บทที่ 11
แนวทางในการเขียนโปรแกรม
(Programming Concepts)

วัตถุประสงค์ของบทนี้

- ความหมายของโปรแกรม และความจำเป็นของการสร้างโปรแกรม
- ความสำคัญในการกำหนดคุณลักษณะของโปรแกรม
- การใช้ผังแบบโครงสร้าง (structure chart), ผังโปรแกรม (program flowchart) และโปรแกรมลำลองในการการออกแบบโปรแกรมแบบมีโครงสร้าง
- ความจำเป็นและความสำคัญในการกำหนดรูปแบบที่ดีในการถอดโปรแกรม (coding a program)

เป็นที่ทราบกันดีว่าถ้าปราศจากโปรแกรมแล้ว ระบบคอมพิวเตอร์ก็ไม่สามารถจะทำงานได้ ดังนั้นการเรียนรู้เรื่ององค์ประกอบของคอมพิวเตอร์นั้น เราจึงจำเป็นต้องเรียนรู้เกี่ยวกับเรื่องของการสร้างโปรแกรมขึ้นมาใช้งาน

โปรแกรมคืออะไร

ความหมายของโปรแกรม ก็คือ ชุดของคำสั่งที่เราเขียนเรียงลำดับขึ้นมาเพื่อให้เครื่องคอมพิวเตอร์ดำเนินงานตามวัตถุประสงค์ที่เราต้องการ แนวทางในการสร้างโปรแกรมนั้นจะประกอบด้วยทั้งหมด 5 ขั้นตอน คือ

1. การกำหนดคุณลักษณะของโปรแกรม
2. การนิยามความต้องการของส่วนนำข้อมูลเข้าและส่วนนำข้อมูลออก
3. การออกแบบโปรแกรม
4. การถอดรหัส
5. การตรวจสอบและทดสอบโปรแกรม

ทั้ง 5 ขั้นตอนที่เรียงลำดับมานั้น จะมีเฉพาะในขั้นตอนที่ 4 เท่านั้น ที่ผู้ดำเนินงานจำเป็นจะต้องมีความรู้ความเข้าใจในภาษาที่จะใช้งานตลอดจนการใช้รหัสของภาษาที่เขียนนั้นๆ

การกำหนดคุณลักษณะของโปรแกรม

ระบบงานที่ปรากฏในธุรกิจโดยทั่วไป เช่น ระบบบัญชีเงินเดือน (payroll system) ระบบการขาย (Sale System) ระบบการสั่งของ (mailing list) ล้วนจำเป็นจะต้องมีโปรแกรมที่จะปฏิบัติงานเพื่อรองรับกิจกรรมของแต่ละระบบทั้งสิ้น การที่จะเขียนโปรแกรมขึ้นมาใช้งานนั้น ก่อนที่จะดำเนินการเขียนโปรแกรมขึ้นมานั้น เรามีความจำเป็นที่จะต้องทำความเข้าใจให้ชัดเจน การอธิบายรายละเอียดนั้นไม่ใช่ดำเนินการในลักษณะในเชิงของโปรแกรมเท่านั้น แต่จะต้องทำความเข้าใจในรายละเอียดในความต้องการของระบบ เช่น แหล่งกำเนิดและรายละเอียดของข้อมูล, ขั้นตอนการนำไปประมวลผล ตลอดจนสุดท้ายก็คือได้ข้อมูลนำเสนอออกมาในรูปลักษณะที่ต้องการ

เป็นที่ยอมรับกันว่า ถ้าเราให้ความสนใจและใช้เวลาในการพิจารณาอย่างรอบคอบในการกำหนดคุณลักษณะของโปรแกรม ก็จะส่งผลให้เราได้รับโปรแกรมสำเร็จที่ถูกต้องและมีประสิทธิภาพ ภาพที่ 11-1 จะเป็นตัวอย่างของการกำหนดคุณลักษณะของโปรแกรม ในโปรแกรมบัญชีเงินเดือน โดยจะมีรายละเอียดบอกถึง ชื่อของโปรแกรม ขอบเขตของงานที่

ต้องปฏิบัติ รวมทั้ง รายละเอียดของอุปกรณ์และรายการที่เป็นทั้งข้อมูลนำเข้าและข้อมูลที่เสนอผล โดยปกติแล้วสำหรับระบบงานที่ค่อนข้างจะใหญ่และมีความสลับซับซ้อนแล้ว มักจะต้องอาศัยทีมงานที่ประกอบด้วย ผู้เขียนโปรแกรมหลายคน รวมทั้งนักวิเคราะห์และออกแบบระบบร่วมด้วย นอกจากนี้ยังต้องอาศัยความคิดเห็นของบุคลากรในระบบงานนั้นร่วมด้วย

ภายหลังการกำหนดคุณลักษณะของโปรแกรมเสร็จแล้ว ภารกิจที่จะต้องดำเนินการต่อไปคือการนำไปพัฒนารายละเอียดของส่วนนำข้อมูลเข้า-และส่วนนำข้อมูลออกต่อไป

FIGURE 11-1
Program specifications are developed prior to writing the payroll program.

PROGRAM SPECIFICATIONS

PROGRAM NAME: Payroll — PROGRAM ID: PAY
PREPARED BY: Don Cassel — DATE: January 15, 1989

Program Description:
The payroll program produces a basic pay statement from an input of hours worked and rate of pay.

Input File(s): Keyboard provides hours and rate.

Output File(s): Screen.

Program Requirements:
1. Accept as input the hours worked and rate per hour for an employee.
2. Calculate gross pay by multiplying hours worked by rate per hour.
3. Calculate a basic tax deduction of 15 percent.
4. Determine the net pay by subtracting the tax from the gross.
5. Display all values on the screen.

สาระสังเขปของโปรแกรม :

โปรแกรมระบบเงินเดือนจะทำงานพื้นฐานว่าด้วยรายละเอียดการจ่ายเงินเดือนโดยรับข้อมูลที่ประกอบด้วย จำนวนชั่วโมงทำงาน และอัตราค่าจ้าง

ข้อมูลนำเข้า (Input File) ใช้อุปกรณ์คือ แป้นพิมพ์ โดยป้อนข้อมูลคือ จำนวนชั่วโมงทำงานและอัตราค่าจ้าง

ข้อมูลเสนอผล (Output File) แสดงผลด้วยจอภาพ โดยมีรายการข้อมูลคือ ชื่อพนักงาน เงินเดือนค่าจ้าง ภาษีและรายได้สุทธิ

ความต้องการของโปรแกรม

1. รับข้อมูลของคนงาน 1 คน ซึ่งประกอบด้วยจำนวนชั่วโมงทำงาน, อัตราค่าจ้างต่อชั่วโมง

2. คำนวณรายรับของคนงานแต่ละคนโดยการนำจำนวนชั่วโมงทำงาน คูณด้วยอัตราค่าจ้างต่อชั่วโมง
3. คำนวณหาภาษีที่จะต้องจ่ายคือ 15%
4. คิดรายได้สุทธิโดยการนำค่าจ้างไปหักด้วยภาษี
5. นำข้อมูลที่ได้ในข้อ 4. แสดงออกทางจอภาพ

ข้อมูลนำเข้าและข้อมูลเสนอผล (Input and Output)

กิจกรรมของโปรแกรมนั้นจะประกอบด้วยองค์ประกอบพื้นฐาน 3 ประการ คือ

- ส่วนนำข้อมูลเข้า (input)
- การประมวลผล (process)
- ส่วนนำข้อมูลออก (output)

ส่วนนำข้อมูลเข้า หมายถึงกิจกรรมการส่งข้อมูลเข้าจากอุปกรณ์ประเภทต่างๆ เช่น แป้นพิมพ์ หรืออุปกรณ์อื่นๆที่เห็นว่าเหมาะสม โดยที่ข้อมูลที่รับมาจากส่วนนำข้อมูลเข้านั้น จะถูกจัดการโดยคอมพิวเตอร์ให้เป็นตามวัตถุประสงค์

ข้อมูลที่นำเข้ามานั้นอาจจะเป็นข้อมูลประเภทนิยามของตัวเลข (numeric) หรือไม่เป็นนิยามของตัวเลข (alphanumeric หรือ alphabetic) ก็ได้แล้วแต่การออกแบบและแล้วแต่วัตถุประสงค์ของการใช้งานนั้น

ตาราง 11-1 จะแสดงตัวอย่างของการใช้ข้อมูลในแต่ละกลุ่ม

Type of Data	Example	Value in Field
Numeric	Account no	34522
	Quantity	25
	Cost	47.29
Alphanumeric	Address	37 Main Street
	Date	6/12/89
	Phone	853-233-0189
Alphabetic	Name	John Wilson
	Description	Compact Disk
	Sex code	F

ตาราง 11-1

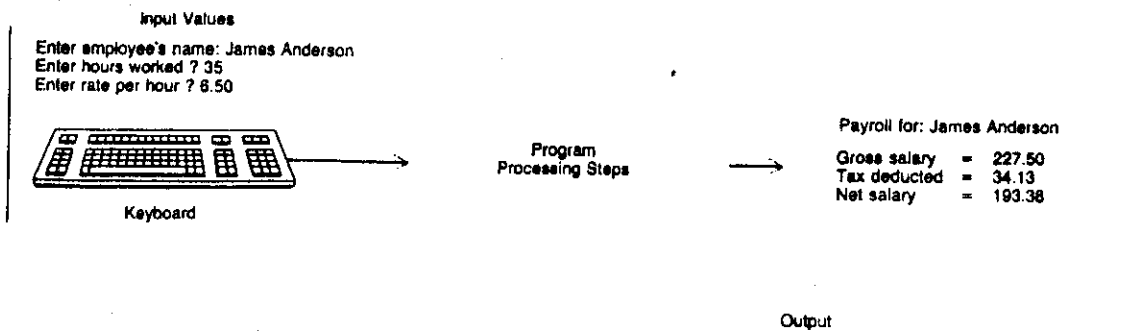
ประเภทข้อมูล	ตัวอย่าง	มูลค่าที่เป็นไปได้
Numeric	หมายเลขบัญชี	34522
	ปริมาณการขาย	25
	ราคาต่อหน่วย	49.50
Alphanumeric	บ้านเลขที่	37 RAMA 1 Rd.
	วันที่	6/7/95
	หมายเลขโทรศัพท์	839-41-23
Alphabetic	ชื่อคนงาน	วิสุนี มากมาย
	รายละเอียด	PC-XT
	รหัสเพศ	F

นอกเหนือจากข้อมูลที่ต้องการดังลักษณะตัวอย่างในตารางแล้ว โปรแกรมลางภาษายังจำเป็นจะต้องรู้จักความยาวของข้อมูลที่จะรับเข้ามาด้วย ในกรณีของข้อมูลที่เป็นตัวเลขนอกเหนือจากขนาดความยาวของข้อมูลแล้วยังจำเป็นที่จะต้องรู้ขนาดตำแหน่งของทศนิยมที่ปรากฏในการใช้งานด้วย การกำหนดรูปแบบของข้อมูลนำเข้านั้นจะมีรายละเอียดแตกต่างกันไปบ้างขึ้นอยู่กับใช้ภาษาโปรแกรมแต่ละภาษา

การนำเสนอผล (Output) นั้น เป็นผลที่เกิดขึ้นอันสืบเนื่องมาจากการนำข้อมูลนำเข้าไปประมวลผล การเสนอผลข้อมูลนั้นเราอาจจะเลือกใช้อุปกรณ์อะไรก็ได้ขึ้นอยู่กับความเหมาะสมของระบบงาน เช่น อาจจะเสนอผลในรูปของจอภาพ ซึ่งเราอาจเรียกว่าแฟ้มสำเนาชั่วคราว (soft copy) หรืออาจจะแสดงผลออกมาในรูปของกระดาษ หรือที่เรียกว่า แฟ้มสำเนาถาวร (hard copy) ในบางกรณีที่ต้องการเก็บผลลัพธ์ไว้ใช้ในคราวต่อไปก็อาจจะบันทึกในตัวกลางที่ใช้ในระบบคอมพิวเตอร์ เช่น เทปแม่เหล็ก (Tape), จานแม่เหล็กชนิดอ่อน (Diskette) หรือ แผ่น ซีดี (CD) แล้วแต่อุปกรณ์ของระบบที่มีอยู่

ภาพ 11-2 จะแสดงขั้นตอนการรับข้อมูลจากแป้นพิมพ์เข้าไปประมวลผลแล้วแสดงออก

ภาพ 11-2 ระบบที่ปรากฏนี้จะรับข้อมูลจากแป้นพิมพ์ไปดำเนินการ แล้วแสดงผลที่จอภาพ



เครื่องมือใช้ในการออกแบบโปรแกรม (The Tools of Program Design)

โปรแกรมพื้นฐานที่ใช้งานกันทั่วๆ ไป เช่น โปรแกรมคิดเงินเดือนดังที่ปรากฏเป็นตัวอย่างนั้น มักจะไม่ค่อยมีปัญหาในการนำไปเขียนโปรแกรม เพราะเราสามารถที่จะเขียนได้โดยอาศัยข้อมูลจากที่ปรากฏในคุณลักษณะที่ปรากฏได้แล้ว (Program Specification) แต่ในกรณีของโปรแกรมที่ค่อนข้างยุ่งยากและมีโครงรูปการทำงานที่สลับซับซ้อนแล้ว เช่นกรณีของการคำนวณเงินเดือนคนงานโดยมีข้อปลีกย่อยของการคำนวณตามข้อกำหนด เช่น ต้องมีการหักค่าสะสม, หักเงินเข้ากองทุนสังคมสงเคราะห์, กรณีของการมีประกันจะคิดเบี้ยเลี้ยงยังชีพจากบริษัทประกัน, การคิดค่าล่วงเวลาในอัตราที่แตกต่าง ปัจจัยทั้งหลายที่กล่าวมานี้ล้วนแต่ทำให้การเขียนโปรแกรมมีรูปแบบที่ยุ่งยาก ซึ่งเราจะใช้ข้อมูลจากคุณลักษณะของโปรแกรม (Program Specification) ไม่เพียงพอเสียแล้ว เราจำเป็นจะต้องใช้เครื่องมืออื่นมาช่วยในการออกแบบเสริมเงื่อนไขเพิ่มเติมดังที่กล่าวมาแล้ว

เครื่องมือใช้ในการออกแบบโปรแกรม จะปรากฏดังนี้

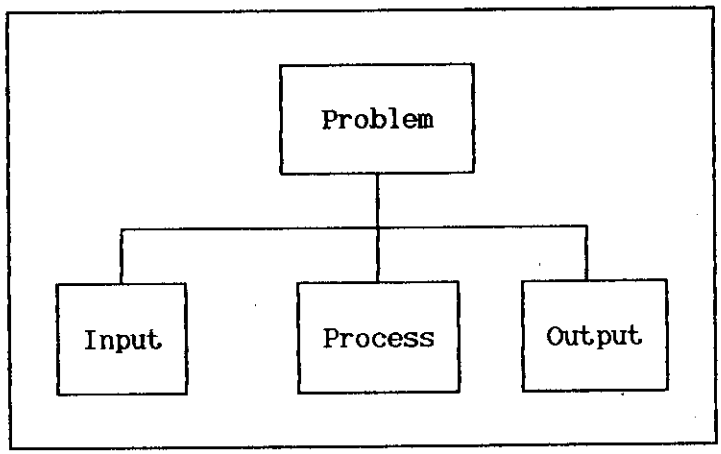
- แผนภูมิแบบโครงสร้าง (Structure Chart)
- ผังโปรแกรม (Program Flowchart)
- โปรแกรมลำลอง (Pseudo Program)

แผนภูมิแบบมีโครงสร้าง (Structure Chart)

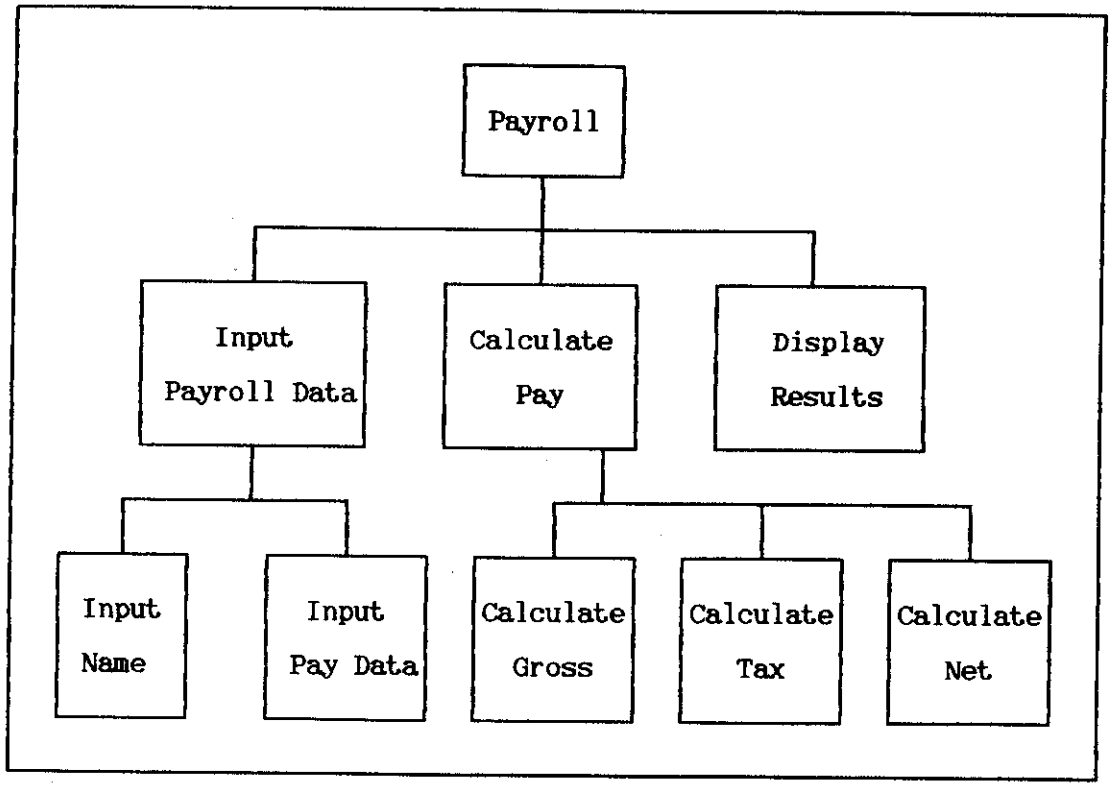
จะเป็นรูปที่ใช้แสดงทิศทางในการแก้ไขปัญหา โดยจะปรากฏในรูปแบบแสดงลำดับขั้นของการแก้ปัญหาจากบนลงล่าง (Top down) ภาพที่ปรากฏ จะแสดงข้อตกลงที่จะต้องดำเนินการ โดยที่ในขั้นลำดับถัดลงมาจะแสดงรายละเอียดของการแก้ปัญหาในแต่ละระดับ เราจะจำแนกแต่ละระดับ แตกรายละเอียดไปเรื่อยๆ จนกระทั่งถึงระดับสุดท้ายที่แตกไปไม่ได้อีกแล้ว

จากรูปที่ปรากฏข้างล่างนี้ เราจะเรียกแต่ละกล่อง (Block) ว่า โมดุล (Module) โดยที่แต่ละโมดุลนั้น หมายถึง กิจกรรมงานที่จะทำหนึ่งกิจกรรม (Task)

ภาพ 11-3 รูปพื้นฐานการเขียนรูปแบบโครงสร้าง



ภาพตัวอย่างแผนภูมิแบบโครงสร้างนั้น จะประกอบด้วยโมดุลนำข้อมูลเข้า, ปฏิบัติการ, และการแสดงผลข้อมูล การใช้รูปนี้เพื่อออกแบบโปรแกรมนั้น ผู้เขียนโปรแกรมจะต้องคำนึงถึงทางแก้ไขปัญหาที่จะปฏิบัติการ โดยจะต้องพิจารณาถึงรูปแบบของข้อตกลงและทิศทางจะปรากฏโดยอาศัยจากโมดุลที่อยู่ระดับบนเป็นสำคัญ แล้วจึงแตกรายละเอียดของแต่ละโมดุลออกให้สอดคล้องกับทิศทางการแก้ปัญหา การแตกโมดุลให้ออกมาเป็นโมดุลย่อยนั้น จะทำให้การนำไปเขียนโปรแกรมง่ายเข้า ให้ดูจากตัวอย่าง เมื่อเรานำภาพ 11-3 มาแตกรายละเอียดออกมา จะได้ภาพโครงสร้างปรากฏในภาพ 11-4 ดังนี้



โปรแกรมลำลอง (Pseudocode)

เครื่องมืออีกรูปแบบหนึ่งที่นิยมใช้กัน ก็คือ การเขียนโปรแกรมลำลอง บางทีเราเรียกเครื่องมือชนิดนี้ว่า **โปรแกรมเทียม** ลักษณะของโปรแกรมลำลองนั้น จะมีรูปแบบคล้ายๆ กับการเขียนโปรแกรมของคอมพิวเตอร์ โดยการเขียนเป็นประโยคในภาษาอังกฤษ แต่ละประโยคนั้น**มักจะไม่คำนึงถึง** รายละเอียด, วากยสัมพันธ์ และไวยากรณ์ของโปรแกรม เพียงแต่จะแสดงทิศทางและกิจกรรมคร่าวๆ ที่จะดำเนินการเท่านั้น ลักษณะของการเขียนโปรแกรมลำลองนั้นมักจะมีเค้าโครงมาจากรูปแบบของโครงสร้างแบบที่ 1 ผู้เขียนโปรแกรมบางคนจะออกแบบโปรแกรมโดยใช้แนวคิดของการออกแบบโมดุลแบบมีโครงสร้างเสร็จแล้วจึงนำภาพนั้นไปเขียนโปรแกรมลำลองต่อไป ก่อนที่จะใช้ภาษาคอมพิวเตอร์จริงๆ ในการเขียนโปรแกรม

คุณลักษณะของโปรแกรมลำลองที่ปรากฏก็คือ จะเป็นโปรแกรมที่แสดงถึงขั้นตอนการดำเนินงานและตรรก (logic) ของการทำงานตลอดจนครอบคลุมถึงการตัดสินใจในแต่ละขั้นของการดำเนินงานในโปรแกรม รวมถึงการทำงานประเภท**วงวน** (looping) ซึ่งหมายถึง

การทำงานซ้ำๆ ภายในข้อกำหนด

ภาพ 11-5 จะแสดงถึงตัวอย่างของการเขียนโปรแกรมลำลองในระบบโปรแกรมคิดเงินเดือน ในภาพจะเห็นกรณีของการวนเวียนทำงานภายใต้กำหนดเพื่อให้โปรแกรมดังกล่าวทำงานกับคนงานที่มีมากกว่า 1 คน ในรูปแบบดังกล่าว จะเรียกว่า "DOWHILE" โดยที่เงื่อนไขของ DOWHILE นั้น ผู้เขียนโปรแกรมจะต้องไปเพิ่มรายละเอียดว่า การกำหนดว่า สิ้นสุดการคิดเงินเดือนของคนงาน จะสามารถกระทำได้โดยรูปแบบใดบ้างที่เหมาะสม

ภาพ 11-5

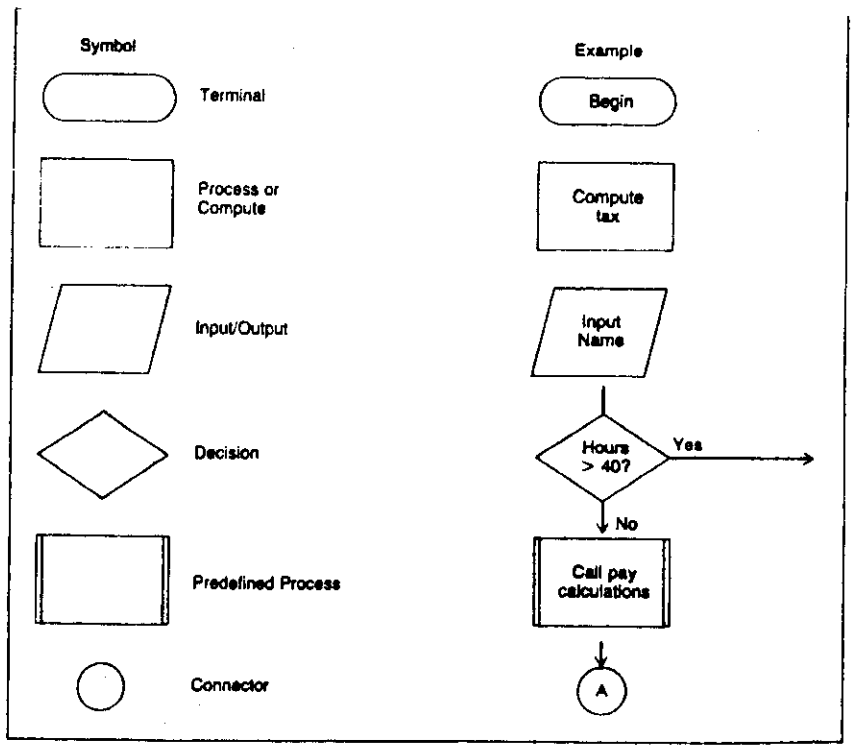
```
Program: Payroll

DOWHILE  another employee
     INPUT  employee's name
     INPUT  hours and rate
     Compute  gross
     Compute  tax at 15% of gross
     Compute  net
ENDDO
```

ผังโปรแกรม (Program Flowcharts)

ในขณะที่โปรแกรมลำลองเป็นการออกแบบโดยใช้รูปแบบการเขียนเลียนแบบโปรแกรม เราอาจจะใช้เครื่องมืออีกชนิดหนึ่งที่เรียกว่า ผังโปรแกรม โดยที่ผังโปรแกรมนั้นจะใช้รูปภาพแสดงขั้นตอนของการทำงานตลอดจนตรรก (logic) ในการทำงาน ผู้เขียนแต่ละคนอาจจะมีความนิยมในการใช้เครื่องมือเหล่านี้ในการออกแบบโปรแกรมแตกต่างกันไป บางคนก็เคยชินกับการใช้ผังโปรแกรม ในขณะที่บางคนชอบรูปแบบของโปรแกรมแบบลำลอง ถึงแม้ว่าเครื่องมือเหล่านี้จะแตกต่างกันไปตามรูปลักษณ์แต่เจตนาจะตรงกัน

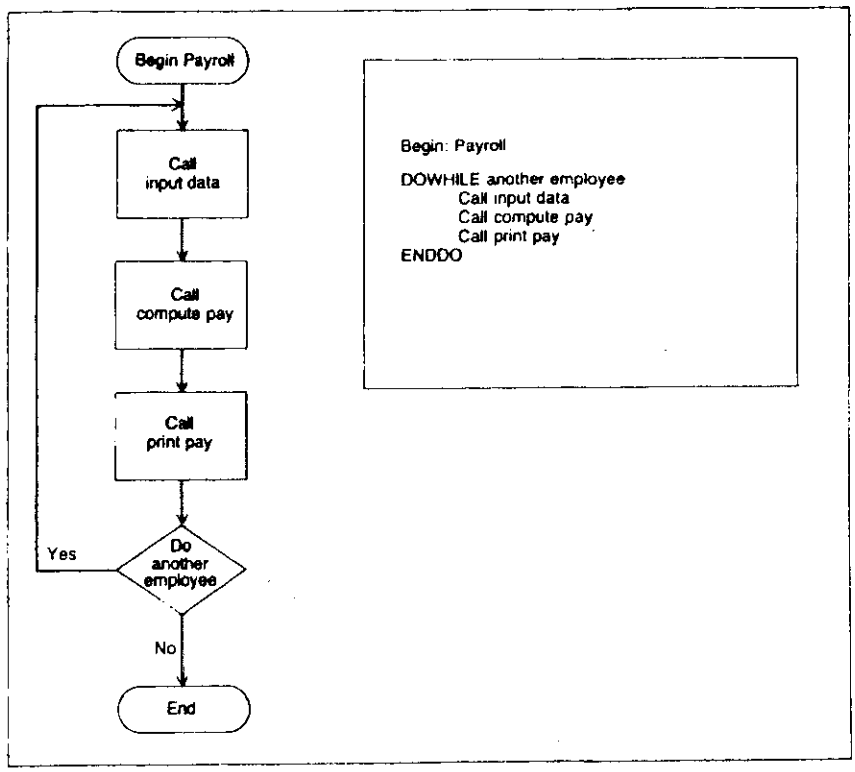
การใช้รูปภาพสัญลักษณ์ต่างๆ ในผังโปรแกรม สัญลักษณ์ที่ปรากฏในภาพที่ 11-6 นั้น จะแสดงถึงความหมายของการนำไปใช้งานในการออกแบบ แต่ละรูปจะเชื่อมต่อกันโดยใช้

FIGURE 11–6
Symbols used for creating program flowcharts.



ภาพ 11–7 จะแสดงถึงการออกแบบโดยใช้ผังโปรแกรมและเปรียบเทียบกับโปรแกรมลำลอง

FIGURE 11-7
Flowchart and pseudocode for the payroll problem. The mainline chart provides the calls to access subroutines for reading data, doing the payroll calculations, and printing the results. The mainline also provides a loop so that additional employees may be processed if required.



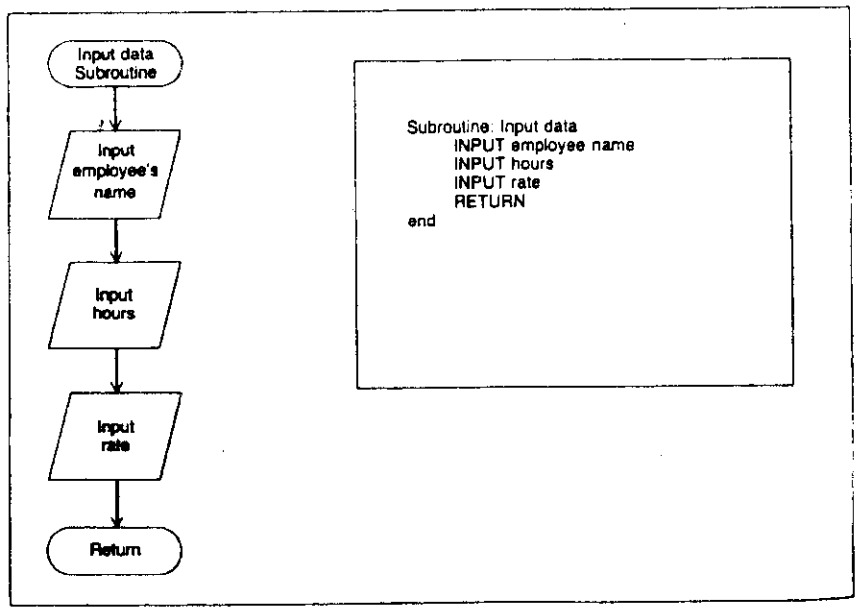
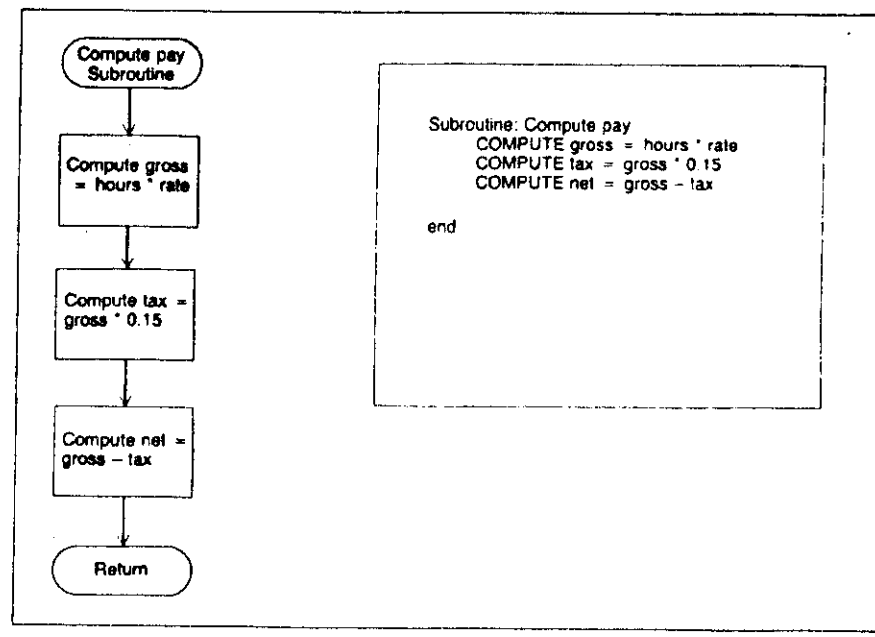
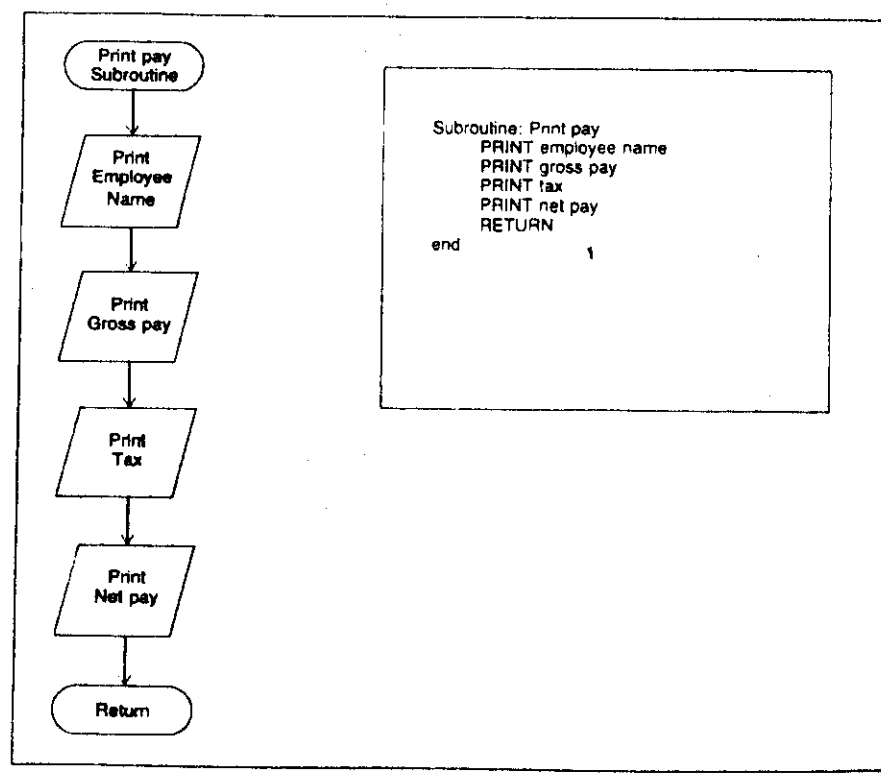


FIGURE 11–7
(Continued)





จากภาพที่ 11-7 นั้น ในสัญญลักษณ์ของ terminal ที่ขั้นต้นนั้นจะปรากฏข้อความ main line ซึ่งเราจะเรียก อีกนัยหนึ่งได้ว่าเป็น **driver module** โดยที่ module ดังกล่าวจะมีหน้าที่ควบคุมกิจกรรมของโปรแกรม และทำหน้าที่เรียกติดต่อกับโปรแกรมย่อย Subroutine) อื่นที่เรียกมาปฏิบัติงานรับใช้ต่อไป ภายหลังเมื่อโปรแกรมย่อย (subroutine) ที่ถูกเรียกใช้นั้นดำเนินงานเสร็จก็จะส่งผลที่ได้มาให้กับ driver module ต่อไป

การออกแบบโปรแกรมแบบมีโครงสร้าง (Control Structure For Programs)

การออกแบบโปรแกรมที่ดีนั้น ปกติจะมีแนวคิดให้ยึดถือบนพื้นฐานของการออกแบบโปรแกรมแบบที่มีโครงสร้าง ในสมัยก่อนจะมีการกำหนดเรื่อง "การออกแบบแบบมีโครงสร้าง" (Structure Programming) นั้น ผู้เขียนโปรแกรมแต่ละคนจะออกแบบโปรแกรมแต่ละคนจะออกแบบโปรแกรมตามแบบอิสระตามใจชอบของตนเอง จึงทำให้เกิดข้อเสียของการไม่ "Well design" ซึ่งเป็นจุดบอดของการเขียนโปรแกรมและพัฒนาโปรแกรมต่อไปในอนาคต ด้วยเหตุนี้เองจึงได้มีการพัฒนาและสร้างต้วแบบแบบมีโครงสร้างขึ้นเพื่อลบข้อเสียดังกล่าว

ตัวแบบโปรแกรมแบบมีโครงสร้างนั้น เราสามารถแบ่งได้เป็น 3 ประเภทคือ

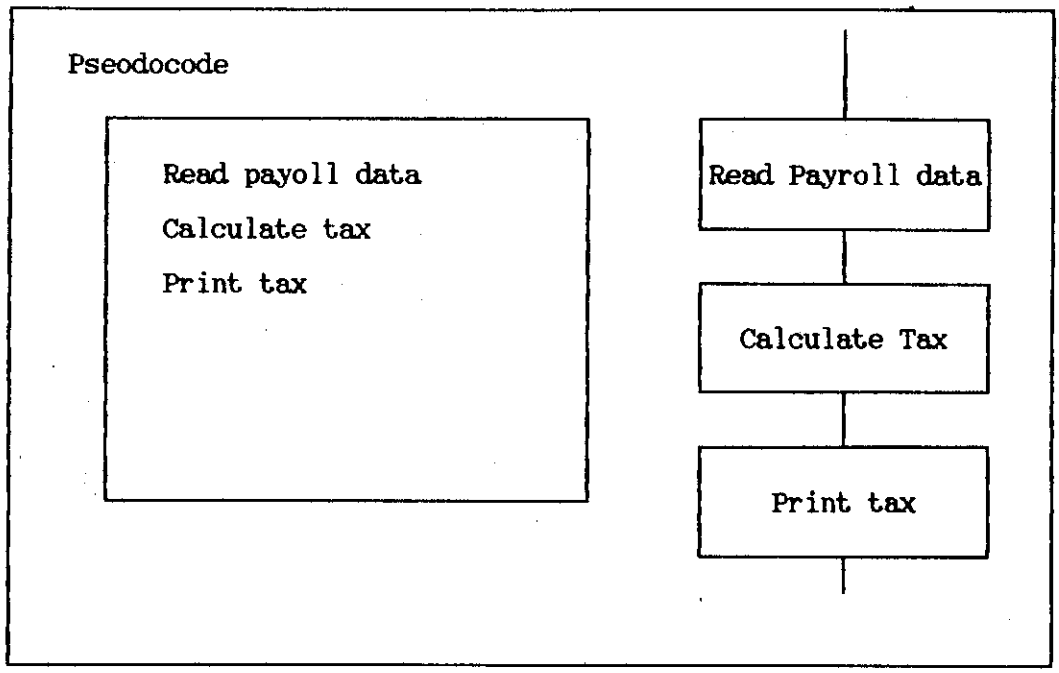
1. รูปแบบทีละลำดับ (Sequence)
2. รูปแบบการเลือก (Selection)
3. รูปแบบการซ้ำและการวน (Repetition)

รูปแบบของโปรแกรมแบบมีโครงสร้างนี้ ไม่ใช่ใช้เฉพาะในการเขียนโปรแกรมด้วยภาษาคอมพิวเตอร์เท่านั้น แต่ยังส่งผลถึงการใช้ในการเขียนโปรแกรมแบบลำลอง และการออกแบบโดยใช้ผังโปรแกรมด้วย

รูปแบบทีละลำดับ (Sequence Structure)

รูปแบบทีละลำดับเป็นรูปแบบที่ง่ายที่สุดในการอธิบายถึง กิจกรรมทำงานที่ไปทีละลำดับ เช่น reading, printing, calculating และแม้แต่การเรียก subroutine

รูปแบบทีละลำดับ



จากรูปที่ปรากฏ จะเปรียบเทียบการเขียนโดยใช้โปรแกรมลำลอง และการใช้ผังโปรแกรม ในกิจกรรมที่เรียกว่า ทีละลำดับ

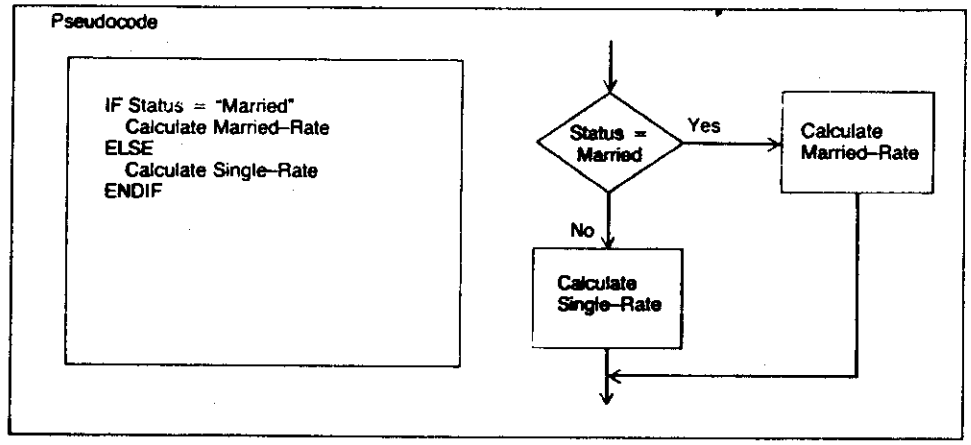
รูปแบบการเลือก (Selection Structure)

โดยปกติแล้วในระบบงานใดๆ ก็ตามล้วนแต่จะต้องมีขั้นตอนการตัดสินใจเป็นองค์ประกอบขั้นพื้นฐานปรากฏอยู่ด้วยเสมอ การจะต้องมีการตัดสินใจนั้น เราจะเขียนโปรแกรมลำลองในรูปแบบของ IF statement

ดังนั้นโปรแกรมีโครงสร้าง จะมีรูปแบบของการตัดสินใจดังนี้คือ

```
Pseudocode
   IF  Status = 'Married'
       Calculate  Married-rate
   ELSE
       Calculate  Single-rate
  ENDIF
```

Selection structure.



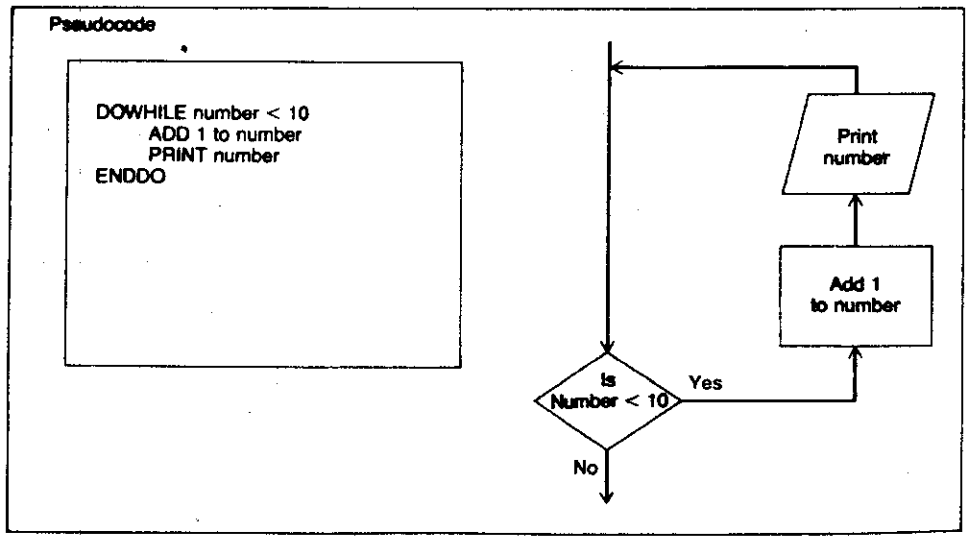
จะเห็นได้ว่าการตัดสินใจนั้นหมายความว่าเลือกกระทำเพียงกิจกรรมเดียว โดยที่กิจกรรมนั้นจะสอดคล้องกับเงื่อนไขที่ตรวจสอบ

รูปแบบการทำซ้ำ (Repetition Structure)

การทำงานซ้ำๆ หรือที่เรียกว่า การวน (loop) ก็เป็นรูปแบบที่ส่วนหนึ่งอาศัยพื้นฐานของการตัดสินใจ แต่จะมีข้อแตกต่างกับรูปแบบการตัดสินใจตรงที่ว่า ภายหลังการตัดสินใจตามเงื่อนไขใดเงื่อนไขหนึ่งแล้วนั้น ถ้าตรงกับเงื่อนไขก็ยังคงเวียนทำงานภายใต้เงื่อนไขนั้นจนกว่าเงื่อนไขการตัดสินใจจะเป็นเท็จจึงจะหยุดดำเนินกิจกรรมที่วนทำนั้น

Pseudocode

```
DOWHILE  number < 10
         ADD 1 to number
         PRINT number
ENDDO
```



การถอดรหัส (Program Coding)

ภายหลังเมื่อออกแบบโปรแกรมโดยใช้เครื่องมือชนิดใดชนิดหนึ่งที่คิดว่าเหมาะสมแล้ว จะมีการตรวจสอบจนกว่าจะแน่ใจว่าไม่มีข้อบกพร่องแล้ว ขั้นตอนต่อไปก็จะดำเนินการนำภาพที่ออกแบบนั้นมาถอดรหัสต่อไป

การออกแบบโปรแกรมนั้น ไม่ใช่มีเจตนาเพียงแต่จะเป็นเครื่องมือในการถอดรหัสเท่านั้น แต่ยังจะต้องคำนึงถึงการใช้เป็นเครื่องมือในการค้นหาข้อผิดพลาดและการทดสอบโปรแกรม (Debuging and testing) ในอนาคตอีกด้วย ตัวอย่างต่อไปนี้เป็นตัวอย่างโปรแกรมที่เขียนขึ้นด้วยภาษาเบสิคโดยใช้ตัวอย่างจากงานของโปรแกรมคำนวณเงินเดือน ที่เคยยกตัวอย่างมาแล้ว

FIGURE 11-8
A BASIC program using good techniques of style to implement the payroll application.

```
100  REM ************************************************************************
110  REM *         Sample Payroll Program                                       *
120  REM * Description:                                                        *
130  REM *   This program reads hours and pay rate data for an employee *
140  REM *   and calculates a gross salary. A 15% tax deduction is        *
150  REM *   computed and the net salary to be paid. After the results    *
160  REM *   are printed the user is allowed to repeat the process.       *
170  REM * Programmer:                                                         *
180  REM *   Don Cassel                                                       *
190  REM ************************************************************************
200  Y$ = "yes"
210  WHILE Y$ = "yes"
215       CLS
220       GOSUB 1000
230       GOSUB 2000
240       GOSUB 3000
250       INPUT "Input another employee (yes/no)";Y$
260  WEND
270  END
1000 REM ************************************************************************
1010 REM *                Input Payroll Data Subroutine                         *
1020 REM ************************************************************************
1030 REM
1040 INPUT "Enter employee's name ",EMP$
1050 INPUT "Enter hours worked ";HOURS
1060 INPUT "Enter rate per hour ";RATE
1070 PRINT
1080 RETURN
2000 REM ************************************************************************
2010 REM *                   Compute Pay Subroutine                             *
2020 REM ************************************************************************
2030 REM
2040 GROSS = HOURS * RATE
2050 TAX = GROSS * .15
2060 NET = GROSS - TAX
2070 RETURN
3000 REM ************************************************************************
3010 REM *                   Print Payroll Subroutine                           *
3020 REM ************************************************************************
3030 REM
3040 PRINT "Payroll for: ";EMP$
3050 PRINT
3060 PRINT "Gross salary = ";GROSS
3070 PRINT "Tax deducted = ";TAX
3080 PRINT "Net salary   = ";NET
3090 PRINT
3100 RETURN
```

จากตัวอย่างของโปรแกรมที่เขียนนี้จะเห็นได้ว่าเป็นการเขียนโปรแกรมที่เป็นมาตรฐาน ทั้งนี้เพราะโปรแกรมดังกล่าวประกอบด้วยองค์ประกอบต่อไปนี้

1. ในคำสั่งแรกจะเป็นคำสั่งไม่ปฏิบัติการ ที่เราเรียกว่า **Comment statement** โดยในคำสั่งนี้จะเป็นการบอกข้อมูลว่า โปรแกรมมีชื่อว่าอะไร ทำงานอะไรบ้าง ตลอดจนใครเป็นผู้เขียนโปรแกรม

2. ในแต่ละคำสั่งเริ่มต้นของโปรแกรมย่อย (Subroutine) จะมีคำสั่งไม่ปฏิบัติงานที่เรียกว่าหมายเหตุเพื่อจะอธิบายผู้อ่านโปรแกรมเข้าใจว่าโปรแกรมย่อยๆ นั้นมีเจตนาของการทำงานเป็นเช่นใด

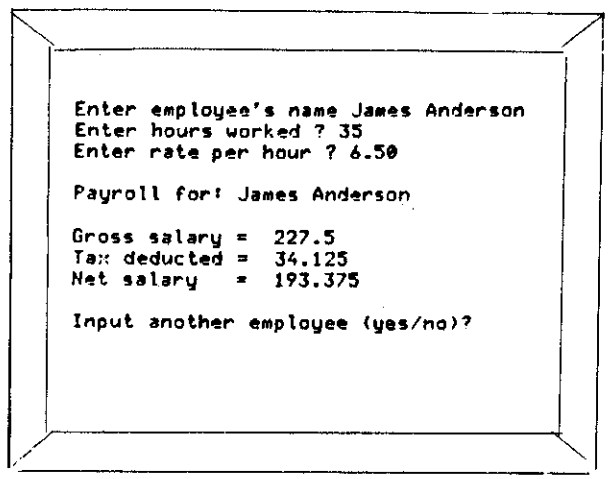
3. การใช้ชื่อตัวแปรเก็บข้อมูลภายในสมองของคอมพิวเตอร์นั้น แทนที่จะใช้ตัวอักษรตัวเดียวที่ไร้ความหมาย ก็จะแทนด้วยชื่อย่อที่เดาได้ว่าจะเป็นที่เก็บข้อมูลในเรื่องอะไร ตัวอย่างเช่น HOURS จะหมายถึงข้อมูลคือชั่วโมงการทำงานของคนงาน และ EMP$ จะหมายถึง ชื่อของคนงาน ดังนี้เป็นต้น

4. ในกรณีที่มีการใช้ วน (loop) จะเห็นได้ว่า มีการเยื้องตำแหน่งกันเพื่อให้ง่ายแก่การสังเกตถึงขอบเขตของการครอบคลุมในแต่ละวังวนนั้น

5. ในกรณีที่มีนิพจน์คณิตศาสตร์ปรากฏหลายองค์ประกอบ ก็ให้ใช้วงเล็บเล็กแบ่งการทำงานทั้งนี้เพื่อให้เข้าใจได้อย่างถูกต้องและง่าย (เครื่องหมายคณิตศาสตร์เหล่านี้ ที่ปรากฏในการใช้งานบ่อยๆ เช่น +, -, *, /)

ภาพ 11-9 จะแสดงถึงผลของการดำเนินงานในโปรแกรมภาพ 11-8 ซึ่งผลออกมาทางจอภาพ จะสังเกตได้ว่าสามบรรทัดแรกในจอภาพจะช่วยนำทางให้ผู้ป้อนข้อมูล ได้ป้อนข้อมูลได้อย่างถูกต้องตามที่โปรแกรมต้องการ ส่วนบรรทัดถัดๆ มาจะเป็นผลจากการดำเนินงานของโปรแกรมแสดงผลลัพธ์จากการทำงาน

FIGURE 11-9
A screen display from the payroll program.



ภาษาโปรแกรม (Programming Language)

การถอดรหัสนั้น ถือเป็นขั้นตอนของการเขียนโปรแกรมสำหรับคอมพิวเตอร์ โดยปกติผู้เขียนโปรแกรมมักจะถอดรหัสออกมาเป็นภาษาคอมพิวเตอร์ภาษาใดภาษาหนึ่งที่เหมาะสม โดยการ

เขียนโปรแกรมนั้นลงในกระดาษหรืออาจจะใช้โปรแกรมประมวลคำเข้ามาช่วยก็ได้

โปรแกรมที่เราเขียนเสร็จแล้วนั้น อาจจะยังมีข้อผิดพลาดปรากฏอยู่ โดยที่ข้อผิดพลาดนั้นอาจจะมาจากสาเหตุดังนี้ คือ

ประเภทที่ 1 ความผิดพลาดด้านไวยากรณ์ภาษา (Syntax Error)

ประเภทที่ 2 ความผิดพลาดทางความหมาย (Logic or Semantic Error)

ประเภทที่ 3 ความผิดพลาดจากการทำงาน (Run Time Error)

ความผิดพลาดประเภทที่ 1 นั้นโดยปกติแล้วตัวแปลภาษาของแต่ละภาษานั้นจะช่วยตรวจสอบในส่วนนี้อยู่แล้ว ผู้ถอดรหัสจึงไม่ค่อยมีปัญหาเท่าไร แต่ความผิดในกรณีที่ 2 และ 3 นั้น จำเป็นจะต้องมีการตรวจสอบเองโดยผู้ออกแบบและผู้เขียนโปรแกรมเพื่อดูว่ามีจุดบกพร่องใดบ้างที่จะต้องแก้ไข

ภาษาคอมพิวเตอร์ ไม่ว่าจะเป็นภาษาอะไรก็ตาม จำเป็นจะต้องใช้โปรแกรมล่ามเพื่อทำหน้าที่ในการแปลโปรแกรมซึ่งกล่าวนั้นให้เป็นภาษาเครื่องต่อไป

โปรแกรมทำหน้าที่ในการแปลภาษาที่เราใช้เพื่อให้เครื่องนำไปทำงานตามขั้นตอนต่อไปนี้ เราอาจเรียกว่า **โปรแกรมล่าม** โดยที่โปรแกรมล่ามนั้น เราจะแบ่งออกมาได้เป็น 2 ประเภทคือ

1. Interpreter หมายถึง โปรแกรมชนิดหนึ่งที่ทำการอ่านคำสั่งแต่ละคำสั่งของโปรแกรมนั้นไปดำเนินการแปลง ให้เป็นคำสั่งที่คอมพิวเตอร์สามารถปฏิบัติงานได้เลย ตัวอย่างของภาษาบางภาษาที่มี Interpreter ใช้งาน เช่น ภาษาเบสิค โดยหลักการการแปลคำสั่งโดยใช้ Interpreter ก็เปรียบเสมือนล่ามฝรั่งเศสที่แปลภาษาฝรั่งเศสมาให้เราฟังเป็นภาษาไทยทีละประโยคจนกว่าจบความ การแปลโดยวิธีนี้นั้น เราจะ**ไม่มี** object program ปรากฏ (โดยที่ object program ก็คือ โปรแกรมผ่านการแปลแล้วแต่เป็นรูปแบบของเลขฐานสอง ที่เราเรียกว่าภาษาเครื่องนั่นเอง)

2. Compiler เป็นโปรแกรมแปลภาษา แต่ละภาษา แต่ละรุ่นจะมี Compiler แยกเป็นเอกเทศไปต่างหาก การใช้ Compiler แปลโปรแกรมนั้น ถ้าอุปมาเปรียบเสมือนล่ามที่ทำหน้าที่แปลภาษา โดยทั่วไปแล้ว ล่ามจะแปลทุกประโยคออกมาในคราวเดียว แทนที่จะแปลทีละประโยคเช่นเดียวกับ Interpreter การใช้ Compiler แปลโปรแกรมนั้นเราจะได้ Object Program ซึ่งจะต้องไปผ่านขั้นตอนอื่นต่อไปถึงจะปฏิบัติงานได้ ซึ่งแตกต่างจากการแปลคำสั่งโดย Interpreter ซึ่งเครื่องจะปฏิบัติงานได้ทันที โดยปกติแล้วประสิทธิภาพการทำงานของโปรแกรมที่ใช้วิธีการของการแปลโดย Compiler จะมีประสิทธิภาพสูง

กว่าโปรแกรมที่ใช้วิธีการของการแปลโดย Interpreter

โปรแกรมที่เขียนด้วยภาษาคอมพิวเตอร์อย่างเช่น COBOL, C เราจะเรียกว่า Source Program ในขณะที่โปรแกรมที่ผ่านการแปลแล้วเราจะเรียกว่า Object Program

ภายหลังการแปล Source Program นั้น Compiler หรือ Interpreter จะตรวจสอบข้อผิดพลาดจากโปรแกรมและพิมพ์รายงานคำสั่งที่ผิดพลาดออกมา การค้นหาข้อผิดพลาดนี้ เราจะเรียกว่า การ debugging โปรแกรม ภายหลังที่ทำการแก้ไขข้อผิดพลาดของโปรแกรมแล้วโปรแกรมดังกล่าวจะถูกนำมาทดสอบ (testing) กระบวนการทดสอบนี้จะรวมความถึงการส่งข้อมูลเข้าไปให้โปรแกรมลองปฏิบัติการดูว่าได้ผลลัพธ์ตรงตามเป้าหมายที่เราต้องการหรือไม่ การกำหนดข้อมูลเพื่อให้โปรแกรมทดสอบนั้น จะประกอบด้วยข้อมูลที่เป็นตัวแทนทุกกรณีที่เป็นไปได้ รวมถึงข้อมูลที่ผิดพลาดเพื่อให้โปรแกรมปฏิเสธการปฏิบัติงานอย่างถูกต้องด้วย ปกติแล้วการทดสอบมักจะใช้เวลามากถ้าหากว่าเป็นโปรแกรมที่ใหญ่และซับซ้อน การทดสอบโปรแกรมนั้นจัดว่าเป็นศิลปะ ซึ่งขึ้นอยู่กับประสบการณ์ของผู้เขียนโปรแกรมแต่ละคนที่จะดำเนินงาน

ในเรื่องของภาษาคอมพิวเตอร์นั้น ได้มีวิวัฒนาการมาตั้งแต่ ภาษาเครื่อง (machine language) มาเป็น ภาษามีความหมาย (symbolic language) และมาเป็นภาษาชั้นสูง (high level language) เช่น COBOL จนกระทั่งปัจจุบันนี้เรามีภาษายุคที่ 4 (fourth generation language) ใช้งาน เราอาจนิยามได้ว่า ภาษาในยุคที่ 3 เช่น COBOL, PASCAL นั้นจัดว่าเป็น procedural language และภาษายุคที่ 4 จัดว่าเป็น nonprocedural language

Procedural Language

ภาษาที่ใช้กันอยู่ทั่วไป คือ BASIC, COBOL, FORTRAN และ Pascal นั้น เราเรียกว่า ภาษาที่เป็น Procedural language ทั้งนี้เนื่องจากเหตุผลที่ว่าภาษาเหล่านี้จะมีการเขียนเรียงลำดับตามด้วยการปฏิบัติงานที่ต้องการ โดยที่ภายในองค์ประกอบของโปรแกรมที่เขียนนั้น จะปรากฏคำสั่งการตัดสินใจและคำสั่งการทำงานซ้ำๆ ตามเงื่อนไขที่เรากำหนดขึ้น ภาษากลุ่มนี้นับว่าใช้กันแพร่หลายมากบนเครื่องหลายตระกูล รวมถึงเครื่องไมโครคอมพิวเตอร์ด้วย

ภาษาเบสิค (BASIC)

ความหมายของภาษาเบสิคมาจากคำว่า **Beginners All-Purpose Symbolic Instruction Code** ภาษานี้พัฒนาขึ้นที่ Dartmouth College โดยใช้บนเครื่อง mainframe ภายใต้ระบบ Time Sharing ภาษาเบสิคนี้นับว่าเป็นภาษาที่ใช้กันแพร่หลายมากในตระกูลของเครื่องไมโครคอมพิวเตอร์ เช่น Apple, Commadore และ IBM PC หรือ IBM Compatible

ลักษณะเด่นของภาษาเบสิค ก็คือมีรูปแบบเป็น โต้ตอบ (interactive) การใช้คำสั่งควบคุมอุปกรณ์คือ แป้นพิมพ์และจอภาพก็เป็นไปได้อย่างง่ายดาย ภาษาเบสิคนั้นได้พัฒนาและแตกออกมาเป็นหลายตระกูลด้วยกัน โดยใช้กับเครื่องต่างตระกูลกัน ถึงแม้ว่าภาษานี้จะมีจุดเด่นดังที่กล่าวมาแล้ว แต่ภาษานี้ก็ยังมีข้อจำกัดในการใช้อยู่หลายประการ จึงทำให้ผู้เขียนโปรแกรมหันไปใช้ภาษาอื่นๆ ที่เพิ่งพัฒนาสำเร็จมาใช้แทน อาทิเช่น ภาษา C หรือ Pascal เป็นต้น

ภาษาโคบอล (COBOL)

คำว่า COBOL ย่อมาจากคำว่า **Common Business-Oriented Language** โดยมีเจตนาที่จะพัฒนามาใช้กับระบบงานทางด้านธุรกิจ ภาษานี้ได้พัฒนาสร้างขึ้น ในปี ค.ศ. 1959 และประสบผลสำเร็จนำมาใช้ในปี ค.ศ.1960

ภาษาโคบอลนี้ มีจุดเด่นในแง่ของการจัดการกับแฟ้มข้อมูลในหลากหลายรูปแบบ เช่น แฟ้มแบบเรียงลำดับ (Sequential File) แฟ้มแบบ ISAM หรือ VSAM ภาษานี้ไม่เหมาะสำหรับระบบงานที่เป็นประเภทโต้ตอบ ในสมัยก่อนนั้นเราไม่สามารถจะใช้งานภาษานี้บนเครื่องไมโครคอมพิวเตอร์ได้ แต่ปัจจุบันได้มีการพัฒนาภาษาโคบอลให้ดำเนินงานได้บนเครื่องไมโครคอมพิวเตอร์

ภาพที่ 11-10 จะเป็นตัวอย่างการเขียนโปรแกรมโดยใช้ภาษาโคบอล ในตัวอย่างดังกล่าวจะแสดงทั้งจุดเด่นและจุดด้อยของภาษานี้ จุดเด่นของภาษานี้ก็คือ อ่านเข้าใจง่ายเพราะมีรูปแบบคล้ายภาษาอังกฤษ ถึงแม้ว่าผู้อ่านจะไม่มีความรู้เกี่ยวกับภาษานี้เลย ก็อ่านแล้วเข้าใจได้ ในส่วนที่เป็นจุดอ่อนของภาษาโคบอล ก็คือ ภาษานี้ประกอบด้วยคำที่สงวนไว้มากมาย และคำสั่งในการเขียนก็ค่อนข้างจะยาว ถึงแม้จะเป็นโปรแกรมที่ทำงานเล็กๆ ก็ตาม ถ้าเทียบกับภาษาเบสิคแล้วภาษาเบสิคจะสั้นกว่ามาก

FIGURE 11–10
A sample COBOL program.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.
     EXP01.
*REMARKS.
*    PRINTS DEPARTMENTAL EXPENSE REPORT.

 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.
     IBM-370-138.
 OBJECT-COMPUTER.
     IBM-370-138.
 SPECIAL-NAMES.
     C01 IS TO-NEW-PAGE.

 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT EXP-IN  ASSIGN TO SYS004-UR-2501-S.
     SELECT EXP-RPT ASSIGN TO SYS005-UR-1403-S.

 DATA DIVISION.

 FILE SECTION.
 FD  EXP-IN
     RECORD CONTAINS 80 CHARACTERS
     LABEL RECORDS ARE OMITTED
     DATA RECORD IS IN-REC.
 01  IN-REC.
     05  IN-DEPT      PIC 9(03).
     05  IN-DATE      PIC X(08).
     05  IN-TYPE      PIC X(13).
     05  IN-AMOUNT    PIC 9999V99.
     05  FILLER       PIC X(50).

 FD   EXP-RPT
      RECORD CONTAINS 133 CHARACTERS
      LABEL RECORDS ARE OMITTED
      DATA RECORD IS OUT-REC.
 01   OUT-REC.
      05  FILLER     PIC X(133).

 WORKING-STORAGE SECTION.
 01 WORK-AREA.
     05 EOF-FLAG    PIC 9       VALUE ZERO.
     05 WORK-TOTAL PIC 9(5)V99 VALUE ZERO.

 01  OUT-HEAD-1.
     05 FILLER      PIC X(015) VALUE SPACES.
     05 FILLER      PIC X(118)
                    VALUE  'DEPARTMENTAL EXPENSE REPORT'.
```

FIGURE 11-10
(Continued)

```
01  OUT-HEAD-2.
    05 FILLER      PIC X(030)
                   VALUE '  DEPARTMENT        DATE
    05 FILLER      PIC X(103)
                   VALUE 'TYPE OF EXPENSE      AMOUNT'.

01  OUT-DETAIL.
    05 FILLER      PIC X(004) VALUE SPACES.
    05 OUT-DEPT    PIC 9(003).
    05 FILLER      PIC X(010) VALUE SPACES.
    05 OUT-DATE    PIC X(008).
    05 FILLER      PIC X(005) VALUE SPACES.
    05 OUT-TYPE    PIC X(013).
    05 FILLER      PIC X(006) VALUE SPACES.
    05 OUT-AMOUNT PIC 9999.99.

01  OUT-FOOTER.
    05 FILLER      PIC X(049) VALUE SPACES.
    05 OUT-TOTAL   PIC 9(4).99.

PROCEDURE DIVISION.

000-HOUSE-KEEPING.
    OPEN INPUT  EXP-IN.
    OPEN OUTPUT EXP-RPT.
    PERFORM 150-PRINT-HEADINGS.

100-MAINLINE.
    PERFORM 200-READ-ROUTINE.
    PERFORM 300-PROCESS-DATA
        UNTIL EOF-FLAG = 1.
    PERFORM 400-TOTAL.
    CLOSE EXP-IN.
    CLOSE EXP-RPT.
    STOP RUN.

150-PRINT-HEADINGS.
    MOVE OUT-HEAD-1 TO OUT-REC.
    WRITE OUT-REC
        AFTER ADVANCING TO-NEW-PAGE.
    MOVE OUT-HEAD-2 TO OUT-REC.
    WRITE OUT-REC
        AFTER ADVANCING 2 LINES.
    MOVE SPACES TO OUT-REC.
    WRITE OUT-REC
        AFTER ADVANCING 1 LINES.
```

```
200-READ-ROUTINE.
    READ EXP-IN
        AT END MOVE 1 TO EOF-FLAG.

300-PROCESS-DATA.
    ADD  IN-AMOUNT  TO  WORK-TOTAL.
    MOVE IN-DEPT    TO  OUT-DEPT.
    MOVE IN-DATE    TO  OUT-DATE.
    MOVE IN-TYPE    TO  OUT-TYPE.
    MOVE IN-AMOUNT  TO  OUT-AMOUNT.
    MOVE OUT-DETAIL TO  OUT-REC.
    WRITE OUT-REC
        AFTER ADVANCING 1 LINES.
    PERFORM 200-READ-ROUTINE.

400-TOTAL.
    MOVE WORK-TOTAL  TO OUT-TOTAL.
    MOVE OUT-FOOTER  TO OUT-REC.
    WRITE OUT-REC
        AFTER ADVANCING 2 LINES.
```

ภาษาฟอร์แทรน (FORTRAN)

ภาษานี้จัดว่าเป็นภาษาที่นิยมใช้กันมากในหมู่ของผู้ที่ทำงานทางด้าน วิศวะ, คณิตศาสตร์ และวิทยาศาสตร์ มานานแล้ว คำว่า FORTRAN ย่อมาจากคำว่า **FOR**mula **TRAN**slation นั่นย่อมหมายความว่าภาษานี้มีจุดเด่นในการใช้กับสูตรในทางคณิตศาสตร์ ถึงแม้ว่าในปัจจุบันนี้จะมีการสร้างภาษาขึ้นมาเพื่อทดแทนในเรื่องของการคำนวณ แต่ภาษานี้ก็ยังนิยมใช้กันอยู่

ภาษาฟอร์แทรนนี้ พัฒนาโดยบริษัท IBM โดยประกาศตัวในปี ค.ศ.1957 ในฐานะของภาษาในระดับสูงที่มีสมรรถนะเพิ่มขีดการทำงานของเครื่องคอมพิวเตอร์ และก็นับว่าได้ประสบผลสำเร็จอย่างงดงาม ในปี ค.ศ.1960 ได้มีการพัฒนาภาษานี้ ให้สามารถทำงานได้บนเครื่องระดับไมโครคอมพิวเตอร์

ตัวอย่างของการเขียนโปรแกรมภาษาฟอร์แทรน

FIGURE 11-11
A FORTRAN program for finding the sum of the series of integers 1 + 2 + 3 + · · · + 100.

```
C       SUM OF A SERIES 1 TO 100
        INTEGER COUNT, SUM
        SUM = 0
        DO 20 COUNT = 1, 100
           SUM = SUM + COUNT
 20     CONTINUE
        PRINT 30, 'THE SUM OF 1 TO 100 IS ', SUM
 30     FORMAT (' ', A23, I6)
        STOP
        END
```

เนื่องจากภาษานี้จะเป็นไปทางคำนวณ เราจึงไม่นิยมใช้ภาษาฟอร์แทรนในการทำงานทางด้านธุรกิจ

ภาษาปาสคาล (Pascal)

ภาษาปาสคาล ถูกพัฒนาโดย Nicklaus Wirth และได้รับการยอมรับและใช้กันอย่างกว้างขวาง ในหมู่ผู้ประกอบอาชีพทางคอมพิวเตอร์ ภาษานี้มีจุดเด่นก็คือจะเป็นอิสระจากระบบเครื่อง (Hardware) ดังนั้น เราจึงใช้ภาษานี้บนเครื่องต่างๆ ตระกูล โดยไม่ต้องแก้ไขส่วนหนึ่งส่วนใดของโปรแกรม ภาษาปาสคาลนับว่าเป็นภาษาต้นตระกูลของการเขียนโปรแกรมแบบมีโครงสร้าง (structure programming) ซึ่งจัดว่าเป็นรูปแบบที่ดีของการออกแบบโปรแกรม

ถ้าเรานับความง่ายในการเรียนรู้แล้ว ภาษาปาสคาลกับภาษาเบสิค จัดว่าอยู่ในระดับเดียวกัน ด้วยเหตุผลที่กล่าวมาแล้ว ภาษาปาสคาลจึงมักจะถูกเลือกให้เป็นภาษาแรกในการเรียนรู้ในหมู่นักคอมพิวเตอร์ ปัจจุบันนี้ได้มีการนำภาษาปาสคาลไปพัฒนา และสร้างโปรแกรมสำเร็จรูปหลายประเภทด้วยกัน รวมทั้งโปรแกรมทางธุรกิจ การใช้ภาษานี้ค่อนข้างจะแพร่หลายมาก โดยเฉพาะอย่างยิ่งบนเครื่องตระกูลไมโครคอมพิวเตอร์

ตัวอย่างโปรแกรมภาษาปาสคาล

FIGURE 11-12
A structured Pascal program. Statements enclosed in braces { } are comments.

```
program interest(input, output);
     var Balance, Interest, Principal: real;
         Period: integer;
     begin
     {Read initial values}
     read(Principal, Interest, Period);
     {Calculate balance with interest}
     Balance := Principal*(1 + Period*Interest)
     writeln('The new balance is',Balance);
end.
```

ภาษาโมดูล่า 2 (Modula 2)

ภายหลังการสร้างงานขึ้นเอกที่มีคุณค่าคือภาษาปาสคาลมาได้ 10 ปี ก็ได้มีการพัฒนาสร้างภาษาขึ้นมาใหม่ โดยมีเจตนาในการที่จะทำให้ดียิ่งขึ้นกว่าภาษาปาสคาล และผลที่ได้ก็คือ ภาษาโมดูล่า 2 นั่นเอง ภาษาที่พัฒนามาได้ใหม่นี้ จะช่วยให้ผู้เขียนโปรแกรมหลีกเลี่ยงความผิดพลาดในการเขียนโปรแกรมที่ปรากฏบ่อย ดังเช่น การเขียนโปรแกรมภาษาปาสคาล และพยายามลดรูปแบบจากปาสคาลให้ง่ายลง มีนักคอมพิวเตอร์หลายๆ คนได้เปลี่ยนจากการใช้ภาษาปาสคาลมาใช้ภาษาโมดูล่า 2 แทนด้วยเหตุผลที่กล่าวมาแล้ว และนอกจากนั้นการเปลี่ยนแปลงจากภาษาปาสคาลมาสู่โมดูล่า 2 ก็กระทำได้ง่ายมาก เพราะทั้ง 2 ภาษานี้มีโครงรูปค่อนข้างคล้ายกันมากๆ

```
MODULE Sampleprogram;
FROM InOut IMPORT
     Writestring, WriteLn;
BEGIN
     WriteString ('Sample Lines of Modula-2 Output');
     WriteLn;
     WriteLn;
     WriteString ('These are sample lines of printed output');
     WriteString ('from a Modula-2 program.');
     WriteLn;
END Sampleprogram.
```

ภาษา C (C language)

ภาษา C พัฒนาขึ้นโดย Bell Laboratories ในต้นปี ค.ศ.1970 และได้นำภาษานี้ไปสร้าง โปรแกรมควบคุมระบบที่มีชื่อว่า UNIX ความสามารถของ C ในแง่ของการนำไป Compile ก็คือ จะให้ object code เทียบเท่ากับ object code ของภาษาแอสแซมบลี้ เราจัดได้ว่าภาษา C เป็นภาษาที่มีประสิทธิภาพในแง่ของการสร้าง object code ในขณะเดียวกันก็ยังเป็นภาษาเทียบได้กับภาษาระดับสูงโดยทั่วๆ ไป นอกจากนี้ยังจัดว่าเป็นภาษาที่มีรูปแบบของโปรแกรมแบบโครงสร้าง สิ่งหนึ่งซึ่ง C ตรงกับภาษาปาสคาลก็คือ ความเป็นอิสระจากส่วนเครื่อง ซึ่งทำให้เราสามารถปฏิบัติงานภาษา C กับเครื่องต่างตระกูลกัน ปัจจุบันนี้ ภาษา C ได้พัฒนามาใช้บนเครื่องไมโครคอมพิวเตอร์อย่างแพร่หลาย สามารถใช้ได้ทั้งระบบปฏิบัติการแบบ MSDOS และ PC-DOS บางคนที่ไม่ชอบภาษา C อาจจะติติงตรงที่ว่า ภาษานี้ค่อนข้างจะยากไปสำหรับผู้ที่เรียนรู้ แต่ข้อจำกัดนี้ก็คงไม่ใช่สาเหตุที่สำคัญมากเมื่อเทียบกับลักษณะเด่นของภาษานี้

```
/*          Sample C language program          */
/*  Finds the sum of integers from 1 to number */
main()
{     int counter = 0; number; sum = 0;
      printf("This program finds the sum of integers from 1 to");
      printf("a number that you enter. Enter the number and ");
      printf("press the return key");
      scanf("%d",&number);
      while (counter < number)
        {
           counter + + ;
           sum = sum + counter;
        }
      printf("The sum of the numbers is %F",sum);
}
```

ภาษายุคที่ 4 (Fourth-Generation Language)

เหตุผลอย่างหนึ่งที่เราจะเห็นได้ว่ามีการใช้เครื่องไมโครคอมพิวเตอร์หรือที่เรียกอีกนัยหนึ่งว่าเครื่องพีซีนั้น ก็เพราะใช้ได้คล่องตัวมากกว่าการต่อเทอร์มินัลเข้ากับเครื่องเมนเฟรม อีกทั้งการใช้เครื่องตระกูลไมโครนั้นก็ใช้ง่าย มีโปรแกรมสำเร็จรูปหลากหลายให้เลือกใช้ อีกทั้งบรรดาโปรแกรมสำเร็จรูปเหล่านี้ ก็มีลักษณะเป็น "มิตรกับผู้ใช้" (user friendly)

ในยุคแรกที่เครื่องรุ่นไมโครคอมพิวเตอร์กำเนิดขึ้นมานั้น ผู้ใช้งานจะต้องเขียนโปรแกรมขึ้นมาใช้เอง โดยส่วนใหญ่จะเลือกภาษาเบสิค แต่มาจนถึงยุคนี้แล้วเรามีโปรแกรมสำเร็จรูปที่ใช้กันแพร่หลาย เช่น LOTUS 1-2-3, EXCEL, DBASE, Word Processor ซึ่งบรรดาโปรแกรมสำเร็จรูปเหล่านี้อำนวยความสะดวกให้กับผู้ใช้ค่อนข้างจะมาก อาทิเช่น เราใช้คำสั่งที่ง่ายๆ เพียงไม่กี่คำสั่งก็สามารถทำงานลุล่วงดังที่ต้องการได้ ในขณะที่ถ้าเราเขียนโปรแกรม procedural language ขึ้นมาใช้งานเองอาจจะต้องเขียนเป็นพันๆ บรรทัด โปรแกรมสำเร็จรูปเหล่านี้ทำให้ผู้ใช้ใช้คำสั่ง nonprocedural command สั้นๆ และง่ายๆ ในการจัดการกับแฟ้มข้อมูล, คำนวณ, พิมพ์รายงาน ได้ทุกเวลาที่ต้องการบนเครื่องคอมพิวเตอร์ คุณสมบัติที่วิเศษเหล่านี้ จะปรากฏในนิยามของภาษาที่เรียกว่าภาษายุคที่ 4 (4GL) ซึ่งภาษากลุ่มนี้จะมีปรากฏทั้งในเครื่องระดับเมนเฟรมจนถึงระดับไมโครคอมพิวเตอร์

การเรียนรู้เพื่อใช้ภาษา 4GL นี้ก็กระทำได้ง่าย ไม่จำเป็นจะต้องใช้งานเฉพาะในขอบเขตของผู้ประกอบอาชีพในทางคอมพิวเตอร์เท่านั้น ลักษณะเด่นของ 4GL นอกเหนือจากที่กล่าวมาแล้ว 4GL ยังมีลักษณะเด่นคือ จะเป็นรูปแบบคล้ายประโยคในภาษาอังกฤษ มีคำแต่ละคำที่สื่อความหมายได้ชัดเจน การตรวจสอบ (Debugging) และทดสอบ (testing) ก็กระทำได้ง่ายกว่าภาษาที่เป็น procedural language

ในหมู่นักออกแบบระบบงาน มักจะใช้ภาษา 4GL ในการสร้างต้นแบบ (Prototype) ให้ผู้ใช้ระบบตรวจสอบดูก่อนที่จะนำไปออกแบบงานจริงต่อไป ภาษา 4GL ยังออกแบบให้ผู้ใช้สามารถใช้สอบถามสิ่งที่ต้องการ (queries) จากฐานข้อมูลของระบบงาน, สามารถสร้างแบบรายงานอย่างง่ายดาย ในหมู่ผู้ประกอบอาชีพการเขียนโปรแกรมนั้น ภาษา 4GL สามารถใช้ในการสร้าง automatic code ได้ดีอีกด้วย ซึ่งจะทำให้การพัฒนาโปรแกรมกระทำได้เร็วขึ้น เนื่องจากในแต่ละโปรแกรมสำเร็จรูปเหล่านี้จะมีเครื่องมือเหล่านี้อำนวยความสะดวกให้อยู่อย่างพร้อมเพรียงอยู่แล้ว

ตัวอย่างของโปรแกรมสำเร็จรูปที่เข้าข่ายดังกล่าว ซึ่งอาจจะจัดได้ว่าเป็นมาตร-

ฐานของการใช้ 4GL ร่วมกับระบบฐานข้อมูล ก็คือ SQL (IBM's Structured Query Language) SQL นั้นต้นกำเนิดการพัฒนานั้นอยู่ใช้งานบนเครื่องเมนเฟรม แต่ปัจจุบันได้พัฒนาสร้าง SQL ให้ใช้บนเครื่องพีซีได้แล้ว ซอฟท์แวร์ที่เข้าข่ายประเภท SQL จะปรากฏเป็นที่นิยมใช้มากมาย เช่น DBASE, Oracle คำสั่งใช้ในการป้อนคำถาม (queries) จะถือเป็นส่วนหนึ่งของ 4GL ที่ใช้การสืบค้นสารสนเทศที่ต้องการ

4GL จัดว่าเป็นแนวทางของการผสานความรู้เพื่อนำไปสู่รากฐานการพัฒนาระบบ ออกแบบระบบ และดูแลรักษาระบบต่อไป

บทสรุป

1. โปรแกรม คือชุดของคำสั่งที่จะให้เครื่องคอมพิวเตอร์ปฏิบัติงาน

2. คุณลักษณะของโปรแกรม จะปรากฏรายละเอียดของแหล่งให้กำเนิดข้อมูล, ปฏิบัติการจะให้เครื่องทำงาน และผลลัพธ์ที่จะได้จากการประมวลผล

3. ข้อมูลนำเข้านั้นเราอาจจะรับได้จากแป้นพิมพ์หรืออุปกรณ์อื่นๆ ของระบบคอมพิวเตอร์

4. ผลลัพธ์ (Output) หมายถึงสารสนเทศที่ได้จากการประมวลผลของเครื่องคอมพิวเตอร์ โดยผลที่ได้นี้จะแสดงออกทางจอภาพ, เครื่องพิมพ์ หรือจัดเก็บไว้ในสื่อ (media) ต่างๆ ก็ได้

5. เครื่องมือเครื่องใช้ที่ใช้ในการออกแบบโปรแกรมนั้น จะประกอบด้วย structure chart, flowchart หรือ pseudo program

6. structure chart จะเป็นผังที่แสดงถึงแนวทางในการแก้ปัญหาโดยเรียงตามลำดับชั้นจากระดับบนถึงระดับล่าง

7. โปรแกรมแบบลำลอง เราถือว่าเป็นการเขียนแบบ dry run ของการเขียนโปรแกรมจริงด้วย ภาษาคอมพิวเตอร์อีกชั้นหนึ่ง โดยที่โปรแกรมลำลอง จะมีรูปแบบคล้ายโปรแกรมที่ใช้งานจริงๆ เพียงแต่ว่าจะขาดรายละเอียดและไม่ปรากฏโครงสร้างที่ซับซ้อนเท่านั้นเอง

8. ผังโปรแกรม (Flowchart) จะประกอบด้วยสัญลักษณ์ที่มีความหมายกันในหมู่ผู้เขียนโปรแกรมและมีทิศทางคือลูกศรชี้ทิศทาง ผังดังกล่าวจะปรากฏส่วนหัวที่เป็นจุดเริ่มต้นที่เราเรียกว่า mainline หรือ driver module ในกรณีที่ driver module นั้นถูกองค์ประกอบของ mainline เรียกไปใช้งาน เราจะเรียก driver module นั้นว่าโปรแกรมย่อย (subroutine, or procedure)

9. โปรแกรมแบบมีโครงสร้าง จะแบ่งเป็น 3 ประเภท คือ แบบทีละลำดับ (sequence), แบบเลือก (selection) และแบบการทำซ้ำ (repetition)

10. ภายหลังการออกแบบ โดยใช้เครื่องมือชนิดใดชนิดหนึ่งที่ปรากฏในข้อ 5. แล้ว เราจะนำไปถอดรหัสต่อไปให้เป็นโปรแกรมภาษาคอมพิวเตอร์ต่อไป

11. การเขียนโปรแกรมนั้นควรจะเขียนในรูปแบบที่ดี เพื่ออำนวยความสะดวกในการใช้งานต่อไป อย่าเขียนโปรแกรมเพราะเพียงเจตนาจะให้คอมพิวเตอร์ใช้งานเท่านั้น

12. Interpreter หมายถึงโปรแกรมที่ใช้ในการแปลภาษาโปรแกรม เช่น

ภาษาเบสิคให้เครื่องปฏิบัติได้

13. Compiler หมายถึง โปรแกรมที่ทำหน้าที่ในการแปลโปรแกรมที่เราเขียน (source program) ให้เป็น ภาษาเครื่อง (object program)

14. การตรวจสอบ (Debugging) หมายถึงกระบวนการการค้นหาข้อผิดพลาดในตัวโปรแกรมโดยที่ Compiler หรือ interpreter จะช่วยในการค้นหาแล้วพิมพ์ออกมา

15. การทดสอบ (Testing) หมายถึงการทดสอบว่าโปรแกรมสามารถปฏิบัติงานได้โดยการกำหนดให้ส่งข้อมูลสมมติเข้าไปปฏิบัติงาน

16. ภาษาที่จัดอยู่ในกลุ่มของ procedural language เช่น COBOL, Pascal, C ด้วยสาเหตุที่ว่าการเขียนโปรแกรมในภาษาเหล่านี้นั้น ผู้เขียนจำเป็นจะต้องกำหนดขั้นตอน, ตรรก, วงจรการทำงานขึ้นมาเอง

17. ภาษา nonprocedural นั้น สามารถใช้งานได้ง่ายมากแม้กระทั่งผู้ฝึกหัดเขียนโปรแกรมที่ยังขาดทักษะในการเขียนโปรแกรมก็ยังสามารถเขียนได้เพราะภาษาตระกูลนี้ จะช่วยให้ผู้เขียนโปรแกรม เน้นที่ผลลัพธ์ที่ได้จากการกระทำเป็นสำคัญ

คำศัพท์ที่สำคัญ

Computer generations
BASIC
C language
COBOL
Compiler
Control structures
Debugging
Driver module
Flowcharts
FORTRAN
Fourth-generation language
Input
Interpreter
Mainline
Modula-2
Module
Nonprocedural
Object program
Output
Pascal
Procedural languages
Program
Programming
Pseudocode
Repetition structure
Selection structure
Sequence structure
Source program
Specifications
SQL
Structure chart
Style
Task
Testing

คำถามท้ายบท

จงจับคู่คำต่อไปนี้กับความหมายในแต่ละประโยค ข้อ 1-6

a. programming

b. specifications

c. flowchart

d. interpreter

e. module

f. debugging

.....1. เป็นสิ่งที่อธิบายถึงรายละเอียดของโปรแกรมที่ว่าด้วยส่วนนำข้อมูลเข้า การนำไปปฏิบัติการและการพิมพ์ผลลัพธ์ที่ได้ออกมา

.....2. เป็นปฏิบัติการเพื่อจะค้นหาและแก้ไขข้อผิดพลาดของโปรแกรมที่เขียนขึ้น เช่น ตกเครื่องหมายจุลภาค

.....3. เป็นสัญลักษณ์ของกล่องที่ปรากฏใน structure chart เพื่อบ่งบอกถึงงานที่จะต้องกระทำในการแก้ปัญหา

.....4. กิจกรรมของการเขียนโปรแกรม

.....5. เป็นโปรแกรมชนิดหนึ่งที่มีหน้าที่ในการนำคำสั่งที่เราเขียน ไปแปลและสั่งให้เครื่องทำงานตามที่ต้องการ โดยการแปลนั้นจะไม่สร้างรหัสภาษาเครื่อง

.....6. เป็นผังชนิดหนึ่ง ใช้เป็นเครื่องมือในการพัฒนา program logic

จงอธิบายคำถามย่อย ๆ ต่อไปนี้

1. จงอธิบายความหมายของคำว่า program และ programming
2. จงให้เหตุผลว่า ทำไมเราจึงต้องเขียนรายละเอียด (Specification) ก่อนที่จะนำไปออกแบบต่อไป
3. ข้อสนเทศในการอธิบายข้อมูลนำเข้าจะต้องมีองค์ประกอบอะไรบ้าง เพื่อที่เราจะได้นำไปเขียนโปรแกรมต่อไปได้อย่างถูกต้อง
4. จงอธิบายถึงกระบวนการในการพัฒนารูปแบบของ structure chart และแต่ละส่วนในผังดังกล่าวนั้นจะมีสัมพันธภาพในแต่ละส่วนอย่างไร
5. จงกล่าวถึงประโยชน์ของการใช้โปรแกรมลำลองที่มีต่อการเขียนขั้นตอนการออกแบบ

โปรแกรม

6. การใช้ผังโปรแกรม กับการเขียนโปรแกรมแบบลำลอง ให้แง่มุมแตกต่างกันอย่างไร โดยเฉพาะอย่างยิ่งในเรื่องของ logic
7. การออกแบบโปรแกรมแบบมีโครงสร้างคืออะไร มีอยู่กี่รูปแบบ
8. อะไรคือสาเหตุที่สำคัญ ของการเขียนโปรแกรมในประเด็นของความชัดเจน, การอ่านง่าย (readable) ในเรื่องนี้ท่านมีข้อเสนอแนะวิธีการเขียนโปรแกรมที่ดีอย่างไร
9. จงอธิบายความแตกต่างระหว่าง interpreter และ compiler
10. จงให้ทัศนะการใช้ภาษา COBOL, BASIC และ Pascal
