บทที่ 2

ภาษาคอมพิวเตอร์กับการออกรายงาน

2.1 ภาษาโคบอล

เป็นภาษาที่จะเหมาะสมแก่การใช้งานประมวลในทางธุรกิจ จึงมีคำสั่งและสิ่งอำนวยประโยชน์ที่ช่วยในการแสดงผลอย่างมาก

2.1.1 คำสั่งที่เกี่ยวกับการแสดงผล

1) <u>WRITE</u> record-name [<u>FROM</u> identifier-1]

```
[ { BEFORE }             { integer { LINE  } }                                  ]
[ {        }  ADVANCING  {         { LINES } }  [ ; INVALID key imperative-  ]  ]
[ { AFTER  }             { TAB               }  [               statement    ]  ]
[                        { PAGE              }                                  ]
```

record-name เป็นชื่อระเบียน (record) ที่อยู่ใน Output file ซึ่งได้กำหนดที่ FILE SECTION ใน DATA DIVISION

identifier-1 เป็นชื่อระเบียนที่ต้องการย้ายข้อมูลมายัง record-name ก่อนแสดงข้อมูล

integer คือจำนวนเต็มบวกบอกจำนวนก่อน-หลังการพิมพ์

imperative statement เป็นคำสั่งประเภทไม่มีเงื่อนไข

จากคำสั่งนี้เครื่องจะนำข้อมูลจาก identifier-1 มายัง record-name หรือข้อมูลของ record เอง (ถ้าไม่มี FROM) เพื่อแสดงผลทางสื่อแสดงผล ที่กำหนดใน ENVIRONMENT DIVISION ทีละ 1 ระเบียน

เมื่อใช้ BEFORE จะทำการแสดงผลลัพธ์ก่อนแล้วจึงเลื่อนบรรทัดหรือหน้ากระดาษ

AFTER จะทำการแสดงผลลัพธ์หลังจากเลื่อนบรรทัดหรือหน้ากระดาษ

TAB มีผลให้แสดงผลลัพธ์ตามตำแหน่งมาตรฐานที่กำหนด

INVALID เมื่อข้อมูลไม่เป็นไปตามลักษณะของแฟ้มที่กำหนด จะทำตาม imperative-statement โดยไม่ทำคำสั่ง Write เช่น

ตัวอย่าง DATA DIVISION

```
Ø1  SHOW.
Ø2  NO              PIC  9(8).
Ø2  FILLER          PIC  X(12).
Ø2  NAME            PIC  X(3Ø).
Ø2  FILLER          PIC  X(5).
Ø2  OLD             PIC  9(2).
```

ใน PROCEDURE DIVISION.

```
WRITE SHOW AFTER ADVANCING 2 LINES.
```

พิมพ์ระเบียน เครื่องจะพิมพ์ข้อมูลในระเบียนชื่อ SHOW ตามเขตข้อมูลที่ระบุหลังจากนั้นจนเลื่อนไป 2 บรรทัด

2) DISPLAY

```
{identifier-1 }  [ identifier-2 ]                    [ UPON  mnemonic-name ]
{             }  [ ,            ] ...                [                     ]
{constant-1   }  [ constant-2   ]
```

เป็นการแสดงเนื้อหาของข้อมูลหรือค่าคงที่ทางจอภาพ

constant-1, constant-2 อาจจะเป็นค่าคงที่ตัวเลขจำนวนเต็มไม่มีเครื่องหมาย หรือตัวอักษรในเครื่องหมาย ' ' ก็ได้

ตัวอย่างเช่น

```
DISPLAY  'ERROR CODE',ID UPON CRT.
```

เครื่องจะแสดงข้อมูลทางหน้าจอ โดยปรากฏข้อความภายในเครื่องหมายคำพูดและค่าข้อมูลของ ID ถ้าสมมติให้ ID = 123 ดังนี้

```
ERROR CODE 123
```

DISPLAY SPACE.

จะเป็นการแสดงช่องว่าง 1 บรรทัด

DISPLAY ' '.

จะเป็นการแสดงช่องว่าง 1 ช่อง

2.1.2 ในส่วนที่อำนวยความสะดวกของภาษาโคบอลนั้น คือ การใช้สัญลักษณ์แทนข้อมูลและสัญลักษณ์ที่ใช้ในการคำนวณมีรูปแบบ ดังนี้

$$\left\{\begin{array}{l}\underline{\text{PICTURE}}\\ \\ \underline{\text{PIC}}\end{array}\right\} \text{ IS Character-string}$$

Character String เป็นส่วนที่แสดงรายละเอียดเกี่ยวกับขนาด ประเภทข้อมูลจะเป็นประเภท numeric, alphabetic, alphanumeric, alphanumeric edited หรือ numeric edited มีสัญลักษณ์ดังนี้

การใช้สัญลักษณ์แทนข้อมูลและสัญลักษณ์ที่ใช้ในการคำนวณ

ก. PICTURE A ใช้สำหรับแสดงข้อมูลที่เป็นตัวอักษร A ถึง Z หรือที่เรียกว่า Alphabetic และที่ว่าง (blank)

ข. PICTURE X ใช้สำหรับแสดงที่เป็นตัวเลข ตัวอักษรหรือเครื่องหมาย หรือที่เรียกว่า Alphanumeric นิยมใช้กันมาก

ค. PICTURE 9 ใช้กับข้อมูลที่เป็นตัวเลข (numeric) เท่านั้น เนื่องจากข้อมูลที่เป็นตัวเลขนั้นอาจจะเป็นจำนวนเต็ม ทศนิยมหรือเลขที่มีเครื่องหมายบวก หรือ ลบ

ง. PICTURE V แทนจุดทศนิยมในเขตข้อมูลที่เป็น input หรือ เขตข้อมูล ที่ใช้ในการคำนวณไม่นับเป็น 1 ตำแหน่ง data item และไม่ควรอยู่ทางขวามือสุด

จ. PICTURE S แทนเครื่องหมายเพื่อนำมาคำนวณด้วย ใช้เฉพาะเขตข้อมูลที่เป็นตัวเลขไม่นับเป็น 1 ตำแหน่ง data item โดยต้องปรากฏทางซ้ายมือของข้อมูลเท่านั้น

ฉ. PICTURE P แทนจุดทศนิยมในเขตข้อมูลที่เป็นตัวเลข โดยไม่นับเป็น 1 ตำแหน่งของ data item สามารถปรากฏทางซ้ายสุดหรือทางขวาสุดของข้อมูลได้โดยไม่ใช้ร่วมกับ '.' (จุด)

ตัวอย่าง

รายการ			คำอธิบาย	
			ข้อมูลสมมติ	ข้อมูลที่แสดงออกมา
02	FIRST-NAME	PIC A(10).	KWANCHAI	KWANCHAI
02	NAME	PIC X(10).	KWAN1234	KWAN1234
02	STUDENT-COUNT	PIC 9(6).	42313	042313

สัญลักษณ์ที่ใช้ในการพิมพ์

ก. PICTURE Z เป็นสัญลักษณ์ที่จะจัดอยู่ในจำพวก replacement ตัว Z จะทำหน้าที่เปลี่ยนเลขศูนย์ซึ่งอยู่ทางซ้ายมือสุดของจำนวนให้เป็นที่ว่างแทน

ข. PICTURE * เป็นสัญลักษณ์ในจำพวก Replacement ทำหน้าที่เปลี่ยนเลขศูนย์ที่อยู่ทางซ้ายมือสุดของจำนวนให้เป็น "*"

ค. PICTURE $ เป็นจำพวก fixed insertion และ replacement ในกรณีที่เป็น fixed insertion จะใช้เครื่องหมายดอลลาร์เพียงตัวเดียวซึ่งเขียนไว้ทางซ้ายมือสุด เมื่อพิมพ์ตัวเลขออกมาจะมีเครื่องหมาย $ ตรงตำแหน่งนั้นทางซ้าย

ในกรณีที่จะใช้เครื่องหมายดอลล่าร์แบบ replacement ให้เขียน "$" มากกว่า 1 ตัวติดกัน ใน character-string ซึ่งเครื่องหมายเหล่านี้จะทำหน้าที่ตัวเลขศูนย์ที่อยู่ทางซ้ายมือสุดของจำนวน และใส่เครื่องหมาย "$" หน้าตัวเลขซ้ายมือสุด

ง. PICTURE + หรือ - เป็นทั้งแบบ fixed insertion และ replacement

ในกรณีที่เป็น fixed insertion ให้เขียนเครื่องหมาย + หรือ - เพียงตัวเดียวไว้ข้างหน้าหรือข้างหลังของชุดอักษรใน PICTURE

ข้อสังเกต : ในกรณีใช้เครื่องหมาย "-" ในชุดอักษรใน PICTURE เมื่อข้อมูลเป็นบวกจะไม่พิมพ์เครื่องหมาย "+" ออกมา ถ้าใช้เครื่องหมาย "+" ในชุดตัวอักษรใน PICTURE จะพิมพ์ + เมื่อข้อมูลเป็น + ในกรณีที่เป็น replacement ให้ใช้เครื่องหมาย + หรือ - หลาย ๆ ตัว ในชุดอักษรของ PICTURE เครื่องหมายเหล่านี้จะตัดศูนย์ที่อยู่ทางซ้ายมือสุดของจำนวนออก แล้วแทนด้วยเครื่องหมาย "+" หรือ "-" เพียง 1 ตัว

จ. PICTURE ที่มี 0,B หรือ . แทรกอยู่ จัดอยู่ในพวก fixed insertion และใช้เมื่อต้องการพิมพ์อักษรเหล่านี้แทรกอยู่ในเลขจำนวนใดจำนวนหนึ่งตรงตำแหน่งที่กำหนด

ฉ. PICTURE ที่มีตัวอักษร DB หรือ CR กำกับ เมื่อข้อมูลตัวเลขมีเครื่อง

หมายเป็นลบ คอมพิวเตอร์จะพิมพ์อักษร "DB" หรือ "CR" ออกมาด้วย แต่ถ้าตัวเลขนั้นมีเครื่องหมายเป็นบวกจะไม่พิมพ์คำ "DB" หรือ "CR" ออกมา

ตัวอย่าง

PICTURE	ข้อมูลสมมุติ	ผลลัพธ์ที่พิมพ์ออกมา
ZZZZ	0000	
ZZZZ	0087	87
****	0000	****
****	0087	**87
$999.99	123^45	$123.45
$$$9999	001234	$1234
-99	22+	22
+99	22+	+22
-99	22-	-22
----	0000	
99B999B99	3645750	36 497 50
99,999	23456	23,456
XB0XB0	PQ	P 0Q 0
9.99DB	8^23-	8.23DB
$$99.99CR	45^67-	$45.67CR

ถึงแม้ว่าภาษาโคบอลจะมีสิ่งอำนวยความสะดวกมาก แต่เป็นภาษาที่มีระเบียบหรือกฎเกณฑ์ค่อนข้างมาก เมื่อเทียบกับภาษคอมพิวเตอร์อื่น ๆ ในการที่จะพิมพ์รูปแบบรายงานออกมานั้น ผู้ใช้จะต้องกำหนดระเบียนของข้อมูลที่จะพิมพ์ไว้ในส่วน DATA DIVISION ซึ่งถ้าในรายงานนั้นมี block ที่จะพิมพ์ เช่น Report Heading, Page Heading เป็นต้น จำนวนมากผู้ใช้ก็จะต้องเสียเวลาในการกำหนดใน DATA DIVISION และทำให้โปรแกรมใหญ่มาก อาจทำให้เกิดข้อจำกัดในเรื่องหน่วยความจำได้ และในการแก้ไขรูปแบบรายงานนั้นก็จะทำให้ยุ่งยากตามไปด้วย ในภาษาโคบอลบนเครื่องไมโคร-คอมพิวเตอร์นั้นไม่มีมีคำสั่งที่เกี่ยวข้องกับการออกรายงานที่กำหนดในส่วน REPORT SECTION โดยมีคำสั่งที่เกี่ยวข้องดังนี้

1) Heading เป็นตัวที่บอกว่า การออกรายงานในแต่ละหน้านั้นจะพิมพ์เริ่มต้นที่บรรทัดที่เท่าไร

2) First Detail เป็นตัวที่บอกว่า การออกรายงานในแต่ละหน้านั้นส่วนที่เป็น Detail line จะพิมพ์เริ่มต้นที่บรรทัดที่เท่าไรของหน้า

3) Last Detail เป็นตัวที่บอกว่า การยอกรายงานในแต่ละหน้านั้นส่วนที่เป็น Detail line จะพิมพ์ท้ายสุดที่บรรทัดที่เท่าไรของหน้า

4) Footing เป็นตัวที่บอกว่า การออกรายงานในแต่ละหน้านั้นบรรทัดสุดท้ายของแต่ละหน้านั้นคือบรรทัดที่เท่าไร

นอกจากนี้ยังมีการกำหนดว่าเขตข้อมูลไหนที่ทำหน้าที่เป็นตัวควบคุมการแบ่งข้อมูล ซึ่งการแบ่งข้อมูลจะแบ่งข้อมูลออกเป็นส่วนย่อย ๆ

จะเห็นได้ว่าภาษาโคบอลนี้ ผู้ใช้จำเป็นที่จะต้องทราบตำแหน่งของรายงานที่ต้องการจะพิมพ์ รวมทั้งต้องเขียนการทำงานของตัวควบคุมการจัดรายงาน เช่น Heading Footing ขึ้นมาเอง ซึ่งจะทำให้โปรแกรมซับซ้อนขึ้น และถ้าต้องการแก้ไขรูปแบบรายงานจำเป็นต้องแก้ไขในโปรแกรม

2.2 ภาษาฟอร์แทรน

เป็นภาษาที่ส่วนมากจะใช้ทางวิทยาศาสตร์และการคำนวณ โดยมีฟังก์ชั่นที่เกี่ยวกับการคำนวณทางวิทยาศาสตร์ จำนวนมากในการอำนวยความสะดวก

2.2.1 คำสั่งที่เกี่ยวกับการแสดงผล

1) Print*[,List]

List เป็นชื่อตัวแปร หรือ นิพจน์ ใช้พิมพ์ค่าของข้อมูลที่เก็บในตัวแปร หรือ นิพจน์ ทางเครื่องพิมพ์โดยที่ตัวแปรหรือ นิพจน์ จะต้องระบุอยู่ใน List ข้อมูลใน List อาจจะมีได้มากกว่า 1 ตัวหรือไม่มีก็ได้ ใช้จุลภาคคั่นระหว่างตัวแปรแต่ละตัว เช่น Print*,A,B,C

2) Write(u,f,err=m,end=n)list

Write(u,f,err=m,end=n)

u เป็นหมายเลขประจำอุปกรณ์นำข้อมูลออก เช่น 6 หมายถึงเครื่องพิมพ์

f เป็นหมายเลขคำสั่ง Format ซึ่งกำหนดรายละเอียดเกี่ยวกับการพิมพ์

m,n เป็นหมายเลขคำสั่งที่เครื่องจะย้ายไปทำ เมื่อเกิดการผิดพลาดจากการรับข้อมูล (Outputerror) และเมื่อหมดข้อมูล (end of file) ตามลำดับ

คำสั่ง Write ทั้ง 2 รูปแบบสามารถที่จะละ Err= และ End= ได้

คำสั่งนี้จะเป็นการแสดงข้อมูลที่ระบุไว้ใน List ทางอุปกรณ์แสดงผลหมายเลข u โดยมีรูปแบบตามที่ระบุไว้ในคำสั่ง Format หมายเลข f และใช้คำสั่งนี้คำสั่งเดียวสามารถแสดงข้อมูลได้หลาย ๆ ระเบียน โดยกำหนดด้วยจำนวนตัวแปรใน List และคำสั่ง Format โดยข้อมูลที่ต่อเนื่องกันจะถูกแสดงออกมาจนกว่าตัวแปรใน List ถูกใช้หมด

3) Write(u,err=m,end=n)List

เป็นคำสั่งที่จะมีการเคลื่อนย้ายข้อมูลโดยไม่มีการเปลี่ยนแปลงรูปแบบข้อมูลแต่อย่างใด โดยจำนวนข้อมูลที่ต้องการจะเคลื่อนย้ายต้องสอดคล้องกับจำนวนตัวแปรใน List โดยจะแสดงผลลัพธ์ออกมา 1 ระเบียน

2.2.2 คำสั่งที่เกี่ยวกับการกำหนดรายละเอียดเกี่ยวกับการพิมพ์

m format (s1,s2,...,sn)

m เป็นหมายเลขประจำคำสั่งที่ถูกระบุในคำสั่ง Read หรือ Write ที่ใช้ควบคู่กับคำสั่ง Format นี้

S1,S2...Sn หมายถึง field descriptor ซึ่งเป็นการกำหนดเขตข้อมูลต่าง ๆ ใน List ตามลำดับก่อนหลัง

ใน Format Specification นั้นจะแบ่งออกเป็นประเภทใหญ่ ๆ ได้ดังนี้

1) ตัวแปรประเภทตัวเลข (Numeric) จะมีรูปแบบหลายรูปแบบตามชนิดของตัวเลขนั้น ดังนี้

ก. F Format เป็นการกำหนดตัวสำหรับตัวแปรชนิด Real, Single Precision หรือ Double Precision

รูปแบบการเขียน

F w.d

W คือความกว้างของเขตข้อมูล

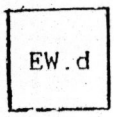
d คือจำนวนตัวเลขหลังจุดทศนิยม

ในกรณีที่ค่าที่แสดงออกมาไม่เต็มเขตข้อมูล เวลาแสดงค่าออกมาจะจัดไว้ชิดขวาของเขตข้อมูล แล้วส่วนที่เหลือก็จะเติมช่องว่างเข้าไปจนเต็มเขตข้อมูล ในกรณีที่มีมากกว่าเขตข้อมูล คอมพิวเตอร์จะแสดงเครื่องหมาย '*' แทนผลลัพธ์

ตัวอย่างเช่น

Format Description	ค่าที่อยู่ภายใน	ผลลัพธ์ที่แสดงออกมา
F 10.4	368.42	368.4200
F 6.4	4739.76	*******

ข. E - Format เป็นรูปแบบข้อมูลที่เป็นตัวเลขที่เป็นทศนิยมแบบยกกำลัง โดยมีรูปแบบ ดังนี้



ผลที่ได้ออกมาจากการแสดงผล จะต้องมีลักษณะประกอบด้วยสิ่งต่าง ๆ ตามลำดับต่อไปนี้

1. เครื่องหมายลบ (ในกรณีที่เป็นลบ)
2. ตัวเลข Ø และจุดทศนิยม
3. จำนวนเลขหลังจุดทศนิยม d หลัก
4. ตัวอักษร E
5. เครื่องหมายของเลขยกกำลัง (+ หรือ ช่องว่าง)
6. ค่าแสดงกำลังของตัวเลข 2 หลัก

เวลาแสดงผลออกมานี้ จะจัดชิดขวาของเขตที่กว้าง W หลัก

Format Description	ค่าที่อยู่ภายใน	ค่าที่แสดงออกมา
E 12.5	76.573	.76573E Ø2

ค. D-Format นั้น ในกรณีที่ค่าที่เราต้องการแสดงอยู่ในรูปของ Double precision และหลักการแสดงผลก็มีลักษณะเช่นเดียวกัน E-Format เพียงแต่เปลี่ยนตัวอักษร E ในข้อมูลให้เป็น D เท่านั้นเอง

ง. I-FORMAT มีรูปแบบ

เป็นการกำหนดสำหรับตัวแปรชนิดจำนวนเต็ม (Integer) เท่านั้นที่จะใช้ในการกำหนดรูปแบบนี้ได้ โดย W ระบุความกว้างของ field

ตัวอย่างเช่น

Format Description	ค่าที่อยู่ภายใน	ค่าที่แสดงออกมา
I6	+281	281
I6	-23261	-23261

2) ตัวแปรประเภทตัวอักษรมีรูปแบบ ดังนี้

A FORMAT ใช้สำหรับแสดงผลข้อมูลที่เป็นตัวอักษร โดยจะเก็บไว้ในตัวแปรชนิดจำนวนเต็ม หรือ ชนิดตัวอักษร โดยมีรูปแบบ

Format Description	ค่าที่อยู่ภายใน	ค่าที่แสดงออกมา
A1	AA	A
A7	ABCD	ABCD

3) การใช้ Nonrepeatable Edit Descriptor ในการแสดงผลข้อมูล

ก. การใช้ Apostrophe หรือ Character Constant โดยเครื่องจะนำกลุ่มตัวอักษรที่อยู่ภายใน ' ' ออกมาพิมพ์

ข. H FORMAT ใช้สำหรับการแสดงข้อความ หรือ ตัวอักษรโดยกำหนดตัวอักษรที่จะนำมาแสดงผลในรูปของ Hollerith Constant

ตัวอย่าง เช่น

4HNAME ผลลัพธ์ NAME

ค. T FORMAT เป็นการกำหนดตำแหน่งของข้อมูลที่จะแสดงผล

ง. การใช้ / ใช้ขึ้นบรรทัดใหม่ในการแสดงผล

จ. X-Format โดยมีรูปแบบ

เมื่อใช้ X-Format สำหรับการแสดงผลจะมีผลทำให้มีช่องว่าง (blank) n ช่องถูกแทรกลงไประหว่าง Output record โดยสามารถทำการกำหนดระยะห่างของข้อมูลที่จะพิมพ์ในการแบ่งแยกข้อมูลออกจากกัน

ตัวอย่างเช่น

คำสั่ง Format	Output
3 Format (1X,1HA, 4X, 2HBC)	A BC
7 Format (3X,4HABCD, 1X)	ABCD

การที่จะออกรายงานด้วยภาษาฟอร์แทรนนั้น ผู้ใช้จำเป็นที่จะต้องทราบตำแหน่งของรายงานเพื่อจะกำหนดในคำสั่ง FORMAT ถ้าต้องการที่จะใช้คำสั่งที่ควบคุมการจัดรายงานที่จะออกมานั้นจำเป็นต้องเขียนโปรแกรมในส่วนนี้ขึ้นมาเอง ซึ่งยุ่งยากและในการแก้ไขรูปแบบรายงานนั้นต้องเสียเวลาในการกำหนดตำแหน่งของรายงานใหม่และต้องไปแก้ไขในโปรแกรมด้วย

2.3 ภาษาซี

เป็นภาษาที่ง่ายต่อการใช้งาน สามารถนำมาเขียนโปรแกรมที่สลับซับซ้อนได้ดี

2.3.1 คำสั่งที่เกี่ยวกับการแสดงผล

1) printf(char *format,...);

format เป็นสตริงที่จะจัดการว่า ผลลัพธ์จะออกมาในรูปแบบไหน

printf เป็นฟังก์ชั่นที่นำค่าจากโปรแกรมส่งมาให้ผู้ใช้โดยผ่านทาง standard output ตัวอย่างเช่น

```
printf("This is Computer Cost $%d",Cost);
```

ในการพิมพ์เว้นบรรทัด 1 บรรทัด ก็จะทำการส่งเลขฐานสิบหกที่มีค่า ØxØD และ ØxØA (carriage return, line feed ตามลำดับ) และในการพิมพ์ค่าตัวแปรที่เป็นตัวเลขหรือ สตริง จะใช้ format specification เข้ามาใช้ในการพิมพ์ค่าออกมา

2) putchar (argument)

เป็นฟังก์ชั่นสำหรับทำการส่งข้อมูล 1 ตัวอักษรจากโปรแกรมเพื่อไปแสดงผลบนจอภาพ

argument เป็นตัวแปรโดยจะเก็บค่าของตัวอักษรเพียง 1 ตัวหรืออาจจะเป็นฟังก์ชั่นได้ แต่ค่าของฟังก์ชั่นนั้นจะต้องได้ผลลัพธ์เพียง 1 ตัวอักษร นอกจากนั้นแล้วอาจจะเป็นค่า excape sequence

ตัวอย่างเช่น

putchar('p') ส่งข้อมูล 1 ตัวอักษรซึ่งอยู่ในเครื่องหมายคำพูด

putchar('\n') ส่งรหัสของ carriage return จะทำการขึ้นบรรทัดใหม่

putchar(ch) ตัวแปร ch ควรจะเป็นชนิดตัวอักษร

3) puts(list)

list จะเป็นตัวแปรหรือค่าคงที่ที่เป็นสตริง ใช้ในการพิมพ์สตริงออกทางจอภาพ โดยเพียงแต่ผ่านค่า address ของสตริงไปให้เท่านั้น ซึ่งจะสะดวกกว่าการใช้ printf() ซึ่งต้องกำหนดรูปแบบของการพิมพ์ด้วย ตัวอย่างเช่น

```
puts("What is your name");
```

2.3.2 format specification

จะใช้ในคำสั่ง printf มีรูปแบบดังนี้

%[flags][width][prcc][F/N/h/l]type

flag เป็นตัวกำหนดผลลัพธ์ว่า เป็น numeric signs, decimal point octal (ตัวเลขฐานแปด) และ hex prefixes (ตัวเลขฐานแปด)

width บอกจำนวนตัวอักษร ต่ำสุดที่จะพิมพ์

prec บอกจำนวนตัวอักษร สูงสุดที่จะพิมพ์สำหรับ integer หรือต่ำสุดที่จะพิมพ์ของ digit

size จะ default ขนาดของ argument โดย N= Near Pointer

F=Far pointer h = Short int, l = Long

โดยจะมีรายละเอียดของ flags ในการแสดงผลดังนี้

TYPE	INPUT ARGUMENT	FORMAT OF OUTPUT
d	เลขจำนวนเต็ม	Signed Decimal Integer
i	เลขจำนวนเต็ม	Signed Decimal Integer
o	เลขจำนวนเต็ม	Unsigned Decimal Integer
u	เลขจำนวนเต็ม	Unsigned Decimal Integer
x	เลขจำนวนเต็ม	Unsigned Hexadecimal Integer (a,b,c,d,f)
X	เลขจำนวนเต็ม	Unsigned Hexadecimal Integer (A,B,C,D,E,F)
f	ตัวเลขทศนิยม	Signed Value รูปแบบ [-]dddd.dddd
e	ตัวเลขทศนิยม	Signed Value รูปแบบ [-]d.dddd e [+/-]ddd
E	ตัวเลขทศนิยม	เหมือน e แต่ใช้ E สำหรับ Exponent
c	ตัวอักษร	ตัวอักษร 1 ตัว
s	สตริง	พิมพ์ตัวอักษรจนกระทั่งพบ '\0'

ภาษาซีมีคำสั่งที่จะให้ออกในตำแหน่งของจอก็จริง แต่ก็จำเป็นที่จะต้องทราบตำแหน่งรายงานที่จะออก เพราะไม่สามารถมองเห็นภาพรวมของรายงานได้ ซึ่งต้องใช้เวลามาก และไม่มีคำสั่งที่เกี่ยวกับการควบคุมการจัดรายงานเช่น Heading Footing เป็นต้น ซึ่งจำเป็นต้องเขียนโปรแกรมขึ้นมาตรวจสอบเองในจุดนี้ ซึ่งก็ยุ่งยากพอสมควร

2.4 ภาษาเบสิก

เป็นภาษาที่โครงสร้างของภาษามีความคล้ายคลึงกับสูตรทางพีชคณิต และขยายให้สมบูรณ์ด้วยคำในภาษาอังกฤษ เช่น Let, Read เป็นต้น จะเหมาะสำหรับใช้แก้ปัญหาทางวิทยาศาสตร์และคณิตศาสตร์ได้ดี

2.4.1 คำสั่งที่เกี่ยวกับการแสดงผล

1) PRINT

PRINT [{ ตัวแปร / "ข้อความ" } [{ , / ; } { ตัวแปร / "ข้อความ" }]]

เป็นคำสั่งที่ใช้แสดงผลลัพธ์หรือข้อความที่ต้องการออกสู่จอภาพ

- หากละตัวแปรหรือข้อความออก เครื่องจะพิมพ์ว่าทิ้งบรรทัด (ใช้ในการเว้นบรรทัด)
- ตัวแปร จะเป็นตัวแปรชนิดใดก็ได้
- ข้อความ ถ้าเป็นข้อความสตริงก์ ต้องอยู่ในเครื่องหมายคำพูด
- การพิมพ์ผลลัพธ์ที่เป็นตัวเลขนั้น ถ้าเลขนั้นเป็นเลขจำนวนลบ จะพิมพ์เครื่องหมายลบนำหน้า แต่ถ้าเป็นจำนวนบวกจะเว้นช่องว่างข้างหน้าเลขนั้นไว้ 1 ช่องด้วย
- ถ้าใช้เครื่องหมาย ; เชื่อมระหว่างตัวแปรหรือข้อความชุดแรกกับตัวแปรหรือข้อความในชุดถัดไป จะมีผลให้ค่าของตัวแปรหรือข้อความชุดหลังถูกพิมพ์ต่อเนื่องจากชุดแรก
- ถ้าใช้เครื่องหมาย , เชื่อม จะทำให้จอภาพถูกแบ่งออกเป็นส่วนส่วนละ 14 คอลัมน์ เพื่อปรากฏค่า

ตัวอย่างเช่น

```
X=4.2, Y= -5, Z=90
PRINT X, Y, Z
```

ผลลัพธ์

```
4.2              -5              90
```

หากใช้เครื่องหมาย ; คั่นระหว่างตัวแปรหรือข้อความในชุดแรกกับชุดถัดไปจะมีผลให้พิมพ์ต่อเนื่องกัน

ตัวอย่างเช่น

```
PRINT X;Y;Z;"BCC"
```

ผลลัพธ์

```
4.2 -5 90 BCC
```

ถ้าตอนท้ายของตัวแปรหรือข้อความไม่มีเครื่องหมายอะไรเลย การพิมพ์ครั้งต่อไปจะขึ้นบรรทัดใหม่เสมอ

ผู้ใช้สามารถกำหนดตำแหน่งพิมพ์ของข้อมูลได้ตามที่ต้องการ โดยใช้ฟังก์ชั่น TAB ซึ่งการใช้ฟังก์ชั่นนี้ สามารถใช้ได้กับคำสั่ง PRINT และ LPRINT เท่านั้น

ตัวอย่างเช่น

```
PRINT TAB(20) A$ TAB(40)N TAB(50)K
```

เครื่องจะพิมพ์ค่าของตัวแปร A$ ที่ตำแหน่งคอลัมน์ 20 พิมพ์ค่าตัวแปร N ที่ตำแหน่งคอลัมน์ 40 และพิมพ์ค่าตัวแปร K ที่ตำแหน่งคอลัมน์ 50

2) PRINT USING

PRINT USING รูปแบบ ; { ตัวแปร / "ข้อความา" } [{ , / ; } { ตัวแปร / "ข้อความา" }]

เป็นคำสั่งที่ใช้แสดงผลลัพธ์หรือข้อความออกสู่จอภาพในรูปแบบที่กำหนด รูปแบบหมายถึงรูปแบบของการพิมพ์ ซึ่งแบ่งออกได้ดังนี้

ก. สำหรับพิมพ์ข้อความสตริงก์

"!" หมายถึงให้พิมพ์เฉพาะตัวอักขระตัวแรกของข้อความสตริงก์

"\ช่องว่าง\" หมายถึงให้พิมพ์ข้อความสตริงก์ออกมาจำนวนตัวเท่ากับจำนวนช่องว่าง +2 (รวมทั้ง 2 อันด้วย) ตัว โดยที่ข้อความนั้นจะถูกพิมพ์ชิดซ้าย ที่เหลือจะเป็นช่องว่าง แต่ถ้าข้อความนั้นยาวกว่าส่วนที่เกินจะถูกตัดทิ้ง เช่น PRINT USING "\ \" จะพิมพ์ออกมา 5 ตัว ช่องว่าง 3 ช่อง

ตัวอย่างเช่น

```
10 A$ = "COMPUTER" : B$ = "BCC"
20 PRINT USING "!"; A$;B$
30 PRINT USING "\   \";A$;B$
40 PRINT USING "\         \";A$;B$
```

ผลลัพธ์

```
CB
COMPBCC
COMPUTER  BCC
```

"&" หมายถึง ให้พิมพ์ข้อความเหมือนที่กำหนดทุกประการ

ตัวอย่างเช่น

```
10 A$ = "COMPUTER' ; B$ = "BCC"
20 PRINT USING "!";A$;
30 PRINT USING "&";B$
```

ผลลัพธ์

```
CBBC
```

ข. สำหรับพิมพ์ตัวเลข

ใช้ระบุจำนวนตำแหน่งของตัวเลขที่ต้องการพิมพ์ออก โดยกำหนด # จำนวนตัวเท่ากับจำนวนตำแหน่งที่ต้องการ

- ถ้ากำหนดจำนวน # มากกว่าตำแหน่งของตัวเลข ตัว

เลขจะพิมพ์ชิดขวา ตำแหน่งที่เหลือทางซ้ายมือจะเป็นช่องว่าง

ตัวอย่างเช่น

```
PRINT USING "##.##"; .78
```

ผลลัพธ์ 0.78

```
PRINT USING "##.## "; 10.2, 5.3, 66.789, .234
```

ผลลัพธ์ 10.20 5.30 66.79 0.23

ช่องว่างที่เพิ่มเข้าไปในรูปแบบจะทำให้แต่ละชุดของตัวเลขพิมพ์ห่างกัน ถ้ามีเครื่องหมาย + ตอนหน้าหรือตอนท้ายของรูปแบบจะทำให้พิมพ์เครื่องหมายของตัวเลขนั้น ๆ ตอนหน้าหรือตอนท้ายของตัวเลขนั้น ถ้ามีเครื่องหมาย - ตอนท้ายของรูปแบบ จะทำให้ตัวเลขที่ออกมาจะพิมพ์เครื่องหมายลบต่อท้ายตัวเลขนั้น

ตัวอย่างเช่น

```
PRINT USING "##.##- "; -68.95, 22.449, -7.01
```

ผลลัพธ์ 68.95- 22.45 7.01-

** ถ้ามีเครื่องหมายดอกจันทน์คู่นี้อยู่หน้ารูปแบบ เครื่องจะพิมพ์ * แทนที่ ช่องว่างด้านหน้าของตัวเลขที่มีตำแหน่งน้อยกว่าจำนวน # โดยนับ ** รวมเป็นจำนวนตำแหน่งของตัวเลขด้วย

ตัวอย่างเช่น

```
PRINT USING "****.# "; 12.39, -0.9, 1765.1
```

ผลลัพธ์ **12.4 **-0.9 1765.1

$$ ถ้ามีเครื่องหมาย dollar คู่ อยู่หน้ารูปแบบ เครื่องจะพิมพ์ $ ตอนหน้าของตัวเลข

ตัวอย่างเช่น

```
PRINT USING "$$####.##"; 456.78
```

ผลลัพธ์ $456.78

**$ ถ้ามีเครื่องหมายนี้อยู่หน้ารูปแบบ จะเกิดผลเหมือนกับใช้ ** และ $$ รวมกัน

ตัวอย่างเช่น

```
PRINT USING "**$###.##"; 2.34
```

ผลลัพธ์ **$2.34

, ใช้เขียนไว้ทางซ้ายมือของจุดทศนิยมในรูปแบบ เพื่อให้เครื่องพิมพ์เครื่องหมาย , ในทุก ๆ 3 ตำแหน่ง ของตัวเลขที่อยู่ทางซ้ายของจุดทศนิยม

ตัวอย่างเช่น

```
PRINT USING "####,.##"; 1234.5
```

ผลลัพธ์ 1,234.50

แต่ถ้าใช้ , ไว้ตอนท้ายของรูปแบบ เครื่องจะพิมพ์เครื่องหมายนี้ต่อท้ายตัวเลขนั้น

ตัวอย่างเช่น

```
PRINT USING "####.##,"; 1234.5
```

ผลลัพธ์ 1234.50,

^^^^ ถ้ามีเครื่องหมายนี้ตอนท้ายของรูปแบบ เครื่องจะพิมพ์ตัวเลขนั้นออกมาในรูปของเลขยกกำลัง (E+XX)

ตัวอย่างเช่น

```
PRINT USING "##.##^^^^"; 234.56
```

ผลลัพธ์ 2.35E+02

```
PRINT USING ".###^^^^-"; -88888
```

ผลลัพธ์ .889E+05-

ถ้ามีเครื่องหมายขีดเส้นใต้ (underscore) นี้อยู่ในรูปแบบ ตัวอักขระที่อยู่ต่อจากเครื่องหมายนี้จะถูกพิมพ์ออกมาด้วย

ตัวอย่างเช่น

```
PRINT USING "_!##.##_!"; 12.34
```

ผลลัพธ์ !12.34!

หากต้องการพิมพ์เครื่องหมายขีดเส้นใต้ออกมาด้วย ให้ใช้รูปแบบ "__"

หากความยาวของตัวเลขมากกว่ารูปแบบที่กำหนด จะปรากฏเครื่องหมาย % นำหน้าตัวเลขนั้น

ตัวอย่างเช่น

```
PRINT USING "##.##"; 111.22
```

ผลลัพธ์ %111.22

```
PRINT USING ".##"; .999
```

ผลลัพธ์ %1.00

ถ้ามีการกำหนดรูปแบบสำหรับตัวเลขยาวเกิน 24 ตัวจะเกิดข้อผิดพลาด "Illegal Function Call Error"

3) LPRINT, LPRINT USING

เป็นคำสั่งที่ให้แสดงผลลัพธ์หรือข้อความออกทางเครื่องพิมพ์ การใช้ LPRINT และ LPRINT USING เหมือนกับการใช้คำสั่ง PRINT และ PRINT USING เพียงแต่แสดงผลออกทางเครื่องพิมพ์ คำสั่ง LPRINT จะพิมพ์ออกเครื่องพิมพ์ในขนาด 80 ตัวต่อ 1 บรรทัด หากต้องการจำนวนตัวที่แตกต่างจากนี้ให้กำหนดได้จากคำสั่ง

```
WIDTH "LPT1:", n
```

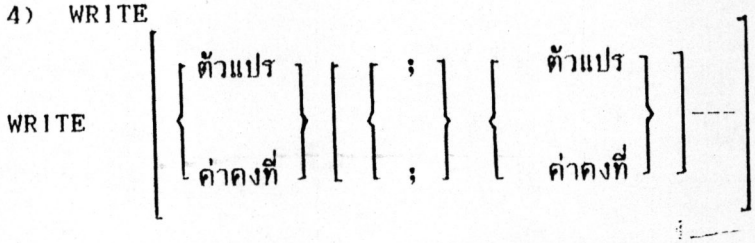
สำหรับ IBM เมื่อ n คือจำนวนตัวที่ต้องการใน 1 บรรทัด

ตัวอย่างเช่น

```
10 X = 45
20 LPRINT
30 LPRINT "This is an example ";
40 LPRINT "of output continuing on ";
50 LPRINT "the same line, followed by ";
60 LPRINT "formatted data. "
70 LPRINT USING "##.##  "; 20.4444, 99, 1 ,X
```

ผลลัพธ์ This is an example of output continuing on the same line, followed by formatted data.
20.44 99.00 1.00 45.00

4) WRITE



เป็นคำสั่งที่ให้พิมพ์ข้อความหรือผลลัพธ์ออกสู่จอภาพ ตัวแปรเป็นได้ทั้ง ค่าคงที่ตัวเลขและค่าคงที่ตัวอักษรและอักขระพิเศษ ถ้าไม่ใส่ตัวแปรและค่าคงที่จะพิมพ์ว่างทั้งบรรทัด เมื่อใช้คำสั่ง WRITE ข้อมูลที่พิมพ์ออกสู่จอภาพจะมีเครื่องหมาย , ปรากฏ เพื่อแบ่งข้อมูลแต่ละตัว และข้อมูลที่เป็นค่าคงที่สตริงก์ จะปรากฏเครื่องหมายคำพูดออกมาด้วย

การใช้เครื่องหมาย , หรือ ; เชื่อมระหว่างตัวแปรหรือค่าคงที่ชุดแรกกับชุดถัดไป ให้ผลเช่นเดียวกัน

การแสดงผลค่าคงที่ หรือตัวแปรที่เป็นตัวเลขจำนวนมาก จะไม่เว้นว่าง 1 ช่องทางด้านหน้าของตัวเลขนั้น

ตัวอย่างเช่น

```
10  A = 80 : B = 90 : C$ = "THAT'S ALL"
20  WRITE A,B,C$
```

ผลลัพธ์ 80,90,"THAT'S ALL"

ในภาษาเบสิคนี้ จะเห็นได้ว่าคำสั่งเหล่านี้ในการที่จะพิมพ์ออกมาจำเป็นต้องทราบตำแหน่งเพื่อที่จะได้กำหนดในรูปแบบการพิมพ์ ซึ่งจะยุ่งยากมาก เพราะจำเป็นต้องเขียนผังออกรายงานขึ้นมาก่อนเพื่อที่จะได้ทราบรูปแบบพอเป็นที่พอใจแล้วก็ถึงจะนับตำแหน่งที่จะพิมพ์ ในการที่จะแก้ไขรูปแบบ โดยจะต้องเข้าไปแก้ไขในตัวโปรแกรมและไม่มีคำสั่งที่ควบคุมรูปแบบรายงานที่ออกมา จำเป็นต้องเขียนโปรแกรมขึ้นมาตรวจสอบเอง

จากภาษาต่าง ๆ เหล่านี้ จะเห็นได้ว่า ทุกภาษาไม่มีคำสั่งที่เกี่ยวข้องกับการควบคุมการออกรูปแบบรวยงาน ซึ่งในการเขียนโปรแกรมต้องเขียนโปรแกรมตรวจสอบขึ้นมาเอง ทำให้ไม่สะดวกเท่าที่ควร และจำเป็นที่จะต้องทราบตำแหน่งที่ต้องการของข้อมูล

ซึ่งจำเป็นต้องการแก้ไขรูปแบบรายงานจำเป็นต้องไปรื้อโปรแกรมที่เขียน ซึ่งถ้าโปรแกรมใหญ่ก็จะทำให้ยุ่งยากมากในการแก้ไข สิ่งเหล่านี้จำเป็นที่จะต้องศึกษาเผื่อให้โปรแกรมอรรถประโยชน์สำหรับออกรายงานทำให้ผู้ใช้เกิดความสะดวกมากยิ่งขึ้น