRAM จะเป็นระบบสากลที่เขาใช้ทั่ว ๆ ไป

7.3 ข้อมูลที่ใช้ร่วมกัน (Common data)

ปกติ, โปรแกรมเรียกจะส่งชื่อของข้อมูล (data-name) บางตัวไปยังโปรแกรมย่อยแล้วโปรแกรมย่อยจะใช้ชื่อข้อมูลเหล่านี้ และข้อมูลของมันเองสร้างข้อเท็จจริง (information) ให้กับโปรแกรมหลัก

ตัวอย่าง สมมติให้โปรแกรมหลัก 1 ชุด ส่งต้นทุนในการผลิต (product cost) และจำนวนที่ผลิต (quantity sold), ในโปรแกรมย่อยคูณต้นทุนในการผลิตกับจำนวนที่ผลิตแล้ว

เก็บผลลัพธ์ไว้ในเนื้อที่ราคาขาย (value sold field) ให้นักศึกษาพิจารณาบางส่วนของโปรแกรมจากรูปที่แสดงข้างล่างนี้

โปรแกรมหลัก (โปรแกรมเรียก)

```
DATA DIVISION
.
.
.
WORKING-STORAGE SECTION.
77  PROD-COST      PIC S9(3)V99     COMP-3.
77  OTY-SOLD       PIC S9(4)        COMP-3.
77  VALUE-SOLD     PIC S9(7)V99     COMP-3.
PROCEDURE  DIVISION.
.
.
.
 CALL 'CALCPROG' USING PROD-COST, QTY-SOLD. VALUE-SOLD.
```

โปรแกรมถูกเรียก (โปรแกรมย่อย)

```
IDENTIFICATION DIVISION
PROGRAM-ID.    CALCPROG.
ENVIRONMENT  DIVISION.
.
.
.
DATA  DIVISION
.
.
.
WORKING-STORAGE  SECTION
.
.
.
LINKAGE  SECTION.
77  PROD-COST      PIC S9(3)V99     COMP-3
```

```
77  QTY-SOLD        PIC S9(4)        COMP-3.
77  VALUE-CALC      PIC S9(7)V99     COMP-3.
PROCEDURE DIVISION USING PROD-COST, QTY-SOLD. VALUE-CALC.
A00-SECTION.
    COMPUTE VALUE-CALC ROUNDED = PROD-COST * QTY-SOLD.
A99-RETURN
    EXIT PROGRAM
```

รูปแสดงการลิงค์ระหว่างโปรแกรมหลักกับโปรแกรมย่อย

จากรูปที่แสดงข้างต้นนี้ในโปรแกรมหลัก คำสั่ง CALL.....USING ส่งชื่อ PROD-COST, QTY-SOLD และ VALUE-SOLD ไปยังโปรแกรมย่อยชื่อ CACLPROG ในโปรแกรมย่อย USING clause ในคำสั่ง PROCEDURE DIVISION ได้กำหนดชื่อไว้ 3 ชื่อเช่นกัน คือ PROD-COST, QTY-SOLD และ VALUE-CALC ชื่อตัวท้ายสุดนี้ผู้เขียนโปรแกรมตั้งใจตั้งให้แตกต่างจากชื่อท้ายสุดในคำสั่ง CALL เพื่อแสดงความชัดเจนในขั้นอธิบาย

กฎเกณฑ์ต่อไปนี้เกี่ยวข้องกับ**ชื่อเหล่านี้** (บางทีเรียกว่า arguments หรือ parameters)

(1) คำสั่ง USING ในโปรแกรมเรียก และโปรแกรมถูกเรียก จะต้องประกอบด้วยจำนวนชื่อที่เท่ากัน แต่ไม่จำเป็นต้องใช้ชื่อเหมือนกัน

(2) ชื่อแรกในโปรแกรมถูกเรียกต้องคู่กับ (match) ชื่อแรกในคำสั่ง CALL.....USING และชื่อที่สองต้องคู่กับชื่อที่สอง เช่นนี้เรื่อยไป

ตัวอย่าง

โปรแกรมเรียก :	PROD-COST	QTY-SOLD	VALUE-SOLD
	↓	↓	↓
โปรแกรมย่อย :	PROD-COST	QTY-SOLD	VALUE-CALC

(3) ในโปรแกรมย่อยต้องมี linkage section (ถ้าโปรแกรมนี้มี working-storage section ให้เขียนหลัง working-storage section) ซึ่งจะเป็นส่วนหนึ่งของโปรแกรมที่เราใช้สำหรับ defined

---

สำหรับเครื่อง IBM 360/370 Usage clause มีรูปแบบต่าง ๆ กันดังนี้

COMPหมายถึง ข้อมูลเป็นเลขฐานสอง (binary data)

COMP-1 และ COMP-2 หมายถึง ข้อมูลชนิด floating-point data

COMP-3 หมายถึง ข้อมูลชนิด package decimal (หนึ่งไบต์ประกอบด้วยเลข 2 ตัว บวกกับเครื่องหมายอีก 1 ตัว)

ชื่อทั้งหมดที่มีอยู่ใน USING clause ชื่อที่คู่กันนั้นจะต้อง defined เนื้อที่เก็บข้อมูลไว้เหมือนกัน เช่น ชื่อแรก ในคำสั่ง CALL.....USING ทั้งในโปรแกรมหลักและใน USING clause ของโปรแกรมย่อย defined ไว้ดังนี้

PIC S9(3)V99 COMP-3.

ถึงแม้ว่า arguments ทั้ง 2 ตัวนี้ นิยามเนื้อที่กับข้อมูลไว้เหมือนกัน แต่อาจจะตั้งชื่อไม่เหมือนกันก็ได้

ใน linkage section จะประกอบด้วยชื่อซึ่งใช้เลขบอกระดับ 77 หรือ 01 และในกลุ่ม 01 อาจจะประกอบเลขบอกระดับ 03, 05 เป็นต้น นักศึกษาอาจเห็นว่าการตั้งชื่อเหมือนกันทำให้โปรแกรมชัดเจนกว่าก็ได้

(4) โปรแกรมหลักต้องเตรียมเนื้อที่ (defined) สำหรับชื่อเริ่มต้น (original names) ในโปรแกรมย่อยอ้างถึงชื่อเหล่านี้ แต่ชื่อเหล่านี้ต้อง defined ใน linkage section ด้วย เพราะมันไม่ได้มีอยู่จริงในโปรแกรมย่อย, โปรแกรมย่อยใช้ตำแหน่งที่อยู่ (address) ของชื่อเพื่อค้นหาและอ้างถึงมูลค่า (content) ของมัน ตัวอย่างเช่น เมื่อโปรแกรมย่อยคำนวณค่า VALUE-SOLD มันจะเก็บผลลัพธ์ใน working-storage section ของโปรแกรมหลัก นอกจากนี้แล้วโปรแกรมย่อยอาจจะไม่ต้องส่งค่าอะไรกลับไปยังโปรแกรมหลักก็ได้

วิธีที่ดีอีกอย่างหนึ่งในการส่งค่าของ arguments ระหว่างโปรแกรมให้ใช้ level 01 entries ตั้งแต่ 1 ชุดขึ้นไป ตัวอย่างเช่น เราอาจส่ง level 01 data group ของ agruments ทั้งหมด 10 ตัว หรือ 12 ตัว โดยกำหนดเฉพาะชื่อ level 01 address เพียงที่อยู่เดียวเท่านั้น จริง ๆ แล้วโปรแกรมย่อยอาจจะอ้างถึงเพียงบางส่วนของ agruments นั้น แต่เนื่องจากค่าของ agruments เหล่านี้มีอยู่จริงเฉพาะในโปรแกรมหลักเท่านั้น การเขียนโปรแกรมตามลักษณะข้างต้นจึงไม่ได้ใช้เนื้อที่หน่วยความจำในโปรแกรมย่อยเพิ่มขึ้นแต่อย่างใด

7.4 Entry points

ถ้าหากว่าโปรแกรมหลักจำเป็นต้องเข้าไปในโปรแกรมย่อย ณ จุดใดจุดหนึ่งที่ไม่ใช่จุดเริ่มต้นของ procedure division เราต้องใช้คำสั่ง ENTRY คำสั่งนี้ ถึงแม้ว่าจะไม่ใช่คำสั่งมาตรฐานของ ANS COBOL แต่ก็ใช้ได้กับคอมไพเลอร์หลายตัว สำหรับรูปแบบของคำสั่ง ENTRY ของเครื่องคอมพิวเตอร์ IBM DOS และ DS มีดังนี้

ENTRY name [**USING** data-name-1 ....]

ตัวอย่างโปรแกรมข้างล่างนี้แสดงโปรแกรมหลัก ชุดหนึ่งเรียกโปรแกรมย่อย 2 แห่งภายใน procedure division โปรดสังเกตว่า routine ที่ถูกเรียกแต่ละชุดจะต้องมีคำสั่ง EXIT PROGRAM ของมันเอง

โปรแกรมหลัก (โปรแกรมเรียก)

```
CALL 'PENS-RTN' USING.....
CALL 'TAX-RTN' USING.....
```

โปรแกรมย่อยถูกเรียก

```
PROCEDURE DIVISION.



ENTRY 'PENS-RTN' USING
P00-PENSION.

    .

P99-RTN. EXIT PROGRAM.
ENTRY 'TAX-RTN' USING.....
S00-PENSION.



S99-RTN. EXIT PROGRAM.
```

7.5 ตัวอย่างโปรแกรม

ซึ่งประกอบด้วยโปรแกรมหลัก 1 โปรแกรม และโปรแกรมย่อยอีก 1 โปรแกรม โปรแกรมหลักส่งตำแหน่งที่อยู่ (address) ของจำนวนชั่วโมงทำงาน และรหัสของงานไปยังโปรแกรมย่อย, โปรแกรมย่อยค้นรหัสของงานจากตารางหนึ่ง เพื่อหาอัตราค่าจ้างในตารางนั้นแล้วคำนวณค่าจ้าง (จำนวนชั่วโมงทำงานคูณกับอัตราค่าจ้าง) โปรดสังเกตว่า ตารางแสดงรหัสของงานและอัตราค่าจ้างจะมีแห่งเดียวใน working-storage section ของโปรแกรมย่อย และปกติตารางนี้จะไม่อยู่ในโปรแกรมหลัก

โปรแกรมหลัก

```
IDENTIFICATION DIVISION.
PROGRAM-ID. MAINP.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT TIME-FILE ASSIGN TO DISK1.
    SELECT REPORT-FILE ASSIGN TO DISK2.
DATA DIVISION.
FILE SECTION.
FD TIME-FILE LABEL RECORDS ARE OMITTED.
01  TIME-REC.
    03  CODE-IN          PIC X(2).
    03  EMPNO-IN         PIC X(5).
    03  NAME-IN          PIC X(15).
    03  JOB-IN           PIC X(2).
    03  HOURS-IN         PIC S99V9.
FD REPORT-FILE LABEL RECORDS ARE OMITTED
01  REPORT-REC.
    03  FILLER           PIC X(11).
    03  EMPNO-OUT        PIC X(5).
    03  NAME-OUT         PIC X(15).
    03  JOB-OUT          PIC X(5).
    03  RATE-OUT         PIC ZZZ9.99
    03  FILLER           PIC X(3).
    03  HOURS-OUT        PIC Z9.9-.
    03  FILLER           PIC X(3).
    03  WAGE-OUT         PIC ZZ,ZZ9.99CR.
```

```
WORKING-STORAGE SECTION.
77  EOF PIC X(3) VALUE 'NO'.
01  COMMON-DATA.
    03  JOBNO          PIC 99.
    03  HOURS          PIC S99V9     COMP-3.
    03  RATE           PIC S999V99   COMP-3.
    03  WAGE           PIC S9(5)V99  COMP-3.
PROCEDURE DIVISION.
A00-BEGIN.
    OPEN INPUT TIME-FILE, OUTPUT REPORT-FILE.
    READ TIME-FILE AT END MOVE 'YES' TO EOF.
    PERFORM B00-PROCESS UNTIL EOF = 'YES'.
    CLOSE TIME-FILE, REPORT-FILE.
    STOP RUN.
B00-PROCESS SECTION.
    MOVE SPACES     TO REPORT-REC.
    MOVE HOURS-IN   TO HOURS.
    MOVE JOB-IN     TO JOBNO, JOB-OUT.
    CALL 'SUBPROG' USING COMMON-DATA.
    MOVE WAGE       TO WAGE-OUT.
    MOVE HOURS      TO HOURS-OUT.
    MOVE RATE       TO RATE-OUT.
    MOVE NAME-IN TO NAME-OUT.
    MOVE EMPNO-IN TO EMPNO-OUT.
    WRITE REPORT-REC AFTER ADVANCING 2 LINES.
    READ TIME-FILE AT END MOVE 'YES' TO EOF.
B99-RETURN.
    EXIT.
```

ตารางแสดงรหัสของงานและอัตราค่าจ้างที่จะให้เก็บในโปรแกรมย่อย

Job no.	Rate
01	07.25
02	06.15
04	07.45
05	08.55
06	07.50
08	09.25
10	08.95
99	00.00

โปรแกรมย่อย

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SUBPROG.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  I       PIC 9(2) VALUE ZEROS.
77  EOS     PIC X(3) VALUE 'YES'.
01  JOB-TABLE.
    03  FILLER PIC X(6) VALUE '010725'.
    03  FILLER PIC X(6) VALUE '020615'.
    03  FILLER PIC X(6) VALUE '040745'.
    03  FILLER PIC X(6) VALUE '050855'.
    03  FILLER PIC X(6) VALUE '060750'.
    03  FILLER PIC X(6) VALUE '080925'.
    03  FILLER PIC X(6) VALUE '100895'.
    03  FILLER PIC X(6) VALUE '990000'.
01  JOB-TABLE-COPY REDEFINES JOB-TABLE.
    03  TAB-ENTRIES OCCURS 8 TIMES
```

```
        O S  JOBNO-TAB PIC 99.
             05 K A T - T A R  PIC 99V99.
    LINKAGE  SECTION.
01  COMMON-DATA.
        03  JOHN0      PIC YY.
        03 HOURS       PIC S99V9     COMP-3.
        03  RATE       PIC S999V999 COMP-3.
        03 W A G E     PIC S9(5)V99 COMP-3.
PROCEDURE DIVISION USING COMMON-D; TA.
A00-MAIN SECTION.
    MOVE 'NO' TO EOS.
    PERFORM C00-SEARCH VARYING I FROM 1 BY 1 UNTIL EOS = 'YES'
A99-RTN. EXIT PROGRAM.
***T A B L E S E A R C H & C A L C  W A G E***
C00-SEARCH SECTION.
    IF  JOBNO = JOBNO-TAB (I)
        MOVE 'YES' TO EOS
        MOVE RATE-TAB(I) TO RATE
        COMPUTE WAGE ROUNDED = t IOURS * RATE
    ELSE
    It' JOBNO < JOBNO-TAB(I)
        MOVE 'YES' TO EOS
        MOVE ZEROS TO HOURS, RATE, WAGE.
WY  RETURN. EXIT.
```

7.6 Linkage editor

ในการ execute โปรแกรม สิ่งแรกต้องคอมไพล์แล้ว link-edited ด้วยโปรแกรม linkage editor ในขั้นคอมไพล์โปรแกรม หมายถึงการแปลโปรแกรมภาษาโคบอล (source program) ให้เป็นรหัสของภาษาเครื่อง (machine language object code) ตรงจุดนี้โปรแกรมยังไม่สามารถนำไป execute ได้ ขั้น link-edited ทำหน้าที่หลัก 2 อย่างคือ

(1) โปรแกรม linkage editor รวมทั้ง input/output modules ที่จำเป็นต้องใช้ทั้งหมดนี้ ได้ถูกคอมไพล์ก่อนแล้ว และแคตตาลอก (cataloged) ใน library ของระบบคอมพิวเตอร์

(2) โปรแกรม linkage editor จะลิงค์โปรแกรมซึ่งแยกคอมไพล์ต่างหากจากกันเข้าด้วยกัน (เช่น ลิงค์โปรแกรมหลักกับโปรแกรมย่อย) แล้วทำบัญชีสำหรับตำแหน่งที่อยู่ของข้อมูลทั้งหมดซึ่งโปรแกรมอ้างถึงใน common

สำหรับโปรแกรมภาษาโคบอล, Linkage editor ถูกเกี่ยวข้องกับคำสั่ง CALL USING ในโปรแกรมเรียก และ Linkage section, USING clause ในโปรแกรมถูกเรียก

โปรแกรม linkage editor เก็บโปรแกรมที่คอมไพล์แล้ว, object program ที่ได้ link-edited แล้วใน system library ที่ซึ่งจะถูก execute อีกครั้งหนึ่ง หลังจากนั้นก็จะถูกลบโดยโปรแกรมที่ได้ link-edited แล้วชุดถัดไป หรืออาจจะจัดเก็บถาวรใน library ก็ได้ สำหรับการ execute งานซึ่งต้องทำอยู่เป็นประจำ

★ ★ ★ ★ ★ ★

ตัวอย่างโปรแกรม

ป้าสายใจตั้งใจฝากเงินให้เป็นทุนการศึกษาของหลาน 3 คน คือ นายสมชาย, นางสาวสมศรีและนายสมพงศ์ ดังนี้

ปีที่ 1 ฝากให้สมชาย 2,000 บาท ฝากให้สมศรี 2,000 บาท ฝากให้สมพงศ์ 1,000 บาท
ปีที่ 2 ฝากให้สมชาย 3,500 บาท ฝากให้สมศรี 2,000 บาท ฝากให้สมพงศ์ 1,000 บาท
ปีที่ 3 ฝากให้สมชาย 2,500 บาท ฝากให้สมศรี 2,000 บาท ฝากให้สมพงศ์ 1,000 บาท
ปีที่ 4 ฝากให้สมชาย 1,000 บาท ฝากให้สมศรี 2,000 บาท ฝากให้สมพงศ์ 1,000 บาท
ปีที่ 5 ฝากให้สมชาย 1,000 บาท ฝากให้สมศรี 2,000 บาท ฝากให้สมพงศ์ 6,000 บาท

ธนาคารคิดดอกเบี้ยเงินฝากประจำร้อยละ 10 ต่อปี ต้องการทราบว่าเมื่อสิ้นปีที่ 10 (ปีที่ 6 จนถึงปีที่ 10 ไม่มีเงินฝากเพิ่ม) หลังจากหักดอกเบี้ยเงินฝากร้อยละ 10 แล้ว หลานทั้ง 3 คน ของป้าสายใจจะมีเงินทุนการศึกษาคนละเท่าไหร่

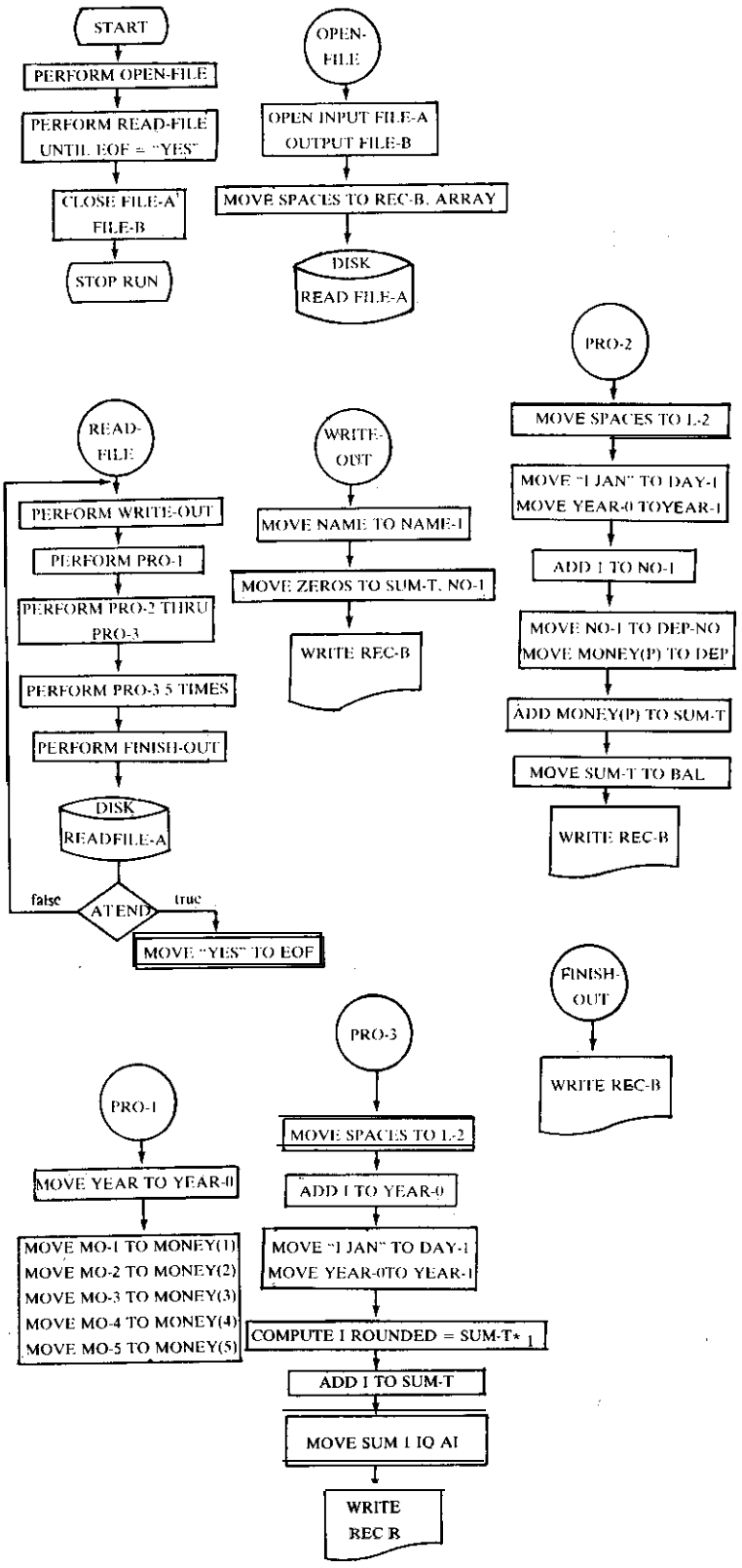
a) ให้นักศึกษากำหนดรูปแบบของ input เอง วาดรูปแสดงด้วย

b) เขียน flowchart และโปรแกรมพิมพ์ output ของหลานแต่ละคนให้มีลักษณะดังตัวอย่างข้างล่างนี้

COLLEGE FUND OF MR. SOMCHAI

Date	Deposite no.	Deposite	Interest	Balance
1 Jan 1986	1	2,000.00		2,000.00
1 Jan 1987			200.00	2,200.00
1 Jan 1987	2	3,500.00		5,700.00
1 Jan 1988			570.00	6,270.00
1 Jan 1988	3	2,500.00		8,770.00
1 Jan 1989			877.00	9,647.00
1 Jan 1989	4	1,000.00		10,647.00
1 Jan 1990			1,064.70	11,711.70
1 Jan 1990	5	1,000.00		12,711.70
1 Jan 1991			1,271.17	13,982.87

FLOWCHART


START
PERFORM OPEN-FILE
PERFORM READ-FILE UNTIL EOF = "YES"
CLOSE FILE-A FILE-B
STOP RUN
OPEN-FILE
OPEN INPUT FILE-A OUTPUT FILE-B
MOVE SPACES TO REC-B, ARRAY
DISK READ FILE-A
READ-FILE
PERFORM WRITE-OUT
PERFORM PRO-1
PERFORM PRO-2 THRU PRO-3
PERFORM PRO-3 5 TIMES
PERFORM FINISH-OUT
DISK READFILE-A
false
AT END
true
MOVE "YES" TO EOF
WRITE-OUT
MOVE NAME TO NAME-1
MOVE ZEROS TO SUM-T, NO-1
WRITE REC-B
PRO-2
MOVE SPACES TO L-2
MOVE "1 JAN" TO DAY-1
MOVE YEAR-0 TOYEAR-1
ADD 1 TO NO-1
MOVE NO-1 TO DEP-NO
MOVE MONEY(P) TO DEP
ADD MONEY(P) TO SUM-T
MOVE SUM-T TO BAL
WRITE REC-B
PRO-1
MOVE YEAR TO YEAR-0
MOVE MO-1 TO MONEY(1)
MOVE MO-2 TO MONEY(2)
MOVE MO-3 TO MONEY(3)
MOVE MO-4 TO MONEY(4)
MOVE MO-5 TO MONEY(5)
PRO-3
MOVE SPACES TO L-2
ADD 1 TO YEAR-0
MOVE "1 JAN" TO DAY-1
MOVE YEAR-0TO YEAR-1
COMPUTE I ROUNDED = SUM-T* 1
ADD 1 TO SUM-T
MOVE SUM 1 IQ AI
WRITE REC B
FINISH-OUT
WRITE REC-B

```
1          IDENTIFICATION DIVISION.
2          PROGRAM-ID.  DEPOSITE4.
3          ENVIRONMENT DIVISION.
4          CONFIGURATION SECTION.
5          SOURCE-COMPUTER.  VAX.
6          OBJECT-COMPUTER.  VAX.
7          INPUT-OUTPUT SECTION.
8          FILE-CONTROL.
9              SELECT FILE-A ASSIGN TO "INPUT1.DAT".
10             SELECT FILE-B ASSIGN TO "OUTPUT1.DAT".
11         DATA DIVISION.
12         FILE SECTION.
13         FD  FILE-A LABEL RECORD IS OMITTED DATA RECORD IS REC-A.
14         01  REC-A.
15             02  NAME PIC X(12).
16             02  YEAR PIC 9(4).
17             02  MO-1 PIC 9(4).
18             02  MO-2 PIC 9(4).
19             02  MO-3 PIC 9(4).
20             02  MO-4 PIC 9(4).
21             02  MO-5 PIC 9(4).
22         FD  FILE-B LABEL RECORD IS OMITTED DATA RECORD IS REC-B.
23         01  REC-B.
24             02  FILLER PIC X(132).
25         WORKING-STORAGE SECTION.
26         77  EOF PIC X(3) VALUE "NO".
27         77  YEAR-O PIC 9(4) VALUE ZEROS.
28         77  SUM-T PIC 9(5)V9(2) VALUE ZEROS.
29         77  NO-1  PIC 99 VALUE ZEROS.
30         77  P    PIC 99 VALUE ZEROS.
31         77  I    PIC 9(4)V9(2) VALUE ZEROS.
32         01  ARRAY.
33             02  MONEY PIC 9(4)V9(2) OCCURS 5.
34         01  HEAD-1.
35             02 FILLER PIC X(50) VALUE SPACES.
36             02 FILLER PIC X(16) VALUE " C O L L E G E  FUND O F ".
37             0    2     NAME-1 PIC X(14).
38             02  FILLER  PIC  X(52).
39         01  HEAD-2.
40             02 FILLER PIC X(33) VALUE SPACES.
41             02 FILLER PIC X(12) VALUE "DATE,".
42             02 FILLER PIC X(18) VALUE "DEPOSITE NO ,".
43             "     2       FILLER PIC X(14) VALUE "DEPOSITE".
44             "2 FILLER PIC X(14) VALUE "INTEREST".
45             02 FILLER PIC X(41) VALUE "BALANCE".
46         01  L-1.
47             02 FILLER PIC X(25) VALUE SPACES.
48             02 FILLER PIC X(80) VALUE A  L  L  "*".
49             02  FILLER PIC X(27) VALUE SPACES.
50         01  L-2.
51             02  FILLER PIC X(30).
52             0     2      DAY-1 PIC X(6).
53             0  2    YEAR-1 PIC Z(4).
54             02 FILLER PIC X(10).
55             0 2  DEP-NO PIC ZZ.
56             02 FILLER PIC X(11).
57             02  DEP     PIC Z,Z(3).Z(2).
```

```
58          *02 FILLER PICX(6).
59             02  INT     PIC Z,Z(3).Z(2).
60             0   2    FILLERPICX(5).
61             02  BAL     PIC Z(2),Z(3).Z(2).
62             02 FILLER PICX(40).
63          PROCEDURE DIVISION.
64          STRUCTURE.
65              PERFORM OPEN-FILL.
66              PERFORMREAD-FILEUNTILEOF= " Y E S ".
67              CLOSEFILE-"  FILE-B.
68              STOP RUN.
69          OPEN-FILE.
70              OPEN INPUT FILE-A  OUTPUT FILE-B.
71              MOVESPACES TO REC-B,ARRAY.
72              READFILE-* A? END  MOVE" Y E S " TOEOF.
73          READ-FILE.
74              PERFORM WRITE-OUT.
75              PERFORM PRO-1.
76              PERFORM PRO-2 THRU PRO-3 VARYING P FROM 1 BY 1 UNTIL P > 5.
77              PERFORM PRO-3  5 TIMES.
78              PERFORM FINISH-OUT.
79              READ FILE-A AT END  MOVE "YES" TO EOF.
80          WRITE-OUT.
81              MOVE NAME TO NAME-1.
82              MOVE ZEROS TO SUM-T, NO-2.
83              WRITE REC-B FROM HEAD-1 AFTER PAGE.
84              WRITE REC-B FROM L-1 AFTER ADVANCING 3.
85              WRITE REC-B FROM HEAD-2 AFTER ADVANCING 2.
86              WRITE REC-B FROM L-1 AFTER ADVANCING 2.
87          PRO-1.
88              MOVE YEAR TO YEAR-0.
89              MOVE NO-1 TO MONEY(1).
90              MOVE NO-2 TO MONEY(2).
91              MOVE NO-3 TO MONEY(3).
92              MOVE NO-4 TO MONEY(4).
93              MOVE NO-5 TO MONEY(5).
94          PRO-2.
95              MOVE SPACES TO L-2.
96              MOVE "1 JAN" TO DAY-1.
97              MOVE YEAR-0 TO YEAR-1.
98              ADD 1 TO NO-1.
99              MOVE NO-1 TO DEP-NO.
100             MOVE MONEY(P) TO DEP.
101             ADD MONEY(P) TO SUM-T.
102             MOVE SUM-T TO BAL.
103             WRITE REC-B FROM L-2 AFTER ADVANCING 2.
104         PRO-3.
105             MOVE SPACES TO L-2.
106             ADD 1 TO YEAR-0.
107             MOVE "1 JAN" TO DAY-1.
108             MOVE YEAR-0 TO YEAR-1.
109             COMPUTE 1 ROUNDED = SUM-T * .1.
110             ADD 1 TO SUM-T.
111             MOVE 1 TO INT.
112             MOVE SUM-T TO BAL.
113             WRITEREC-BFROML-2AFTERADVANCING2
114         FINISH-OUT.
```

```
115                 WRITE REC-B FROM L-I AFTER ADVANCING 2.
116
```

HR. SOMCHAI 1966~0003500250010001000
MISS SOMSRI 1986200020002000200020009
HR. SOMPONG 1986100010001000100006000

COLLEGE FUND OF MR. SOMCHAI

DATE	DEPOSITE NO.	DEPOSITE	INTEREST	BALANCE
1 JAN 1986	1	2,000.00		2,000.00
1 JAN 1987			200.00	2,200.00
1 JAN 1987	2	3,500.00		5,700.00
1 JAN 1988			570.00	6,270.00
1 JAN 1988	3	2.500.00		8,770.00
1 JAN 1989			877.00	9.647.00
1 JAN 1989	4	1,000.00		10,647.00
1 JAN 1990			1,064.70	11,711.70
1 JAN 1990	5	1,000.00		12,711.70
1 JAN 1991			1,271.17	13,982.87
1 JAN 1992			1,398.29	15,381.16
1 JAN 1993			1,538.12	16,919.28
1 JAN 1994			1,691.93	18,611.21
1 JAN 1995			1,861.12	20,472.33
1 JAN 1996			2,047.23	22,519.56

COLLEGE FUND OF MISS SOMSRI

DATE	DEPOSITE NO.	DEPOSITE	INTEREST	BALANCE
1 JAN 1986	1	2,000.00		2,000.00
1 JAN 1987			200.00	2,200.00
1 JAN 1987	2	2,000.00		4,200.00
1 JAN 1988			420.00	4,620.00
1 JAN 1988	3	2,000.00		6,620.00
1 JAN 1989			662.00	7,282.00
1 JAN 1989	4	2,000.00		9,282.00
1 JAN 1990			928.20	10,210.20
1 JAN 1990	5	2,000.00		12,210.20
1 JAN 1991			1,221.02	13,431.22
1 JAN 1992			1,343.12	14,774.34
1 JAN 1993			1,477.43	16,251.77
1 JAN 1994			1,625.18	17,876.95
1 JAN 1995			1,787.70	19,664.65
1 JAN 1996			1,966.47	21,631.12

COLLEGE FUND OF MR. SOMPONG

DATE	DEPOSITE NO.	DEPOSITE	INTEREST	BALANCE
1 JAN 1986	1	1,000.00		1,000.00
1 JAN 1987			100.00	lrlOO.OO
1 JAN 1987	2	1,000.00		2,100.00
1 JAN 1988			210.00	2,310.00
1 JAN 1988	3	1,000.00		3,310.00
1 JAN 1989			331.00	3,641.00
1 JAN 1959	4	1,000.00		4,641.00
1 JAN 1990			464.10	5r105.10
1 JAN 1940	5	6,000.00		11,105.10
1 JAN 1991			lrl10.51	12,215.61
1 JAN 1992			1,221.56	13,437.17
1 JAN 1993			1,343.72	14,780.89
1 JAN 1994			1,478.09	lbr258.98
1 JAN 1995			1,625.90	17,884.88
1 JAN 1996			1,788.49	19,673.37

เมื่ออยู่ในรูปของโปรแกรมหลัก และโปรแกรมย่อยจะเขียนดังนี้

```
FUND                                                    25-SEP-1986 14:0
SOURCE LISTING                                          25-SEP-1986 13:4

    1           IDENTIFICATION DIVISION.
    2           PROGRAM-ID.  FUND.
    3           ENVIRONMENT DIVISION.
    4           CONFIGURATION SECTION.
    5           SOURCE-COMPUTER.  VAX.
    6           OBJECT-COMPUTER.  VAX.
    7           INPUT-OUTPUT SECTION.
    8           FILE-CONTROL.
    9               SELECT FILE-A ASSIGN TO "INPUT1.DAT".
   10               SELECT FILE-B ASSIGN TO "OUTPUT5.DAT",
   11           DATA DIVISION.
   12           FILE SECTION.
   13           FD FILE-A LABEL RECORD IS OMITTED DATA RECCRD IS REC-A.
   14           01  REC-A.
   15               02  NAME PIC X(12).
   16               02 YEAR PIC 9(4).
   17               02  MC-1 PIC 9(4).
   18               02  mc-2 PIC 9(4).
   19               02  MC-3 PIC 9(4).
   20               02 NC-4 PIC 9(4).
   21               02 MC-5 PIC 9(4).
   22           FD  FILE-B LABEL RECORD IS OMITTED DATA RECORD IS REC-0.
   23           01  REC-B.
   24               02 FILLER PIC X(132).
   25           WORKING-STORAGE SECTION.
   26           77  ECF PIC X(3) VALUE "NO".
   27           77  NC-1 PIC 99 VALUE ZEROS.
   28           77  P  PIC  99  VALUE  ZEROS.
   29           77  SUM-T PIC 9(5)V9(2) VALUE ZEROS.
   30           77  YEAR-C PIC 9(4) VALUE ZEROS.
   31           01  ARRAY1.
   32               07 MONEY PIC 9(4)V9(2) OCCURS 5.
   13           01  HEAD-1.
   34               02 FILLER PIC X(50) VALUE SPACES.
   35               02  FILLER PIC X(16) VALUE "COLLEGE FUND OF".
   36               0   2     NAME-1 PIC X(14).
   37               02 FILLER PIC X(52).
   38           01  HEAD-2.
   39               02  FILLER PIC X(33) VALUE SPACES.
   40               02 FILLER PIC X(12) VALUE "DATE".
   41               02 FILLER PIC X(18) VALUE "DEPOSITE NO.".
   42               0     2       FILLER PIC X(14) VALUE "DEPOSITE",
   43               02 FILLER PIC X(14) VALUE "INTEREST".
   44               02 FILLER PIC X(41) VALUE "BALANCE".
   45           01  L-1
   46               02 FILLER PIC X(25) VALUE SPACES.
   47               02 FILLER PIC X(80) VALUE ALL "*".
   48               02 FILLER PIC X(27) VALUE SPACES.
   49           01  L-2.
   50               02  FILLER PIC X(30).
    5       1           2       DAY-1 (6).
   52               0     2     YEAR-1 PIC 9(4).
   53               01 FILLER PIC X(10).
   54               02 OFF-NO PIC ZZ.
    5               02 FILLER PIC X(11).
   56               0         2         DEP PIC Z,Z(3).Z(2).
   57               02  FILLER PIC X(6).
```

```
58              02  INT  PIC  Z,Z(3).Z(2).
59              02  FILLER PIC X(5).
60              02  BAL    PIC Z(2),Z(3).Z(2).
61              02  FILLER PIC X(40).
62          PROCEDURE DIVISION.
63          STRUCTURE.
64              PERFORM OPEN-FILE.
65              PERFORM READ-FILE UNTIL EOF = "YES".
66              CLOSE FILE-A  FILE-B.
67              STOP RUN.
68          OPEN-FILL.
69              OPEN  INPUT  FILE-A  OUTPUT FILE-B.
70              MOVE SPACES TO REC-B, ARRAY1.
71              READ FILE-A AT END  MOVE "YES" 10 ECF.
72          READ-FILE.
73              PERFORM WRITE-OUT.
74              PERFORM PRO-I.
75              PERFORM PRO-2 THRU FRO-3 VARYING P FROM 1 BY 1 UNTIL P > 5.
6 i             PERFORM PRO-3
77              PERFORM FINISH-OUT.
78              READ FILE-A AT END MOVE "YES" TO ECF.
79          WRITE-OUT.
80              MOVE NAME TO NAME-1.
81              MOVE ZEROS TO P, NO-I. SUB-T.
82              WRITE REC-B FROM HEAD-1 AFTER PAGE.
83              WRITE REC-B FROM L-1 AFTER ADVANCING ?.
84              WRITE REC-B FROM HEAD-2 AFTER ADVANCING 2.
85              WRITE REC-B FRO!, L-1 AFTER ADVANCING 2.
86          PRO-1.
87              MOVE YEAR TO YEAR-O.
88              MOVE MO-1 TC MONEY(1).
89               CVE PO-2 TC MONEY(2).
90              MOVE "O-3 TC MONEY(3).
91              MOVE PO-', TC MONEY(4).
92              MOVE MO-5 TC MONEY(5).
93          PRO-2.
94              MOVE SPACES TO L-2.
95              MOVE "1 JAN" TO DAY-1.
96              MOVE YEAR-O TO YEAR-1.
97              ADD 1 TO NO-i.
98              MOVE NO-1 TC DEP-NO.
99              MOVE MONEY(P) TO DEP.
100             ACC MONEY(P) TO SUM-T.
101             MOVE SUM-T TO BAL.
102         ** W          *B FROM L-1 AFTER ADVANCING 2.
103         PRO-3.
104             MOVE SPACES TO L-2.
105         ****************************************
106             CALL "SUBCPROG" USING YEAR-O, SUM-T, L-2.
107         ****************************************
108             MOVE "1 JAN" PO DAY-1.
109             WRITE REC-B FRO,, L-2 AFTER ADVANCING 2.
110         FINISH-OUT.
111             WRITE REC-a FROM L-1 AFTER ADVANCING 2.
112
```

SUBCPROG 25-SEP-1986 14:4
'SOURCE LISTING 25-SEP-1986 14:4

```
    1          IDENTIFICATION DIVISION.
    2          PROGRAM-ID.  SUBCPROG.
    3          ENVIRONMENT DIVISION.
    4                SECTION.
    5          COURCE-COMPUTER.VAX.
    6          OBJECT-COMPUTER.  VAX.
    7          DATA CIVISION.
    8          WORKING-STORAGESECTION.
    9          77  I  PIC 9(4)V9(2) VALUE ZEROS.
   10          LINKAGE SECTION.
   11          77  YEAR-C PIC9(4).
   12          77  SUM-T   PIC 9(5)V9(2).
   13          01  L-2.
   14              02  FILLER PIC X(36).
   15              02  YFAR-1 PIC 9(4).
   16              02  FILLER PIC X(37).
   17              02  INT    PIC Z,Z(3).Z(2).
   18              02FILLER PICX(5).
   19              07  BAL    PIC Z(2),Z(3).Z(2).
   20          PROCEDUREDIVISICNUSING YEAR-O,SUM-T,L-Z.
   21          AAA.
   22              ADD 1 TO YEAR-O.
   23              MCVEYEAR-OT  O  YEAR-1.
   24              COMPUTEIROUNDED =SUM-T*.1.
   25              MCVE 1 TO INT.
   26              CCMPUTESUM-T = i +SUN-f.
   27              MCVE SUM-T TO BAL.
   28          BBB.
   29              EXIT  PROGRAM.
```

แบบฝึกหัด

จงเขียน flowchart และโปรแกรมเพื่อสร้างตารางเก็บหมายเลขงานอัตราค่าจ้าง แล้วจึงอ่านข้อมูลตามรูปแบบที่กำหนดให้ คำนวณและพิมพ์ข้อมูลแต่ละเรคคอร์ด, ค่าจ้าง และผลรวมทั้งหมด ค่าจ้างทั้งหมด โดยใช้ข้อมูลตัวอย่างที่กำหนดมานี้

คอลัมน์

1 - 5	หมายเลขพนักงาน
7 - 21	ชื่อพนักงาน
23 - 24	หมายเลขงาน
26 - 28	จำนวนชั่วโมงทำงาน (ทศนิยม 1 ตำแหน่ง)

ตัวอย่าง

11111	ANDERSON	04 105
11111	ANDERSON	01 215
11111	ANDERSON	01 315
11111	ANDERSON	01 105
11111	ANDERSON	04 105
22222	BROWN	0 2 400
22222	BROWN	02 500
22233	CARPENTER	01 994
22233	CARPENTER	04 016
22233	CARPENTER	02 250
22235	DIXON	02 400
22240	EMMETT	01 150
22240	EMMETT	01 150
22240	EMMETT	04 050
22301	FLANDERS	01 255
22301	FLANDERS	01 255
22301	FLANDERS	03 000
22301	FLANDERS	01 050
22355	GAROWSKI	04 015

22355	**GAROWSKI**	04	015
22355	GAROWSKI	04	**015**
22360	HANDERSON	02	080
22360	HANDERSON	02	080
22360	HANDERSON	02	080
22360	HANDERSON	02	080
22360	HANDERSON	**02**	**080**

ข้อมูลข้างต้นนี้เรียงตามลำดับหมายเลขพนักงาน, พนักงานแต่ละคนจะมีข้อมูลกี่เรคคอร์ดก็ได้ ในแต่ละเรคคอร์ด ให้ใช้หมายเลขงานค้นหาอัตราค่าจ้างจากตารางซึ่งกำหนดไว้ดังนี้

ตารางแสดง หมายเลขงานและอัตราค่าจ้าง

Job no.	Rate
01	0725
02	0615
04	0745
05	0855
06	0750
08	0925
10	0895
99	0000

จากนั้นจึงคำนวณค่าจ้างโดยใช้สูตร

ค่าจ้าง = จำนวนชั่วโมงทำงาน * อัตราค่าจ้าง

แล้วพิมพ์ข้อมูลของแต่ละเรคคอร์ด, ผลรวมจำนวนชั่วโมงทำงาน, ผลรวมค่าจ้างของพนักงานคนนั้นและผลรวมทั้งหมด ดังตัวอย่างข้างล่างนี้ (ในกรณีที่มีการบันทึกข้อมูลผิดพลาด เช่น หมายเลขพนักงานไม่เรียงลำดับ, หมายเลขงานผิด, ให้พิมพ์ข้อความบอกไว้โดยไม่ต้องคำนวณค่าจ้างของเรคคอร์ดนั้นและให้ทำการประมวลผลสำหรับเรคคอร์ดต่อไป)

WEEKLY WAGE REPORT

EMPLOYEE	NAME	JOE	RATE	HOURS	WAGE	
11111	ANDERSON	04	7.45	10.5	78.23	
		01	7.25	21.5	155.88	
		01	7.25	31.5	228.38	
		01	1.25	10.5	76.13	
		04	7.45	10.5	78.23	
				84.5	616.85 •	**
22222	BROWN	02	6.15	40.0	246.00	
!		02	6.15	50.0	307.50	
				90.0	553.50	***
22233	CARPENTER	01	7.25	99.9	724.28	
		04	7.45	1.0	7.45CR	
		02	6.15	25.0	153.75	
				123.9	870.58	***
22235	DIXON	02	6.15	40.0	246.00	
				40.0	246.00	***
22240	EMMETT	01	7.25	15.0	108.75	
		01	7.25	15.0	108.75	
		04	7.45	5.0	37.25	
				35.0	254.75 •	⊠●
22301	FLANDERS	01	7.25	25.5	184.88	
		01	7.25	25.5	184.88	
		03	0.00	0.0	0.00	JOB NO.NOT IN TABLE
		01	7.25	5.0	36.25	
				56.0	406.01 •	⊠⊠
22355	GAROWSKI	04	7.45	1.5	11.18	
		04	7.45	1.5	11.18	
		04	7.45	1.5	11.18	
				4.5	33.54	***
22360	"ANDERSON	02	6.15	8.0	49.20	
		02	6.15	8.0	49.20	
		02	6.15	8.0	49.20	
		02	6.15	8.0	49.20	
		02	6.15	8.0	49.20	
				40.0	246.00	***
				473.9	3,227.23	