บทที่ 6
การเขียนโปรแกรม

ลักษณะของการเขียนโปรแกรมที่ดีนั้น ควรจะมีองค์ประกอบดังนี้

1. มีส่วนของ Comment เพื่ออธิบายสิ่งต่าง ๆ ในโปรแกรม

2. มีการเขียนโปรแกรมให้อยู่ในลักษณะที่ง่ายแก่การตรวจสอบและทำความเข้าใจ ต้องอย่าลืมว่า "มนุษย์เราจะต้องเป็นผู้อ่านโปรแกรมก่อนคอมพิวเตอร์" ดังนั้นจึงต้องมีรูปแบบการเขียนที่ง่ายแก่การตรวจสอบ ความเข้าใจตรงกันทั้งผู้อ่านโปรแกรมกับคอมพิวเตอร์ซึ่งเป็นผู้ปฏิบัติการ

Comment แบ่งเป็น 3 ประเภท

1. **Prorolouge Comments**

แต่ละ program, subroutine or procedure (function) ควรจะมี statement เริ่มต้นบาง statement เพื่ออธิบายว่าจะทำงานอะไร โครงสร้างของ Comment จะประกอบด้วย

1. A description of What the program does.
2. Usage : How to call it or use it.
3. A list and explanation of the important variable or arrays.
4. Instruction to on input/output. List any files.
5. A list of subroutine used.
6. The name of any special scientific methods used, together with a reference in which more information can be found.
7. Some indication of how long it takes.
8. Amount of core needed
9. Special operating requirements.
10. Author
11. Date written

2. **Directory Comments**

ในกรณีที่โปรแกรมมีขนาดยาวมาก ควรจะสร้าง directory or table of contents in comment from at the beginning of the program. The table of contents จะประกอบด้วยชื่อ, location and function for each module.
หมายเหตุ ในแต่ละ module ควรจะมีการ label ชื่อของแต่ละ module หรือ comment ในหน้าที่ของตนเองด้วย

3. **Explanatory Comments**

จะเป็น Comment แทรกใน program เพื่ออธิบาย code บางประเภทซึ่งดูเข้าใจยาก เช่น Comment เกิดก่อนการวน loop หรือ Comment ก่อนหน้า If-test แต่การ Comment ในกรณีนี้ควรจะทำเมื่อจำเป็นจริงๆ เท่า ๆ นั้น ไม่เช่นนั้นจะเป็นการ Comment แบบพร่ำเพรื่อ (โดยเฉพาะ high level language ที่อ่านเข้าใจง่ายอยู่แล้ว)
ตัวอย่างการ Comment ไม่ก่อให้เกิดประโยชน์ เช่น

```
{Check If less than Zero}
   if x < 0 then _____
```

ควรจะเปลี่ยนเป็นรูปแบบนี้จะดีกว่า

```
{Perform Negative Account Processing of Total Costs
are less than Zero}
```

* programmer มักจะไม่ค่อยชอบ Comment เพราะ

1. ความสามารถทางภาษา
2. คิดว่าตัวเองจำทุกอย่างในโปรแกรมได้หมด
3. ไม่มีเวลา

Placement Comments

สร้าง

- box ของ comment line
- มีการเว้น blank line เพื่อให้อ่านง่าย
- บอกขอบเขตที่ comment line อธิบายว่าถึงคำสั่งใดในโปรแกรม
- indent comment line

Incorrect comments are worse than no comments at all

ข้อแนะนำในการเขียนโปรแกรม

1. เว้นช่องว่างให้ดูง่าย

   FOR I:=1,100 ---> FOR I := 1, 100

   1 + A*B ---> ทำก่อน

   1+A * B ---> เข้าใจยากกว่า

ในบางภาษา เช่น FORTRAN, COBOL อนุญาตให้ใช้ col. 73-80 ไว้เขียน Comment ได้ด้วย

Standard Style

ประโยชน์ของการทำให้เป็น standard เพื่อขจัดสิ่งต่อไปนี้

1. They might restric future growth and progress.
2. They might be tool restrictive for universal usage.
3. They might be tool cumbersome for universal use.

Comments

- easy to debug
- ป้องกันการหลงลืมในบางส่วน
- ทราบ version แต่ละครั้งที่ update
- แบ่งขั้นตอนการทำงาน
- อธิบายถึง routine ต่าง ๆ และการส่งค่า parameter
- การ Trace on ในบางคำสั่งซึ่งเป็น interactive จะกลายเป็น Comment เมื่อไม่ใช้คำสั่งนั้นแล้ว
- ช่วยนำทางผู้อ่านโปรแกรม

แบบฝึกหัด

1. Define the foolowing terms.
   (a) Prologue comments (b) Directory cemments
   (c) Explanatory comments (d) Indinting
2. Why should programs be readable?
3. How often should comments be insertes?
4. Name some obvious places to put comments.
5. Each program needs some general comments, What items should be included in these introductory comments?
6. How should variable names be chosen?
7. What rules can be used to producereadale abbreviations for variable name.
8. What considerations should be used in naming files?
9. What are the advantages of using standard abbreviations?
10. Where should blank lines be used?]
11. Why is an pleuse of aaces desirable?
12. Why should anddentification name be punched iln program cards?
13. Why shouldn't you sequence number a program by units?
14. Why shouldn't multiple statemenrs be plwfd on the same line?
15. In your programming language, where could alphabetized liss be used?
16. Find some examples in aprogram where parentheses will improve readable?
17. Modify one of your programs so that indenting lis used. Dose it improtereadable?

18. Can you think of any other generalized techniques tomake programs more readable?
19. Is there a statement in your programming language tocause an eject in the source listing?
20. What does the following program do? Criticise the program from a style point of view and then rewrite it.

```
45  READ(5,3@)x,y
    IF(X,GT,40)GOTO10
32  FORMAT(2F8.2)
    GOTO 21
10  R=Y*40+Y*1.5*(X-40)
 1  WRITE(6,32)R
    GOTO 45
        21  R=Y*X
            GOTO1
            END
```

21. What does the code below do?

FORTRAN

```
        DO 5 = 1,N
           DO 5 J = 1,N
5             XMAT(I,J) = (I/J) * (J/I)
```

After you figure out what this pieceof code does, recode it in a clearer manner. Hintfor non-FORTRAN programmers: I,J are integer variables that always result in integer results by truncaion if necessary. Find three eamples of tricky or cryptic coding and present them to the clas for recoding.

22. If yourprogramminglanguagehas no reserved words(i.e., FORTRAN,PL/1), write a small program, using variable names that that are usually for commands. See if someone. else can figure out what the program does.

23. Try reading some programs.Some sources ara program libraries, friends, and wastepaper baskets. Give such programs a grade (A to F) on style and readability. Were you able to learn anything by reading them?

24. Is computer programming an art or a science? Some people belive that it will always be an art; others believe that it is a science.After all,it is called computer science! Write a paper discussing the pros and cons of this question. To begin, decide on the difference beween art and science. A good place to start researching this topic is in Donald E. Knuth , "Computer Programming as an Art,"Communications of the ACM", December 1974.