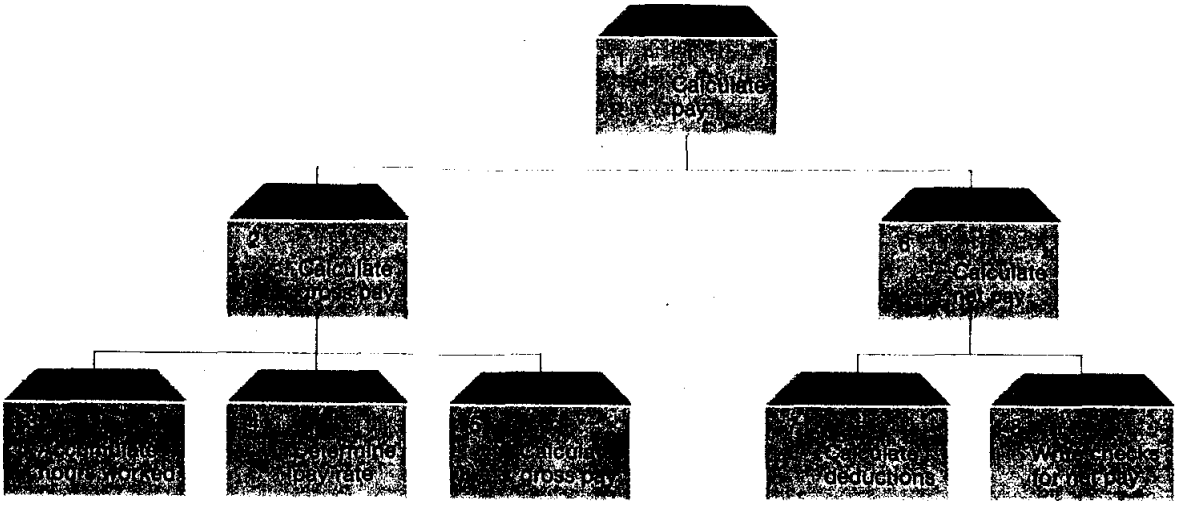


รูปที่ 8-9 แสดงระดับชั้นการทำงานของโปรแกรมเงินเดือน

8.5.3 HIPO charts

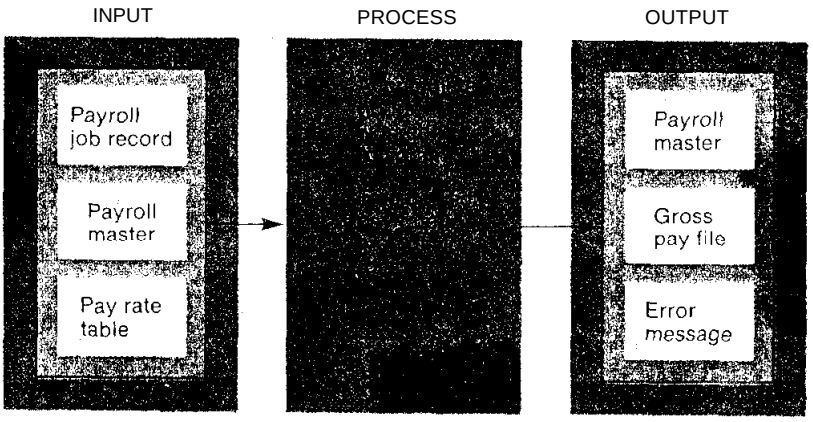
HIPO chart เป็นเครื่องมือสำหรับออกแบบโปรแกรมแบบ Top-down ซึ่งนำ hierarchy + input/processing/output เป็นการระบุถึงรายละเอียดของข้อมูลเข้า ข้อมูลออก และการประมวลผลของโมดูลต่างๆ โปรแกรมเมอร์ หรือผู้ออกแบบ ต้องกำหนด

(1) Output โดยกำหนดรูปแบบของผลลัพธ์, สื่อข้อมูล (media), การจัดระเบียบ (organization), ปริมาตร (Volume), ความถี่และจุดหมาย (frequency and destination)

(2) Input โดยกำหนดถึงแหล่งกำเนิด (source), รูปแบบ, สื่อข้อมูล, การจัดระเบียบ, ปริมาตร และความถี่

(3) Processing การประมวลผลที่จำเป็นซึ่งเป็นการคำนวณทางคณิตศาสตร์ การเปรียบเทียบทางตรรก และวิธีการที่แปลง (transform) จากข้อมูลเข้าเป็นผลลัพธ์

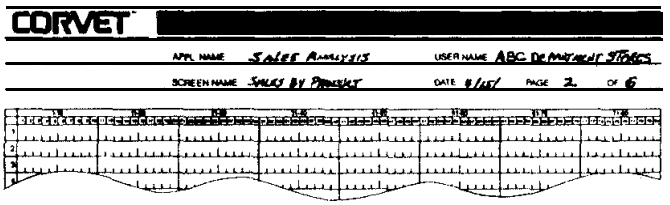
HIPO chart ของโมดูลหลัก เรียกว่า main control module ซึ่งจะถูกปฏิบัติงานเป็นโมดูลแรก ซึ่งผู้ออกแบบสามารถสร้างโมดูลในระดับล่างในโปรแกรมได้ ดังรูปภาพที่ 8-10 ซึ่งเป็น HIPO chart ของการคำนวณเงินได้สุทธิ ซึ่งเป็น module ที่ 2 ของรูปภาพที่ 8-9



รูป 8-10 เป็น HIPO ของการคำนวณเงินเดือน

8.5.4 Layout Forms

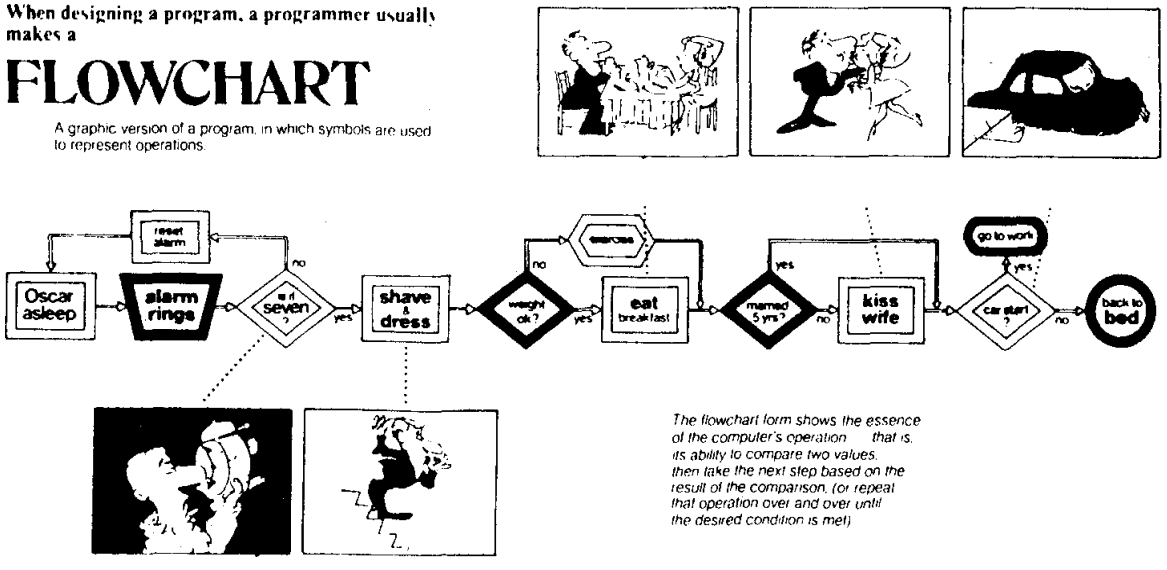
เป็นรูปแบบแผนผังที่ใช้สำหรับออกแบบรูปแบบของข้อมูลนำเข้า ผลลัพธ์ และสื่อเก็บข้อมูล โดยปกติเป็นรูปแบบที่เตรียมไว้ก่อน ในการออกแบบผู้ออกแบบเพียงแต่ทดลองใส่ข้อมูลหรือข้อความที่ต้องการใน Layout นี้ ซึ่ง Layout Forms นี้ ผู้ออกแบบสามารถนำไปออกแบบเอกสาร ข้อมูลเข้า ข้อมูลออก แฟ้มข้อมูล ซึ่งแสดงออกทางจอภาพ หรือเป็นรูปแบบของรายงานก็ได้



รูปที่ 8-11 เป็นผังของการออกแบบหน้าจอ

8.5.5 Flowcharts

Flowcharts (ผังงาน) เป็นเครื่องมือที่สำคัญสำหรับการเขียนโปรแกรมคอมพิวเตอร์และการวิเคราะห์ระบบ ผังงานคือกราฟฟิคที่ใช้แทนขั้นตอนในการแก้ปัญหา หรือแสดงส่วนประกอบของระบบ เป็นลำดับ หรือการตัดสินใจที่จะกระทำในระบบ

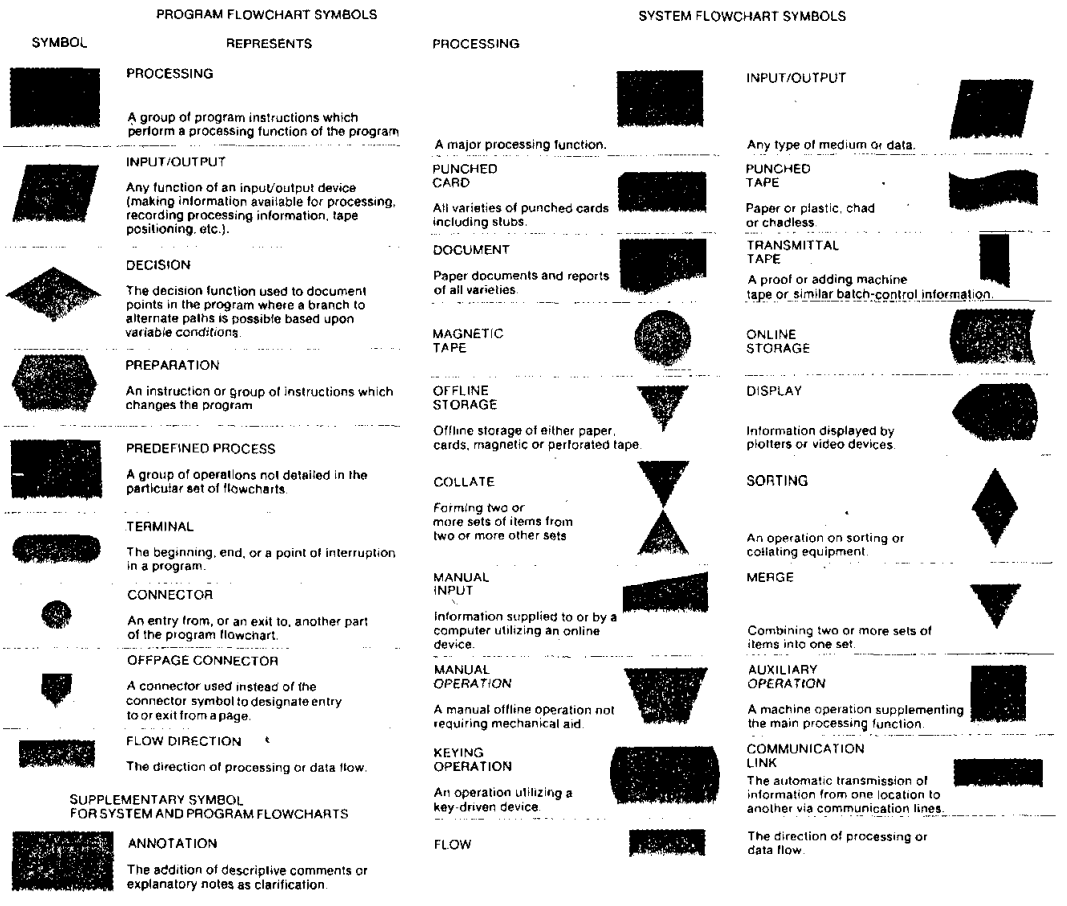


ผังงานแบ่งเป็น 2 ชนิด คือ

(1) system flowchart เป็นผังงานที่แทนส่วนประกอบและการไหลของระบบ

(2) program flowchart เป็นผังงานแทนขั้นตอนการประมวลผลข้อมูลที่กระทำในโปรแกรม

ซึ่งผังงานนี้โดยทั่วไป จะใช้สัญลักษณ์ผังงาน แทนการทำงานดังรูปภาพที่ 8-13



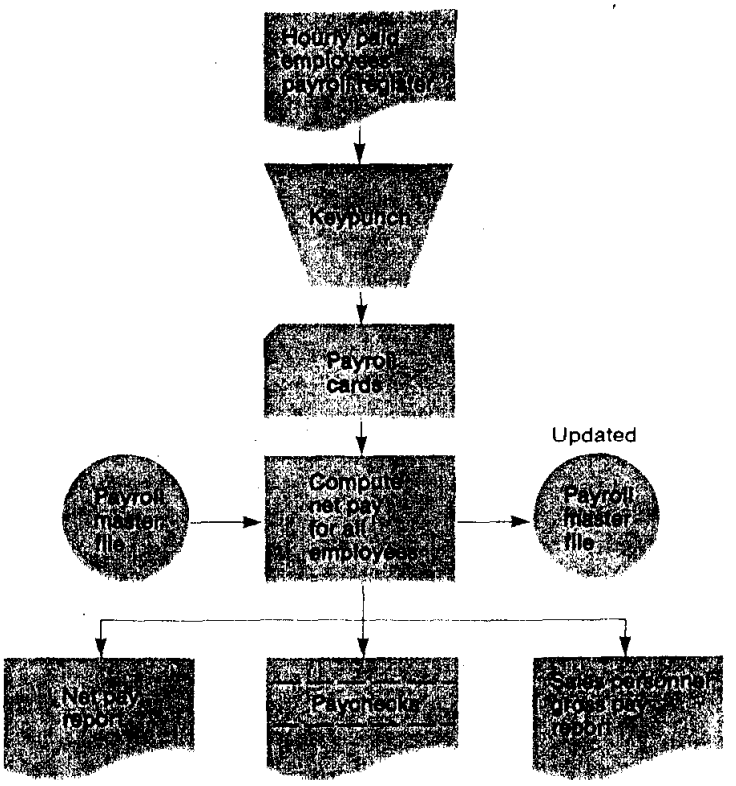
รูปที่ 8-13 แสดงสัญลักษณ์ของผังงาน

8.5.5.1 System Flowcharts

ผังระบบเป็นเครื่องมือสำหรับการพัฒนาระบบ โดยใช้สำหรับแสดงการไหลของข้อมูลในส่วนต่างๆ ของการประมวลผลข้อมูลของระบบ ไม่ว่าจะเป็นสถานะของข้อมูลนำเข้า (input) การประมวลผล (processingg) การแสดงผล (output) และการเก็บข้อมูล (storage) โดยไม่แสดงถึงการประมวลผลโดยละเอียด

รูปที่ 8-14 เป็นผังงานระบบเงินเดือนอย่างง่าย

จากรูปภาพ 8-14 แสดงถึงระบบเงินเดือน โดยใช้สัญลักษณ์ของผังงานพื้นฐาน โดยรูปสี่เหลี่ยมด้านขนาน แทน ข้อมูลเข้า หรือข้อมูลออก (INPUT/OUTPUT) ส่วนสี่เหลี่ยมผืนผ้า แทนการประมวลผล ลูกศรแทนทิศทางการไหลของข้อมูล



รูปที่ 8-15 แสดงผังงานของระบบเงินเดือน

รูปที่ 8-15 เป็นผังงานแสดงการไหลของข้อมูลในระบบเงินเดือนที่ละเอียดกว่ารูปที่แล้ว โดยแสดงถึงสื่อต่างๆ ที่ใช้ในระบบ ไม่ว่าจะเป็นสื่อที่ใช้สำหรับนำเข้าข้อมูล สื่อที่ใช้เก็บข้อมูล และสื่อที่ใช้ในการนำข้อมูลออก

8.5.5.2 Program Flowcharts

เป็นผังงานโปรแกรม แสดงถึงขั้นตอนต่างๆ ในการทำงานของโปรแกรมโดยละเอียด ซึ่งผู้ออกแบบสามารถใช้สำหรับ

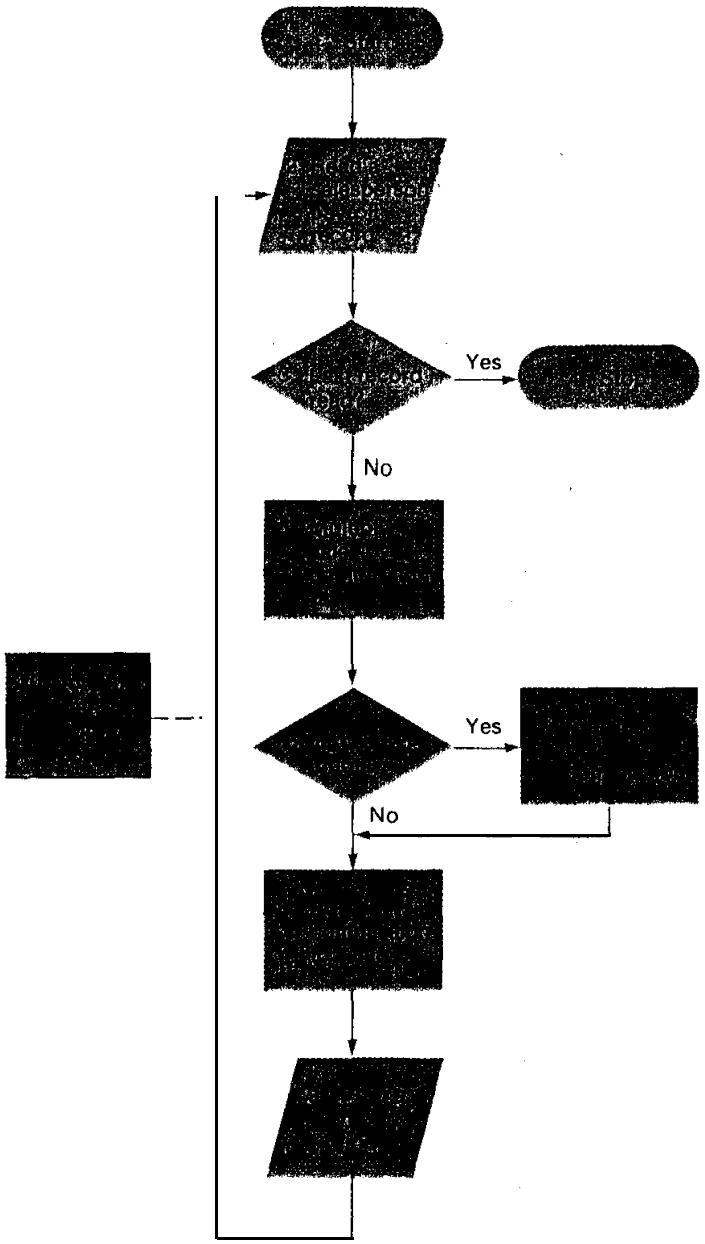
(1) ดูตรรกและลำดับของขั้นตอนต่างๆ ในการประมวลผล

(2) ทดลองวิธีการเขียนโปรแกรมต่างๆ

(3) ติดตามขั้นตอนการประมวลผลทั้งหมด

ผังงานโปรแกรมที่สมบูรณ์ จะช่วยแนะทาง ในการเขียนคำสั่ง (coding) การทดสอบ (testing) การจัดทำเอกสาร (documentation) และการบำรุงรักษา (maintenance)

General program flowchart-salesperson payroll report



1 This is the start of the program
2. A salesperson payroll' record is read as illustrated by the input/output symbol. It should contain data "fields" like the name, monthly salary, commission rate, and sales quota of each salesperson. The data record could be in the form of a punched card, or could be stored on magnetic tape or disk.
3. This is the "last record" decision point. Has the last data record been read?
4. If the answer is yes, the program comes to a stop.
5. If the answer is no, the processing symbol indicates that the sales amount on this data record should be multiplied by the commission rate to compute the sales commission earned.
6. Another decision pomt. Has the sales made exceeded the sales quota set for this salesperson?
7. If the answer is yes. a 10 percent bonus (10 percent of the normal commission) IS added to the commission earned.
6. If the answer IS no (and also whenever completing step 7) the sales commission earned is added to the regular monthly salary to compute the monthly gross pay for the salesperson.
9. A line on the Salesperson Payroll Report is printed. This would probably include the name, quota, sales. commission, salary, and gross pay for each salesperson.
10. Thiscommenfsymbol pointsoutthemain *loop* of the program.

รูปที่ 8-16 เป็นผังงานของโปรแกรมในการพิมพ์รายงานเงินเดือนของพนักงานขาย

รูปที่ 8-16 เป็นผังงานโปรแกรมของการคิดเงินเดือนของพนักงานขาย โดยใช้สัญลักษณ์ผังงานแทนการปฏิบัติการ

(1) เป็นจุดเริ่มต้นของโปรแกรม

(2) เป็นสัญลักษณ์ของการนำข้อมูลเข้าหรือออก (INPUT/OUTPUT) ในที่นี้หมายถึงการอ่านระเบียนของพนักงานขาย ซึ่งระเบียนที่อ่านนี้ ประกอบด้วย ฟิลด์ชื่อ ฟิลด์เงินเดือน ฟิลด์อัตราค่าคอมมิชชั่น และฟิลด์โควต้าการขาย ซึ่งระเบียนทั้งหมดของพนักงานขายนี้ อาจเก็บในเทปแม่เหล็ก หรือ ดิสก์ก็ได้

(3) เป็นสัญลักษณ์ของทางเลือก ซึ่งเป็นเงื่อนไขในการทำงาน โดยถ้าข้อมูลที่อ่านใน (2) นั้นเป็นเรคอร์ดสุดท้าย ให้ไป (4) แต่ถ้าไม่ใช่ ให้ไป (5)

(4) เป็นการหยุดการทำงานของโปรแกรม

(5) เป็นสัญลักษณ์ในการประมวลผล โดยทำการคำนวณค่าคอมมิชชั่นที่ได้รับจากการนำยอดขาย คูณด้วยอัตราค่าคอมมิชชั่น

(6) เป็นสัญลักษณ์ โดยตรวจสอบว่า ยอดขายของพนักงาน เกินโควต้าหรือไม่ ถ้าเกิน ไป (7) ไม่เกิน ไป (8)

(7) เพิ่มค่าคอมมิชชั่นอีก 10 เปอร์เซนต์

(8) ทำการคำนวณ รายรับสุทธิ จาก การนำเงินเดือนประจำบวกกับค่าคอมมิชชั่นที่ได้รับ

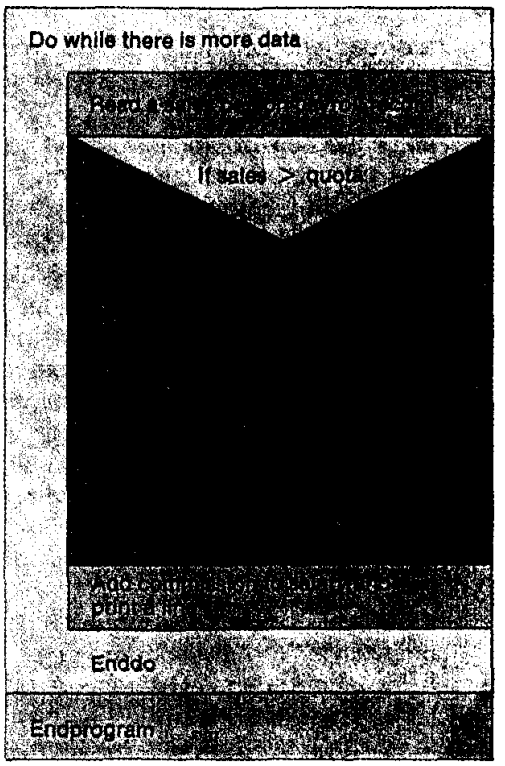
(9) พิมพ์รายงานเงินเดือนของพนักงาน อันประกอบด้วย ชื่อ, โควต้า, ยอดขายสินค้า, ค่าคอมมิชชั่น, เงินเดือน และรายรับสุทธิ

(10) เป็นสัญลักษณ์หมายเหตุ ซึ่งแสดงการทำงานวนรอบของโปรแกรม

8.5.3 Structured Flowcharts

ผังงานโครงสร้าง เป็นเครื่องมือที่ใช้แสดงขั้นตอนในโปรแกรม โดยใช้ โครงสร้างควบคุม (Control structures) และใช้รูปแบบ box-within-box ซึ่งแสดงถึงการปฏิบัติการในลำดับต่างๆ ในทิศทางจาก บนลงล่าง (top-down)

flowchart-salesperson
payroll report processing



รูปที่ 8-17 แสดงผังงานโครงสร้างของการออกรายงานเงินเดือนของพนักงาน

8.5.6 Pseudocode

เป็นเครื่องมือที่ใช้ออกแบบรายละเอียดของโปรแกรม ซึ่งแสดงถึงตรรกในการประมวลผลของโมดูลต่างๆ ในโปรแกรม โดยเขียนเป็นภาษาอังกฤษ อธิบายถึงการทำงาน โปรแกรมเมอร์สามารถใช้ความคิดเพื่อแก้ไขปัญหาต่างๆ ในโปรแกรมได้ ซึ่งแต่ละโมดูลเป็นการแก้ปัญหาในส่วน
รายละเอียด HIPO chart นั่นเอง

คำอธิบายที่เขียนนี้ จะไม่ใช่คำสั่งภาษาโปรแกรมใดๆ การเขียนจะอธิบายถึงหน้าที่ประมวลผลเป็นสำคัญ ซึ่ง Pseudocode นี้สามารถนำไปพัฒนาโดยใช้ภาษาโปรแกรมใดๆ ก็ได้ ไม่ขึ้นกับภาษาใดโดยเฉพาะ Pseudocode ที่ดีจะช่วยให้โปรแกรมเมอร์เขียนคำสั่งโปรแกรมได้ง่ายขึ้น

Pseudocode example for salesperson payroll report processing

```
Begin Program
  Read a salesperson payroll record
  Do while there is more data
    multiply sales by commission rate
    If sales greater than quota
      then add 10% bonus to commission
    End-If
    Add commission to salary
    Print a report line
    Read a sales person payroll record
  End-Do
End Program
```

รูปที่ 8-18 เป็นตัวอย่างของคำสั่งเทียมของการประมวลผลเงินเดือนพนักงาน

8.5.7 Decision Tables

ตารางการตัดสินใจ หรือ decision tables เป็นเครื่องมือที่สำคัญในการวิเคราะห์ระบบ และสำหรับโปรแกรมเมอร์ ในการออกแบบและวิเคราะห์โปรแกรมที่ซับซ้อน โปรแกรมส่วนมากที่ซับซ้อนจะประกอบด้วยคำสั่งเงื่อนไข และทางเลือกในการปฏิบัติงานหลายทาง โปรแกรมใดที่มีทางเลือกมากๆ จะทำให้โปรแกรมนั้นมีความซับซ้อนและยากตามไปด้วย ซึ่งก่อให้เกิดความผิดพลาดในการปฏิบัติงานได้

ดังนั้นโปรแกรมใดที่มีทางเลือกของการทำงานโดยปฏิบัติการตามเงื่อนไขในโปรแกรมได้หลายทาง สามารถใช้ ตารางการตัดสินใจ เพื่อช่วยแสดงตรรกของโปรแกรมและระบบได้

รูปแบบโดยทั่วไปของตารางการตัดสินใจ ประกอบด้วย

(1) condition stub เป็นรายละเอียด เงื่อนไข หรือคำถาม ซึ่งจะให้ผลลัพธ์ได้ 2 กรณีคือ จริงและเท็จ โดยปกติจะเป็นเงื่อนไขที่บรรจุในสัญลักษณ์ตัดสินใจของผังงาน

(2) action stub เป็นรายละเอียดของคำสั่ง หรือ การปฏิบัติการทั้งหมด ที่ระบบสามารถกระทำได้

(3) condition entry เป็นส่วนที่แสดงถึงผลลัพธ์ของเงื่อนไขต่างๆ ของ condition stub

(4) action entry เป็นส่วนที่แสดงถึง การปฏิบัติการที่ถูกกระทำจากเงื่อนไขต่างๆ

Table Heading	Decision Rule Heading
Condition statements	Condition entries
Action statements	Action entries

รูปที่ 8-19 เป็นรูปทั่วไปของตารางการตัดสินใจ

ตารางการตัดสินใจส่วนมาก ประกอบด้วย หัวตาราง (heading) และจำนวนของกฎเกณฑ์ต่างๆ ที่ใช้ในการตัดสินใจ ในแนวตั้ง จะเรียกว่า body เป็นส่วนของ condition entry และ action entry ซึ่งแสดงถึงกฎในการตัดสินใจ ถ้าระบุเงื่อนไขหนึ่งๆ จะเกิดเหตุการณ์หนึ่งๆ ที่ต้องกระทำ เช่น if condition 1 เป็นจริง กระทำ action 1 เป็นต้น โดยเงื่อนไขนี้อาจมีความซับซ้อน นอกจากการแสดงค่าความเป็นจริง และเท็จ ยังอาจเกิดจากการเปรียบเทียบค่าของข้อมูล โดยใช้สัญลักษณ์ < <= >= > เช่น if money > 100 then ซื้อขนมเค็ก เงื่อนไขนี้ขึ้นอยู่กับจำนวนเงินที่มีอยู่ในกระเป๋า ถ้าท่านมีเงินมากกว่า 100 บาท สิ่งที่ท่านจะกระทำคือ ซื้อขนมเค็ก เป็นต้น

ในการพัฒนาโปรแกรม ตารางตัดสินใจ ถือว่าเป็นเครื่องมือที่ช่วยให้ผู้พัฒนาเข้าใจระบบได้ดี เพราะแสดงถึงกฎเกณฑ์ต่างๆ ทั้งหมดในระบบที่มีความแตกต่างกันทั้งเงื่อนไขและการกระทำ

	Payroll Table No. 1	Decision Rule Numbers						
		1	2	3	4	5	6	7
Conditions	Hourly paid employee	Y						
	Salaried employee		Y					
	Executive employee			Y				
	Unclassified employee				Y			
	Salesperson					Y	Y	Y
	Made sales?					N	Y	Y
	Exceeded quota?					N	N	Y
Actions	Compute wages	X						
	Compute salary		X					
	Compute sales salary					X	X	X
	Compute commission						X	X
	Compute bonus							X
	Salesperson gross pay processing					X	X	X
	Net pay processing	X	X			X	X	X
	Go to payroll table number:			2	3			

รูปที่ 8-20 แสดงถึงตารางตัดสินใจของระบบเงินเดือน

8.6 การเขียนคำสั่งโปรแกรม (Program Coding)

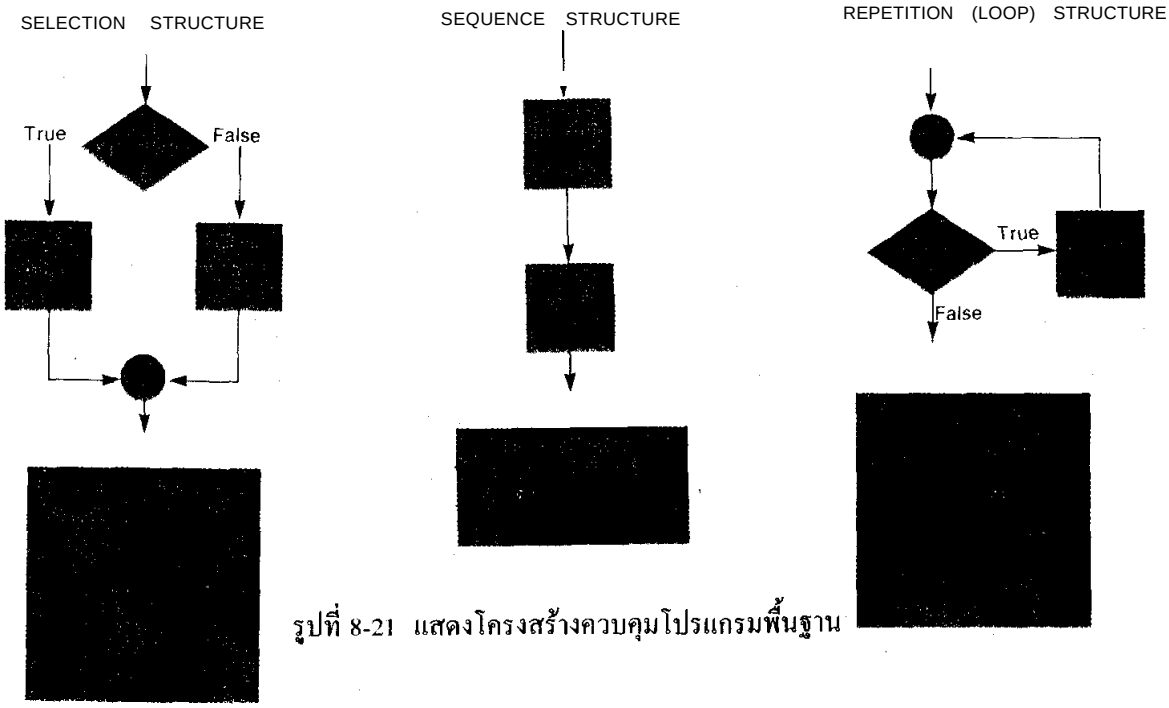
การเขียนคำสั่งโปรแกรมเป็นขั้นตอนในการแปลง (convert) ตรรกที่ได้ออกแบบในระยะการออกแบบโปรแกรม ให้เป็น กลุ่มของคำสั่งโปรแกรมภาษา เพื่อสั่งให้คอมพิวเตอร์ปฏิบัติตาม โปรแกรมภาษาในปัจจุบันมีมากมายหลายภาษา ซึ่งเหมาะกับงานทางด้านต่างๆ ซึ่งแต่ละภาษามีการเขียนที่แตกต่างกัน ทั้งรูปแบบ กฎเกณฑ์ต่างๆ ดังนั้น ผู้เขียนควรศึกษาถึงรูปแบบ และกฎเกณฑ์ต่างๆ เหล่านี้ก่อน

8.6.1 คำสั่งโครงสร้าง (Structured Coding)

โปรแกรมภาษาใดๆ ที่นิยมใช้ในปัจจุบัน จะประกอบด้วย คำสั่งโครงสร้างพื้นฐาน 3 อย่าง คือ

1. ลำดับ (sequence)
2. ทางเลือก (selection)
3. วนรอบ (loop หรือ repetition)

ซึ่งเป็นส่วนสำคัญ ในการเขียนโปรแกรมโครงสร้างแบบบนลงล่าง (top-down structure) จะช่วยให้การเขียนโปรแกรมเป็นมาตรฐาน (standardizes) และเข้าใจได้ง่าย รวมทั้งการแก้ไขง่ายอีกด้วย



รูปที่ 8-21 แสดงโครงสร้างควบคุมโปรแกรมพื้นฐาน

Sequence Structure

เป็นโครงสร้างลำดับ ซึ่งแสดงถึงลำดับของคำสั่งหรือการปฏิบัติงาน กล่าวคือ คำสั่งซึ่งอยู่ก่อนจะถูกปฏิบัติงานก่อน ดังนั้น คำสั่งโปรแกรมซึ่งเก็บไว้ในคอมพิวเตอร์ก่อนจะถูกทำงานก่อน

Selection Structure

โครงสร้างทางเลือก หรือเรียกว่า decision หรือ IF-THEN-ELSE ก็ได้ เป็นโครงสร้างซึ่งแสดง ทางเลือกของการทำงาน โดยขึ้นกับผลของเงื่อนไข โดยเงื่อนไข นี้ผลลัพธ์ มี 2 ทางคือ จริง (True) และเท็จ (False) ถ้าเงื่อนไขเป็นจริงจะกระทำอย่างหนึ่ง ถ้าเงื่อนไขเป็นเท็จจะกระทำอีกอย่างหนึ่ง

Repetition (Loop) Structure

โครงสร้างวนรอบ หรือเรียกว่า DO-WHILE หรือ DO-UNTIL ก็ได้ เป็นโครงสร้างที่กระทำหน้าที่ หรือคำสั่ง ซึ่งขึ้นกับเงื่อนไข โดยการทำงานจะเป็นการทำงานซ้ำๆ กัน ซึ่งจะหยุดการทำงานวนรอบนี้ ก็ต่อเมื่อเงื่อนไขเป็น เท็จ (False)

โปรแกรมภาษาที่มีคำสั่งโครงสร้างพื้นฐานเหล่านี้ เช่น Pascal, PL/I, COBOL เป็นต้น เป็นโปรแกรมภาษาที่สามารถเขียนเป็นโครงสร้างแบบ Top-down การออกแบบและการเขียนคำสั่งก็ง่ายต่อการควบคุม เพราะไม่มีคำสั่งในการกระโดดข้าม (branching) จากการไหลหลักๆ ของโปรแกรม (main flow) โมดูลในการควบคุมหลัก เรียกว่า mainline โดยปกติควรไหลจากบนลงล่าง โดยไม่มีการกระโดดจากล่างไปบน ซึ่งควรหลีกเลี่ยงการใช้คำสั่ง GO TO ให้น้อยที่สุด

8.6.2 ชนิดของคำสั่ง (Types of Instructions)

คำสั่งต่างๆ ที่โปรแกรมเมอร์สามารถใช้สำหรับเขียนโปรแกรม สามารถแบ่งออกได้เป็น 6 พวกด้วยกันคือ

(1) Specification instructions

เป็นคำสั่งซึ่งใช้สำหรับกำหนดรายละเอียดของสื่อข้อมูลที่ใช้ ซึ่งอาจเป็นขนาด หรือรูปแบบของ ระเบียนข้อมูล และแฟ้มข้อมูล หรือการกำหนดค่าคงที่ในโปรแกรม หรือการจัดสรร (allocation) หน่วยความจำ เช่น คำสั่ง FORMAT ของโปรแกรมภาษา FORTRAN หรือคำสั่ง PICTURE ของโปรแกรมภาษา COBOL เป็นต้น

(2) Input/Output instructions

เป็นคำสั่งที่เกี่ยวข้องกับการส่งต่อข้อมูลระหว่าง CPU กับอุปกรณ์ในการรับ-ส่งข้อมูล เช่น คำสั่ง READ ในภาษา Pascal คำสั่ง PRINT ในภาษา BASIC เป็นต้น

(3) Data movement instructions

เป็นคำสั่งที่ใช้สำหรับเปลี่ยนแปลงค่าของข้อมูลในหน่วยความจำหลัก เช่น คำสั่ง MOVE, SHIFT หรือ STORE เป็นต้น

(4) Arithmetic instructions

เป็นคำสั่งที่ใช้ในการคำนวณ เช่น คำสั่ง ADD, SUBTRACT เป็นต้น

(5) Logical instructions

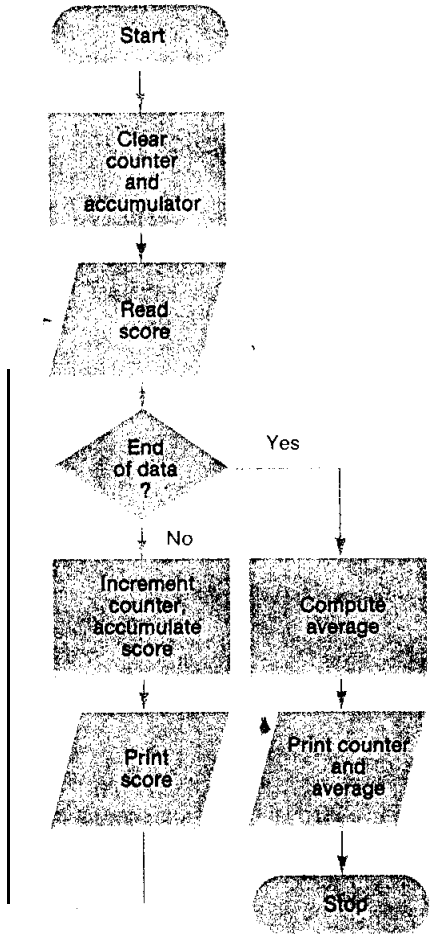
เป็นคำสั่งที่ใช้สำหรับเปรียบเทียบค่า หรือ ทดสอบเงื่อนไข เช่น คำสั่ง IF, THEN หรือ COMPARE เป็นต้น

(6) Control instructions

เป็นคำสั่งเพื่อใช้หยุด หรือเริ่มต้น การทำงานของโปรแกรม, เปลี่ยนลำดับการทำงานของโปรเซส, ควบคุมการใช้โปรแกรมย่อย คำสั่งที่ใช้ควบคุม เช่น คำสั่ง DO, RETURN, STOP เป็นต้น

FIGURE 8-22 Statements and flowchart of a simple BASIC program

```
'10 REM AVERAGE EXAM SCORE PROGRAM
'20 LET C = 0
'30 LET T = 0
'40 READ S
'50 IFS = 9999 THEN 100
'60 PRINT S.
'70 c = c t 1
*80 T = T + S
'90 GO TO 40
'100 A = T/C
'110 PRINT "NUMBER OF STUDENTSTAKING EXAM = "; C
*120 PRINT-AVERAGE EXAM SCORE = "; A
*130 DATA 87, 64, 95, 77, 82, 73, 98, 70, 63, 91, 9999
'140 END
'RUN
67649577627398706391
NUMBER OF STUDENTS TAKING EXAM = 10
AVERAGE EXAM SCORE = 60.0
```



รูปที่ 8-22 แสดงผังงานและโปรแกรมภาษา BASIC

8.7 การตรวจสอบความถูกต้องของโปรแกรม (Program Verification)

การตรวจทานโปรแกรม โดยทั่วไปเรียกว่า debugging ซึ่งเป็นระยะหนึ่งในการเขียนโปรแกรม ซึ่งรวมถึงการตรวจสอบ (checking) การทดสอบ (testing) และ การทำให้ถูกต้อง (correction) เพราะการเขียนคำสั่งโปรแกรมใหม่ๆ อาจเกิดข้อผิดพลาด (bugs) ได้ง่าย

8.7.1 ข้อผิดพลาดในการเขียนโปรแกรม (Programming Error)

ข้อผิดพลาดในการเขียนโปรแกรม สามารถแบ่งเป็น 3 ชนิดคือ syntax, logic และ system design errors

syntax error เป็นข้อผิดพลาดที่เกิดจากการเขียนคำสั่งโปรแกรมผิดรูปแบบ ไวยากรณ์ของภาษา

logic error เป็นข้อผิดพลาดที่เกิดจากการใช้ตรรกผิดในโปรแกรม ซึ่งเงื่อนไขผิดมีผลให้การกระทำตามเงื่อนไขผิดไปด้วย

System design error เป็นข้อผิดพลาดจากการออกแบบระบบ ทำให้ผลของการทำงานไม่เป็นที่ยอมรับของผู้ใช้ ซึ่งความผิดพลาดนี้ อาจเกิดจากการสื่อสารระหว่างโปรแกรมเมอร์กับผู้วิเคราะห์ระบบ หรือผู้ใช้ระบบ ไม่ดี

ความผิดพลาดต่างๆ ในการเขียนโปรแกรม syntax error เป็นความผิดพลาดที่ค้นพบได้ง่ายกว่า logic error เพราะสามารถตรวจพบได้ในระหว่างทำการแปลภาษา ส่วน logic error จะตรวจพบเมื่อโปรแกรมทำงานจนได้ผลลัพธ์ที่ไม่ถูกต้องเท่านั้น

8.7.2 การตรวจสอบ (Checking)

การตรวจสอบโปรแกรม ต้องมีขึ้นระหว่างการออกแบบโปรแกรม, การเขียนโปรแกรม, การตรวจทานโปรแกรม เพื่อให้แน่ใจว่าสามารถแก้ปัญหาได้ตรงตามความต้องการ ตามวัตถุประสงค์ที่ได้ตั้งไว้ เครื่องมือที่ใช้ในการออกแบบถูกต้องตามตรรก ในการประมวลผล รวมทั้งคำสั่งโปรแกรมที่เขียนขึ้น สามารถแปลได้โดยปราศจากข้อผิดพลาดต่างๆ และสามารถทำงานได้อย่างถูกต้อง สมบูรณ์

8.7.3 Structured walkthroughs

เป็นเครื่องมือของการออกแบบ, ขียนคำสั่ง ตรวจสอบความผิดพลาดของการเขียนโปรแกรมที่ดี ซึ่งเป็นวิธีการที่ผู้เขียนโปรแกรม แสดงผลงานให้โปรแกรมเมอร์อื่นๆ ตรวจดู ซึ่ง

การทำงานนี้ เป็นแนวความคิดของการเขียนโปรแกรมเป็นทีมงาน ซึ่งมีการกำหนด ให้พัฒนาโปรแกรมเดียวกันภายใต้การควบคุมของหัวหน้าโปรแกรมเมอร์ (chief programmer) โดยสมาชิกในทีมงานจะช่วยกันตรวจดูถึงการออกแบบและการเขียนคำสั่งเป็นช่วงๆ เป็นประจำ ของแต่ละโมดูล ซึ่งทำให้ค่าใช้จ่ายของการทวนสอบ (Verification) น้อยลง โดยสามารถพบความผิดพลาดได้ในระยะเริ่มแรกของการเขียนโปรแกรม โดยไม่ต้องคอยจนกระทั่งตรวจพบในระยะของการทดสอบโปรแกรม ซึ่งเป็นการยากที่จะทราบว่าจุดใดที่เกิดข้อผิดพลาด ซึ่งมีผลทำให้ค่าใช้จ่ายสูงตามไปด้วย

8.7.4 การทดสอบ (Testing)

การทดสอบ เป็นการตรวจสอบการทำงานของโปรแกรม โดยใช้ข้อมูลทดสอบ (test data) เพื่อดูผลลัพธ์จากการทำงาน การทดสอบที่ดีนั้นควรใช้ข้อมูลที่แตกต่างกัน เพื่อให้สามารถกระทำได้ในทุกๆ เงื่อนไข ในการทำงานของโปรแกรม รวมทั้งควรทดลองข้อมูลที่ไม่ถูกต้องด้วย โปรแกรมที่ดีเมื่อใส่ข้อมูลผิดๆ จะไม่เกิด error แต่จะแสดงข้อความเพื่อเตือนเท่านั้น

ในการเขียนโปรแกรมโครงสร้างที่ใช้โปรแกรมภาษาระดับสูงนั้น ผู้ทำการทดสอบโปรแกรม สามารถแบ่งการทดสอบโปรแกรมออกเป็นส่วนๆ ซึ่งเรียกว่า โมดูล เพื่อสะดวกในการทดสอบซึ่งง่ายต่อการค้นหาข้อผิดพลาด และง่ายต่อการแก้ไขให้ถูกต้องด้วย เมื่อทดสอบโมดูลย่อยๆ เหล่านี้จนไม่มีข้อผิดพลาด จึงนำมารวมเป็นโปรแกรมหลัก เพื่อทดสอบโปรแกมหลักอีกครั้งหนึ่ง

การทดสอบโปรแกรมกรณีที่ต้องการให้โปรแกรมที่พัฒนาขึ้นมา แทนที่การประมวลผลข้อมูลเก่าซึ่งกำลังดำเนินอยู่ การทดสอบต้องกระทำขนานกันไปกับระบบเก่า ซึ่งเราเรียกการประมวลผลแบบนี้ว่า parallel processing ดังนั้นการทดสอบระบบต้องทดสอบจนกว่าจะแน่ใจว่าสามารถกระทำแทนที่ระบบเก่าได้ จึงจะนำเอาการปฏิบัติงานของระบบเก่าออกไปได้

8.8 เอกสารโปรแกรม (Program Documentation)

เป็นเอกสารซึ่งบันทึกรายละเอียดของการออกแบบ การเขียนคำสั่ง ซึ่งมีประโยชน์ต่อการวิเคราะห์ความผิดพลาดของโปรแกรม การแก้ไขปรับปรุงโปรแกรม หรือ การรวมโปรแกรมกรณีเกิดการสูญหาย โดยเฉพาะโปรแกรมเมอร์หลักที่เขียนโปรแกรมเกิดลาออกไป ดังนั้นควรเก็บรวบรวมข้อมูลต่างๆ เอาไว้

Program Specifications
Describe what the program is supposed to do.

Program Description
Consists of Structure charts, HIPO charts, pseudocode, input/output and storage layout sheets, program flowcharts, decision tables, program listing, and a narrative description of what the program does.

Verification Documentation
Includes listings of test data and results. memory dumps, and other test documents

Operations Documentation
Consists of operating instructions which describe the actions required of the computer operator during the processing of the computer program.

Maintenance Documentation
A detailed description of all changes made to the program after it was accepted as an operational program.

รูปที่ 8-23 เป็นรายละเอียดของเอกสารโปรแกรม

8.9 การบำรุงรักษาโปรแกรม (Program Maintenance)

ขั้นตอนสุดท้ายในการเขียนโปรแกรมคอมพิวเตอร์ จะเริ่มหลังจากโปรแกรมคอมพิวเตอร์ที่พัฒนาขึ้นนั้นเป็นที่ยอมรับจากผู้ใช้และได้ปฏิบัติงานมาแล้วชั่วระยะเวลาหนึ่ง ต่อมาเกิดความจำเป็นบางอย่างเกิดขึ้น เช่น ต้องการปรับปรุงระบบให้มีประสิทธิภาพดีขึ้น หรือต้องการแก้ไขหน้าที่บางอย่าง หรือต้องการขยายขีดความสามารถของโปรแกรม หรืออาจเกิดความผิดพลาดในโปรแกรมต้องการแก้ไขให้ถูกต้อง โดยอาจมีผลมาจากนโยบายของบริษัทที่เปลี่ยนไป หรือระเบียบทางราชการบังคับ หรือ การแข่งขันทางธุรกิจ ฯลฯ การเปลี่ยนแปลงต่างๆ อาจเกิดได้กับทั้งฮาร์ดแวร์และซอฟต์แวร์ของระบบ การบำรุงรักษาโปรแกรมเป็นหน้าที่หลักของการประมวลผลข่าวสารของหน่วยงาน เกี่ยวข้องกับ การวิเคราะห์ การออกแบบ การเขียนคำสั่ง การตรวจสอบ การเปลี่ยนแปลงเอกสารให้ทันสมัย ซึ่งจะมี maintenance programmers เป็นผู้ที่รับผิดชอบหน้าที่บำรุงรักษาโปรแกรม ซึ่งในความเป็นจริงแล้ว ผู้เขียนโปรแกรม กับผู้ที่บำรุงรักษาโปรแกรมจะเป็นคนละทีมงานกัน ดังนั้นจึงเป็นการยากที่ต้องแก้ไขโปรแกรมที่ไม่ได้พัฒนาขึ้นมา ดังนั้นสิ่งสำคัญของการเขียนโปรแกรมโครงสร้าง จะช่วยให้การเขียนโปรแกรมเป็นมาตรฐาน และทำให้ง่ายต่อการอ่านและเข้าใจ

สรุปท้ายบท

1. วัฏจักรในการพัฒนาระบบ แบ่งเป็น 6 ขั้นตอน คือ System investigation, System analysis, System design, Software development, Systems implementation, Systems maintenance ขั้นตอนในการพัฒนาโปรแกรม แบ่งเป็น 6 ขั้นตอน คือ Program analysis, Program design, Program coding, Program verification, Program documentaion และ Program maintenance
2. ปัญหาที่เกิดขึ้นในการพัฒนาระบบคือ คุณภาพไม่ดี ผลผลิตต่ำ และค่าใช้จ่ายสูง การแก้ไข โดยเขียนเป็นโปรแกรมโครงสร้าง (structure Programming) ใช้โปรแกรมช่วย (computer-assisted programming) และ ผู้ใช้ (user)
3. การเขียนโปรแกรมโครงสร้าง คือวิธีการเขียนโปรแกรม ซึ่งออกแบบโปรแกรมจากบนลงล่าง (Top-down) และใช้จำนวนโครงสร้างควบคุม (control structure) จำกัด เครื่องมือที่นิยมใช้เช่น Top-down design, structure and HIPO charts, pseudocode, structure coding และ structure walkthroughs
4. User programming ได้ถูกจัดเตรียมสำหรับผู้ใช้เพื่อสนับสนุนฮาร์ดแวร์ เช่น development intelligent workstations สำหรับซอฟต์แวร์ เช่น development system, data base management systems, fourth generation language และ application packages และสนับสนุนหน่วยงาน เช่น ศูนย์กลางข่าวสาร (information center)
5. เครื่องมือในการออกแบบโปรแกรม เช่น structure charts, HIPO charts, flowchart, pseudo code, และ decision table นอกจากเป็นเทคนิคที่ใช้ในการออกแบบโปรแกรมแล้วยังมีส่วนช่วยในการเขียนคำสั่ง debugging การจัดทำเอกสาร และ บำรุงรักษาโปรแกรม อีกด้วย
6. โปรแกรมคอมพิวเตอร์ที่พัฒนาขึ้นมานั้นสามารถทำการตรวจทาน โดยการ debugging และแก้ไขได้เท่าที่จำเป็น โดย program maintenance

คำศัพท์ที่สำคัญ

Programming cycle
Systems development cycle
Stages of programming
Structured programming
Computer-assisted programming
User programming
Application development systems
Information centers
Programming tools
Program analysis
Program design
Algorithm
Top-down design
Program modules
Structured charts
HIPO charts
System flowcharts
Program flowcharts
Layout forms
Structured flowcharts
Program loops
Pseudocode
Decision tables
Program coding
Basic control structures
Types of computer instructions
Program verification
Debugging
Syntax errors
Logic errors
System design errors
Program checking
Structure walkthroughs
Program testing
Program documentation
Program maintenance

คำถาม

1. ทำไมถึงต้องมีการเปลี่ยนวิธีการเขียนโปรแกรม?
2. การเขียนโปรแกรมคอมพิวเตอร์ มีการเปลี่ยนแปลงไปอย่างไร ซึ่งมีผลกระทบต่อผู้ใช้ และโปรแกรมเมอร์อย่างไร
3. โปรแกรมโครงสร้าง (structure programming) คืออะไร จงบอกถึงประโยชน์ที่ได้
4. Top-down design คืออะไร มีประโยชน์อย่างไร
5. structure chart, HIPO chart, layout forms, decision table และ Pseudocode มีวัตถุประสงค์เพื่ออะไร
6. ขั้นตอนในการเขียนโปรแกรมคอมพิวเตอร์ มีอะไรบ้าง จงอธิบายโดยละเอียด
7. การบำรุงรักษาระบบเกิดขึ้นเมื่อใด จงยกตัวอย่างประกอบ
8. เอกสารของโปรแกรม มีความสำคัญอย่างไร จงอธิบายให้เห็นจริง
9. ความผิดพลาดในการเขียนโปรแกรม มีอะไรบ้าง จงยกตัวอย่างประกอบ