บทที่ 6 โครงสร้างและไวยากรณ์ภาษาซี

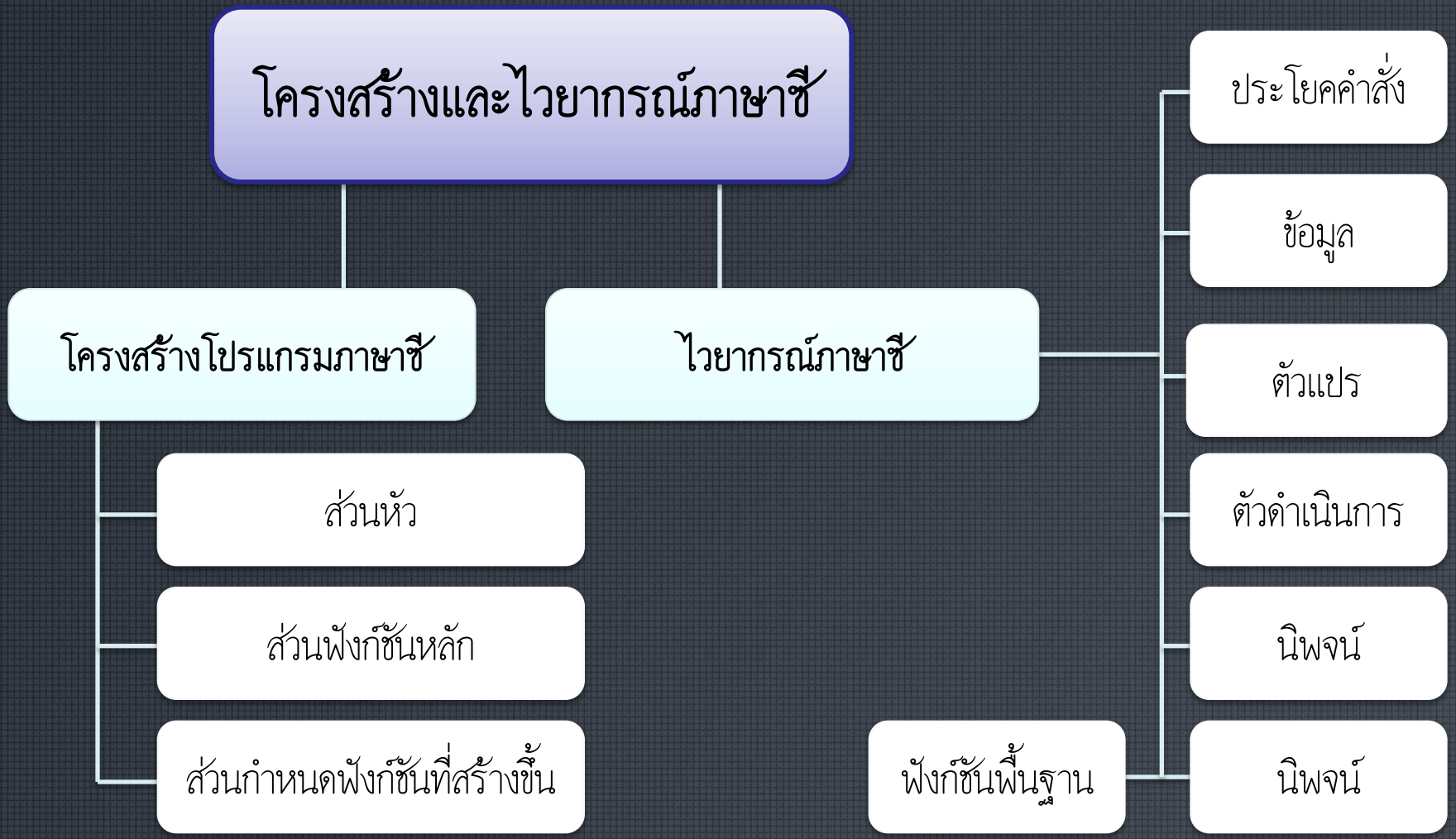
สาระการเรียนรู้

1. โครงสร้างโปรแกรมภาษาซี
2. ประโยคคำสั่งในภาษาซี
3. ข้อมูล ค่าคงที่และตัวแปร
4. ตัวดำเนินการและนิพจน์
5. ฟังก์ชันพื้นฐาน

สมรรถนะการเรียนรู้

1. อธิบายส่วนประกอบของโครงสร้างภาษาซีได้
2. จำแนกคำสั่งในภาษาซีได้
3. บอกลักษณะการเข้าถึงข้อมูลได้
4. ประกาศค่าคงที่ในโปรแกรมในภาษาซีได้
5. จำแนกลักษณะข้อมูลพื้นฐานได้
6. ประกาศตัวแปรในโปรแกรมภาษาซีได้
7. จำแนกชนิดของตัวแปรได้
8. บอกข้อกำหนดในการตั้งชื่อตัวแปรได้
9. บอกวิธีการเปลี่ยนประเภทตัวแปรได้
10. จำแนกประเภทของตัวดำเนินการในภาษาซีได้
11. เขียนนิพจน์ในภาษาซีได้
12. อธิบายการใช้ฟังก์ชันพื้นฐานในภาษาซีได้
13. เขียนโปรแกรมภาษาซีรับข้อมูลและแสดงผลข้อมูลได้

แผนผังความคิด (Mind Mapping) ของหน่วยการเรียนรู้



โครงสร้างโปรแกรมภาษาซี

โปรแกรมที่เขียนด้วยภาษาซี มีลักษณะโครงสร้างแบ่งออกเป็นส่วน ๆ ดังนี้

1.ส่วนคำสั่งพรี-โพรเซสเซอร์ (Pre-processor Command)

2.ส่วนประกาศ (Global Declarations)

3.ส่วนฟังก์ชันหลัก (Main Function)

4.ส่วนกำหนดฟังก์ชันที่สร้างขึ้น (User Defined Functions)

ส่วนหัว (Header)

เป็นส่วนที่อยู่ส่วนแรกของโปรแกรม อยู่ก่อนฟังก์ชันหลัก (Main Function) ในส่วนยังมีองค์ประกอบที่สำคัญ ได้แก่ ส่วนคำสั่งพรี-โพรเซสเซอร์ ส่วนประกาศตัวแปรแบบโกลบอลและประกาศฟังก์ชันที่สร้างขึ้นใช้เอง

1. ส่วนคำสั่งพรี-โพรเซสเซอร์ (Pre-processor Command)
2. ส่วนประกาศ (Global Declarations)

ประโยคคำสั่งพรีโพรเซสเซอร์

- #include Include text from a file
- #define Define a macro
- #ifdef Test if a symbol is defined
- #ifndef Test if a symbol is not defined
- #if Test if compile-time condition hold
- #elif Same elseif
- #endif End a pre-processor conditional
- #line Give a line number for compile message
- #pragme Implementation dependent directive
- #undef Undefine a macro

ตัวอย่างการใช้คำสั่งพรีโพรเซสเซอร์

#include

ใช้สำหรับสั่งให้คอมไพเลอร์ นำไฟล์ที่ระบุเข้ามารวมกับโปรแกรมก่อนที่จะทำการแปลโปรแกรม สามารถเขียนได้ 2 รูปแบบ คือ

#include <files.h>

โปรแกรมจะเริ่มค้นหาไฟล์จากไดเร็กทอรี่ที่กำหนด เช่น

#include <conio.h>

#include <stdio.h>

#include "files.h"

โปรแกรมจะเริ่มค้นหาไฟล์จากไดเร็กทอรี่ปัจจุบันก่อน ถ้าหาไม่พบก็จะไปค้นหาในไดเร็กทอรี่ที่กำหนด เช่น

#include "conio.h"

#include "stdio.h"

ตัวอย่างการใช้คำสั่งพรีโพเซสเซอร์

#define

ใช้สำหรับประกาศและกำหนดค่าคงที่

รูปแบบ

#define ชื่อค่าคงที่ ค่าที่กำหนด

เช่น

```
#define PI  3.1415926

#define NAME "Nuntchayathorn"
```

ส่วนประกาศ (Global Declarations)

อยู่ในส่วนหัว (Header) ของโปรแกรมที่อยู่ถัดจากส่วนคำสั่งพรี-โพรเซสเซอร์ เป็นส่วนที่ใช้ในการประกาศตัวแปรแบบโกลบอล (Global Variable) หรือ ประกาศฟังก์ชันขึ้นใช้เองในโปรแกรม User-Defined Function

- Global Variable Declaration
- User-Defined Function Declaration

รูปแบบการประกาศฟังก์ชัน

- void function_name();
- void function_name(arguments);
- return_type function_name();
- return_type function_name(arguments);

ส่วนฟังก์ชันหลัก (Main Function)

เป็นส่วนที่กำหนดการทำงานหลัก เมื่อสั่งรันโปรแกรมจะทำงานในส่วนนี้ทันที สามารถเขียนได้ 2 รูปแบบ คือ

- รูปแบบของโปรแกรมหลัก

```
void main( ) {
        local variable declaration;
        statements;
      ...
}
```

- รูปแบบของฟังก์ชันหลัก

```
int main( ) {
        local variable declaration;
        statements;
      ...
        return 0;
}
```

ส่วนกำหนดฟังก์ชันที่สร้างขึ้น (User Defined Functions)

ในภาษาซีสามารถเขียนฟังก์ชัน ได้ 4 รูปแบบ

- แบบไม่มีการรับค่าและส่งค่ากลับ
- แบบมีการรับค่าแต่ไม่มีการส่งค่ากลับ
- แบบไม่มีการรับค่าแต่มีการส่งค่ากลับ
- แบบมีการรับค่าและส่งค่ากลับ

ส่วนกำหนดฟังก์ชันที่สร้างขึ้น (User Defined Functions)

1. แบบไม่มีการรับค่าและส่งค่ากลับ

```
void function_name( ) {
        local variable declaration;
        statements;
}
```

ส่วนกำหนดฟังก์ชันที่สร้างขึ้น (User Defined Functions)

2. แบบมีการรับค่าแต่ไม่มีการส่งค่ากลับ

```
void function_name( parameter ) {
        local variable declaration;
        statements;
}
```

ส่วนกำหนดฟังก์ชันที่สร้างขึ้น (User Defined Functions)

3. แบบไม่มีการรับค่าแต่มีการส่งค่ากลับ

```
return_type function_name( ) {
        local variable declaration;
        statements;
        var = ... ;
        return var;
}
```

ส่วนกำหนดฟังก์ชันที่สร้างขึ้น (User Defined Functions)

4. แบบมีการรับค่าและส่งค่ากลับ

```
return_type function_name( parameter ) {
        local variable declaration;
        statements;
        var = ... ;
        return var;
}
```

ประโยคคำสั่ง (Statements) ในภาษาซี

ประโยคคำสั่งในภาษาซีมี 2 ลักษณะ คือเป็นคำสั่งและฟังก์ชัน ประโยคคำสั่งสามารถใช้ได้ทันที ส่วนฟังก์ชันจะต้องมีการนำไลบรารีไฟล์ที่ได้นิยามฟังก์ชันต่าง ๆ ไว้ โดยการ include ไฟล์เหล่านั้นไว้ในส่วนหัวโปรแกรม ประโยคคำสั่ง (Statements) ทุกคำสั่งจะพิมพ์ด้วยอักษรตัวเล็กทั้งหมด และต้องจบคำสั่งทุกคำสั่งด้วยเครื่องหมาย ; (Semi-colon) เสมอ อาจมีข้อยกเว้น ที่ไม่ต้องใส่เครื่องหมาย ; (Semi-colon) เช่น กรณีที่คำสั่งนั้น มีขอบเขตคำสั่ง { } อาจไม่ต้องจบคำสั่งด้วยเครื่องหมาย ; ก็ได้ (จะใส่ ; หรือไม่ใส่ก็ได้)

คำสั่งในภาษาซี

คำสั่งในภาษาซี สามารถแบ่งเป็นกลุ่มคำสั่งต่าง ๆ ได้ดังนี้ คือ

- คำสั่งประกาศตัวแปร (Variables Declaration)
- คำสั่งกำหนดค่า (Assignments)
- คำสั่งเงื่อนไข และ คำสั่งเลือกทำ (Condition)
- คำสั่งวนรอบการทำงาน (Looping)
- ฟังก์ชันแสดงผล (Output)
- ฟังก์ชันรับข้อมูล (Input)
- ฟังก์ชันการคำนวณ ตรวจสอบและแปลงข้อมูลต่าง ๆ
- ฟังก์ชันเกี่ยวกับสตริง (String functions)
- ฟังก์ชันเกี่ยวกับวันและเวลา และเกี่ยวกับเสียง
- ฟังก์ชันเกี่ยวกับการทำงานในโหมดกราฟิก
- ฟังก์ชันดำเนินการเกี่ยวกับไฟล์

คำสั่งประกาศตัวแปร (Variables Declaration)

รูปแบบ type var_list;

เช่น

```
int a,b,c;
float f;
double d,ans;
char ch;
char str[30];
```

เป็นต้น

คำสั่งกำหนดค่า (Assignments)

รูปแบบ var = Expression;

เช่น

```
a = 0;
b = a;
n= 10;
sum = n*(n+1)/2;
```

เป็นต้น

คำสั่งเงื่อนไข และ คำสั่งเลือกทำ (Condition)

- if
- if else
- switch/case

คำสั่งวนรอบการทำงาน (Looping)

- do while
- while
- For

ฟังก์ชันแสดงผล (Output)

ฟังก์ชันที่ถูกนิยามไว้ในไฟล์ conio.h, stdio.h ได้แก่

- printf( )
- putchar( )
- puts( )
- cprintf( )
- clrscr( )
- clreol( )
- gotoxy( )

เป็นต้น

ฟังก์ชันรับข้อมูล (Input)

ฟังก์ชันที่ถูกนิยามไว้ในไฟล์ conio.h, stdio.h ได้แก่

- scanf( )
- getch( )
- getche( )
- getchar( )
- gets( )

เป็นต้น

ฟังก์ชันการคำนวณ ตรวจสอบและแปลงข้อมูลต่าง ๆ

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ stdlib.h ได้แก่

- abs( )
- atoi( )
- atof( )
- atol( )
- rand( )
- randomize( )
- qsort( )
- max( )
- min( )
- strtod( )
- strtol( )
- stetold( )
- swap( )
- system( )
- time( )

เป็นต้น

ฟังก์ชันการคำนวณ ตรวจสอบและแปลงข้อมูลต่าง ๆ

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ ctype.h ได้แก่

- isalnum( ) เป็นต้น
- isalpha( )
- isdigit( )
- islower( )
- isupper( )
- tolower( )
- toupper( )

ฟังก์ชันการคำนวณ ตรวจสอบและแปลงข้อมูลต่าง ๆ

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ math.h ได้แก่

- exp( )
- sqrt( )
- pow( )
- sin( )
- cos( )
- tan( )
- log( )
- log10( )
- ceil( )
- floor( )
- fabs( )

เป็นต้น

ฟังก์ชันเกี่ยวกับสตริง (String Functions)

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ string.h ได้แก่

- strlen( )
- strcpy( )
- strcat( )
- strcmp( )

เป็นต้น

ฟังก์ชันเกี่ยวกับวันและเวลา และเกี่ยวกับเสียง

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ dos.h ได้แก่

- gettime( )
- getdate( )
- settime( )
- setdate( )
- sound( )
- delay( )
- nosound( )

เป็นต้น

ฟังก์ชันเกี่ยวกับการทำงานในโหมดกราฟิก

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ graphics.h ได้แก่

arc( )
bar( )
bar3d( )
circle( )
cleardevice( )
closegraph( )
drawpoly( )
ellipse( )
fillellipse( )
fillpoly( )
floodfill( )
getcolor( )
getimage( )
getmaxx( )
getmaxy( )
getx( )
gety( )
imagesize( )
initgraph( )
line( )
linerel( )
lineto( )
moverel( )
moveto( )
outtext( )
outtextxy( )
pieslice( )
putimage( )
putpixel( )
rectangle( )
setbkcolor( )
setcolor( )
setfillpattern( )
setfillstyle( )
settextstyle( )

เป็นต้น

ฟังก์ชันดำเนินการเกี่ยวกับไฟล์

ฟังก์ชันต่าง ๆ ที่ถูกนิยามในไฟล์ stdio.h ได้แก่

- fopen( )
- fgetc( )
- fgets( )
- fputs( )
- fputc( )
- fscanf( )
- fprintf( )
- fseek( )
- feof( )
- fclose( )

เป็นต้น

การเขียนหมายเหตุในโปรแกรมภาษาซี

- **รูปแบบการเขียนหมายเหตุ บรรทัดเดียว (Single-Line Comment)**

  โดยพิมพ์เครื่อง Double slash // หน้าข้อความที่ทำเป็นหมายเหตุ

รูปแบบ

```
// คำอธิบาย หรือ หมายเหตุ เขียนไว้ในบรรทัดเดียว
```

- **รูปแบบการเขียนหมายเหตุแบบหลายบรรทัด (Multi-Line Comment)**

  โดยขึ้นต้นด้วยเครื่องหมาย /* และสิ้นสุดด้วยเครื่องหมาย */

รูปแบบ

```
/*
    คำอธิบาย หรือ หมายเหตุต่าง ๆ สามารถเขียนได้หลาย ๆ บรรทัด
*/
```

ข้อมูล (Data)

หมายถึง สิ่งที่เกี่ยวกับปัญหา หรือสิ่งที่นำมาใช้ในการเขียนโปรแกรม เมื่อนำข้อมูลเข้าสู่คอมพิวเตอร์ ข้อมูลจะถูกเก็บไว้ในส่วนของหน่วยความจำหลัก ในพื้นที่ที่ได้จองไว้สำหรับเก็บข้อมูล โดยมีการกำหนดชื่อในการเข้าถึงข้อมูลนั้น ซึ่งชื่อที่ใช้สำหรับอ้างอิงถึงข้อมูล มี 2 ลักษณะ คือ เป็นค่าคงที่ และตัวแปร

ค่าคงที่ (Constant)

ทำหน้าที่สำหรับเก็บพักข้อมูล ตามแต่ละชนิดที่ได้ประกาศไว้ แต่เป็นค่าที่ไม่สามารถเปลี่ยนแปลงได้อีก จุดประสงค์เพื่อที่จะนำชื่อค่าคงที่ไปใช้ในโปรแกรม ในภาษาซี ควรกำหนดชื่อค่าคงที่ด้วยตัวพิมพ์ใหญ่ เพื่อให้แตกต่างจากตัวแปร การกำหนดค่าคงที่ ในภาษาซีสามารถทำได้ 2 รูปแบบ คือ

- **รูปแบบที่ 1 เป็นการกำหนด Macro**

```
#define  cont_name  value
```

เช่น

```
#define  PI  3.1415926
#define OWNER "Nuntchayathorn"
```

- **รูปแบบที่ 2 เป็นการประกาศค่าคงที่**

```
const type cont_name = value ;
```

เช่น

```
const float PI = 3.1415926 ;
const char OWNER[ ] = "Nuntchayathorn" ;
```

ข้อมูลพื้นฐาน (Primitive Data Type)

Literals

คือคำที่ใช้แสดงค่าคงที่ ในภาษาซีสามารถแบ่งเป็น 4 ประเภท ดังนี้

- Integer literal
- Floating-point literal
- Character literal
- String literal

Integer literal

ค่าคงที่จำนวนเต็ม

- เลขฐานสิบ (Decimal) เป็นเลขฐานสิบที่ใช้ทั่วไป
- เลขฐานแปด (Octal) สามารถนำไปคำนวณได้ โดยการพิมพ์ตัวเลขขึ้นต้นด้วยเลข 0 เช่น 071, 032, 0144
- เลขฐานสิบหก (Hexadecimal) สามารถนำไปคำนวณได้ โดยการพิมพ์ขึ้นต้นด้วย 0x หรือ 0X เช่น 0xff, 0x54, 0XA9

เลขจำนวนเต็ม

- ในโปรแกรม Turbo C/C++ V3.0 มี 2 ขนาดคือมีขนาด 16 บิต กับ 32 บิต แบ่งเป็น 4 ชนิด คือ

  - int มีขนาด 16 บิต

  เป็นแบบคิดเครื่องหมาย มีค่าอยู่ระหว่าง -32768 ถึง +32767

  - unsigned int มีขนาด 16 บิต

  เป็นแบบไม่คิดเครื่องหมาย มีค่าอยู่ระหว่าง 0 ถึง 65535

  - long มีขนาด 32 บิต

  เป็นแบบคิดเครื่องหมาย มีค่าอยู่ระหว่าง -2147483648 ถึง +2147483647

  - unsigned long มีขนาด 32 บิต

  เป็นแบบไม่คิดเครื่องหมาย มีค่าอยู่ระหว่าง 0 ถึง 4294967295

Floating-point literal

คือค่าคงที่เลขทศนิยม การพิมพ์หรือแสดงตัวเลขทศนิยม มีอยู่ 2 รูปแบบ คือ

- แบบมาตรฐาน เช่น 0.123456
- แบบวิทยาศาสตร์ เช่น 1.2345e-1 หรือ 1.2345E-1

เลขจำนวนจริง

ในภาษาซี มีเลขทศนิยมอยู่ 3 ชนิด คือ

- แบบ float

  มีขนาด 32 บิต มีค่าอยู่ระหว่าง 3.4 x 10-38 ถึง 3.4 x 10+38

  แสดงจุดทศนิยม 6 หลัก

- แบบ double

  มีขนาด 64 บิต มีค่าอยู่ระหว่าง 1.7 x 10-308 ถึง 1.7 x 10+308

  แสดงจุดทศนิยม 15 หลัก

- แบบ long double

  มีขนาด 80 บิต มีค่าอยู่ระหว่าง 3.4 x 10-4932 ถึง 1.1 x 10+4932

  แสดงจุดทศนิยม 19 หลัก

Character literal

คือค่าคงที่ตัวอักษร ที่มีความยาว 1 ตัวอักขระ โดยต้องเขียนอยู่ในเครื่องหมาย ' ' (Single Quote) ซึ่งแบ่งออกเป็นตัวอักษรที่มองเห็นได้ กับตัวอักษรที่ใช้ในการควบคุมการพิมพ์

- ตัวอักษรปกติ

ได้แก่ ตัวอักษรที่ปรากฏอยู่แป้นพิมพ์ a-z, A-Z, 0-9, และสัญลักษณ์ต่าง ๆ บนแป้นพิมพ์

- ตัวอักษรที่ใช้ในการควบคุมการพิมพ์

ใช้แทนการแสดงผลอักขระหรือแป้นพิมพ์บางตัว เช่น แทนคีย์ Enter, Tab, Backspace เป็นต้น จะต้องแสดงโดยใช้ Escape Sequences ซึ่งได้แก่ \\, \', \", \r, \n, \f, \t, \b, \0

String literal

หมายถึงข้อมูลที่มีตัวอักขระตั้งแต่ 1 ตัวขึ้นไป ต้องเขียนอยู่ใน เครื่องหมาย " " (Double Quote)

ค่าคงที่ String เช่น "Hello World", "Samutsongkhram", "คอมพิวเตอร์", "1234567890" ทุกสตริงจะปิดด้วยรหัส \0 เสมอ

ในการจองพื้นที่ในการจัดเก็บสตริงต้องจองเก็บรหัสปิดนี้เพิ่มอีกหนึ่งตัวอักษร ดังนั้นในการประกาศตัวแปรสำหรับกำหนดขนาดความยาวของข้อความต้องเผื่ออีกหนึ่งตำแหน่งสำหรับจัดเก็บข้อมูลปิดสตริงนี้เสมอ ในภาษาซี สามารถกำหนดความยาวของข้อความได้ไม่เกิน 254 ตัว

ตัวแปร (Variable)

ตัวแปรทำหน้าที่สำหรับเก็บพักข้อมูล ตามแต่ละชนิดที่ได้ประกาศไว้ สามารถเปลี่ยนแปลงค่าได้ตลอดเวลา ดังนั้นตัวแปรจึงหมายถึง ชื่อที่ใช้ในการอ้างอิงพื้นที่ในหน่วยความจำหลักที่จองไว้เพื่อเก็บข้อมูล การจองหน่วยความจำทำได้โดยการประกาศตัวแปร ซึ่งสามารถอ้างอิงถึงข้อมูลได้โดยไม่ต้องรู้ตำแหน่งที่เก็บจริงของข้อมูล ซึ่งการจองหน่วยความจำสำหรับตัวแปรแต่ละตัวจำเป็นจะต้องมีการประกาศตัวแปรก่อน

ประเภทของตัวแปร

- **ตัวแปรประเภทพื้นฐาน (Scalar)**

  เป็นตัวแปรที่ใช้ในการแทนค่าข้อมูลได้เพียงค่าเดียว

- **ตัวแปรประเภทตัวแปรชุดหรืออาร์เรย์ (Array)**

  เป็นตัวแปรที่สามารถเก็บข้อมูลไว้ได้หลายค่าโดยใช้ชื่อตัวแปรเดียวกัน โดยมีตัวชี้ (Index) เป็นตัวระบุตำแหน่งที่เก็บค่าข้อมูล

ช่วงของข้อมูล (Range)

Type	Range
char	-128 ถึง +127
int, short	-32,768 to +32,767
long	-2,147,483,648 to +2,147,483,647
unsigned char	0-255
unsigned int	0 to 65,535
unsigned long	0 to 4,294,967,295
float	-3.4e-38 to +3.4e+38
double	-1,7e-308 to +1.7e+308
long double	-3.4e-4932 to +1.1e+4932

หลักการตั้งชื่อตัวแปร

- ประกอบด้วยอักษรภาษาอังกฤษ a ถึง z และตัวเลข 0 ถึง 9 แต่ตัวแรกต้องเป็นตัวอักษร A-Z หรือ a-z หรือขีดเส้นใต้ _ (Underscore) เท่านั้น
- ห้ามเว้นวรรค สามารถใช้เครื่องหมาย _ และ $ ในการตั้งชื่อได้
- ห้ามใช้คำสงวน (Reserved Word) ในการตั้งชื่อ
- ในขอบเขตเดียวกัน ห้ามตั้งชื่อซ้ำกัน
- การพิมพ์ชื่อโดยใช้อักษรตัวใหญ่กับตัวเล็กถือว่าเป็นคนละตัวแปรกัน
- ควรตั้งชื่อให้สื่อความหมาย หรือเป็นคำนาม
- นิยมตั้งชื่อตัวแปรเป็นตัวเล็ก ถ้าเป็นค่าคงที่จะใช้ตัวพิมพ์ใหญ่ทั้งหมด

คำสงวน (Reserved Word)

auto	extern	sizeof
break	float	static
case	for	struct
char	goto	switch
const	if	typedef
continue	int	union
default	long	unsigned
do	register	void
double	return	volatile
else	short	while
enum	signed	

การเปลี่ยนประเภทของตัวแปร

วิธีการเปลี่ยนประเภทของตัวแปรในภาษาซี ทำได้ 2 วิธี คือ

1.การเปลี่ยนประเภทของตัวแปรโดยอัตโนมัติ (Implicit type conversion)

2.โดยวิธีที่เรียกว่าการ Casting

ตัวดำเนินการ (Operator)

หมายถึง ตัวที่ทำหน้าที่เชื่อมโยงค่าหรือกระทำกับค่าต่าง ๆ กับข้อมูลและตัวแปรตามคำสั่ง เพื่อให้ได้ผลลัพธ์ตามต้องการ แบ่งเป็น 6 ประเภท

- ตัวดำเนินการทางคณิตศาสตร์
- ตัวดำเนินการทางตรรกะ
- ตัวดำเนินการเปรียบเทียบ
- ตัวดำเนินการเพิ่มค่าและลดค่า
- ตัวดำเนินการระดับบิต
- ตัวดำเนินการกำหนดค่า

ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic or Mathematic Operator)

- + บวก,
- - ลบ,
- * คูณ,
- / หาร และ
- % หารเอาเศษ (Remainder) or Modulo

ตัวดำเนินการทางตรรกะ (Logical Operator)

&&	And	และ
||	Or	หรือ
!	Not	ไม่, ตรงกันข้าม

ตัวดำเนินการเปรียบเทียบ (Relational Operator)

<	น้อยกว่า	Less than
<=	น้อยกว่าหรือเท่ากับ	Less than or Equal
>	มากกว่า	Greater than
>=	มากกว่าหรือเท่ากับ	Greater than or Equal
==	เท่ากับ, เท่ากัน	Equal
!=	ไม่เท่ากับ, ไม่เท่ากัน	Not equal

ตัวดำเนินการเพิ่มค่าและลดค่า (Increment and Decrement Operator)

++ เพิ่มค่า Increment

-- ลดค่า Decrement

ตัวดำเนินการระดับบิต (Bit-wise Operator)

&	And
|	Or
^	Exclusive Or
~	Complement
>>	Right Shift
<<	Left Shift

ตัวดำเนินการกำหนดค่า (Assignment Operator)

+=	&=
-=	|=
*=	^=
/=	<<=
%=	>>=

ลำดับการทำงานของตัวดำเนินการ

1. ()
2. !, ++, --
3. *, /, %
4. +, -
5. <, <=, >, >=
6. ==, !=
7. &&
8. ||
9. *=, /=, %=, +=, -=

ทิศทางการทำงาน จากซ้ายไปขวา

นิพจน์ (Expression)

หมายถึง การนำค่าคงที่ ตัวแปร และตัวดำเนินการมาประกอบกัน เพื่อเขียนเป็นคำสั่งให้ตัวแปลภาษาคอมพิวเตอร์เข้าใจ และสามารถทำงานตามคำสั่งได้ผลลัพธ์ตามต้องการ

สูตร	นิพจน์
• $\sum_{k=1}^{n} k = 1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$	• k1 = n * (n + 1) / 2;
• $f1 = (a+b)^2 = a^2 + 2ab + b^2$	• f1 = a * a + 2 * a * b + b * b;
• $c = \sqrt{a^2 + b^2}$	• c = sqrt(a * a + b * b);
• $K = \frac{1}{4}a^2\sqrt{3}$	• K=0.25 * a * a * sqrt(3);

Basic Function in C Mode Text

- clrscr0; - ล้างหน้าจอในโหมดเท็กซ์
- clreol0; - ลบข้อความจากตำแหน่งเคอร์เซอร์ไปจนสุดบรรทัด
- delline0; - ลบบรรทัด
- gotoxy0; - ย้ายตำแหน่งเคอร์เซอร์ไปยังตำแหน่งที่ระบุ
- wherex0; - คืนค่าตำแหน่งคอลัมน์ที่เคอร์เซอร์ปรากฏ (1-80)
- wherey0; - คืนค่าตำแหน่งบรรทัดที่เคอร์เซอร์ปรากฏ (1-25)
- textcolor0; - กำหนดสีตัวอักษร (0-15)
- textbackground0; - กำหนดสีพื้น (0-7)
- textmode0; - เข้าสู่โหมดเท็กซ์ (C80)

Basic Function in C Mode Text (ต่อ)

- cprintf0; - แสดงผลข้อความหรือค่าตัวแปรตามสีที่กำหนด
- printf0; - แสดงผลข้อความหรือค่าตัวแปร
- sprintf0; - แปลงข้อมูลตัวเลขให้เป็นข้อความ
- puts0; - แสดงผลข้อความ
- putch0; - แสดงผลอักขระ
- getch0; - รับค่าอักขระจากแป้นพิมพ์โดยจะไม่แสดงอักขระที่ป้อน
- getche0; - รับค่าอักขระจากแป้นพิมพ์โดยจะแสดงอักขระที่ป้อน
- gets0; - รับค่าข้อความจากแป้นพิมพ์มาเก็บไว้ในตัวแปรสตริงที่ระบุ
- scanf0; - รับค่าข้อมูลจากแป้นพิมพ์มาเก็บไว้ในตัวแปรที่ระบุ
- window0; - กำหนดพื้นที่หน้าต่าง

Basic Function Graphics Mode

- initgraph0; - เข้าสู่โหมดกราฟิก
- closegraph0; - ออกจากโหมดกราฟิก
- cleardevice0; - ล้างหน้าจอในโหมดกราฟิก
- setcolor0; - กำหนดสี
- setfillstyle0; - กำหนดรูปแบบการระบาย
- settextstyle0; - กำหนดรูปแบบข้อความ
- arc0; - วาดเส้นโค้ง
- bar0; - วาดแท่งสี่เหลี่ยม
- bar3d0; - วาดแท่งสี่เหลี่ยมสามมิติ
- circle0; - วาดวงกลม

Basic Function Graphics Mode (ต่อ)

- ellipse0; - วาดวงรี
- pieslice0; - วาดส่วนซีกวงกลม
- line0; - วาดเส้นตรงจากจุดหนึ่งไปยังจุดหนึ่งตามที่ระบุพิกัด
- lineto0; - วาดเส้นตรงจากจุดเดิมไปยังอีกจุดที่ระบุพิกัด
- linerel0; - วาดเส้นตรงจากจุดเดิมไปยังจุดที่อยู่ห่างตามระยะห่างที่กำหนด
- moveto0; - ย้ายพิกัดไปที่ใหม่ตามที่ระบุ
- moverel0; - ย้ายพิกัดไปที่ใหม่ห่างจากพิกัดเดิมตามระยะห่างที่กำหนด
- floodfill0; - ระบายสีที่ตำแหน่งพิกัดที่กำหนดไปจนจุดสิ้นสุดตามสีที่ระบุ
- outtext0; - แสดงข้อความที่กำหนดในพิกัดปัจจุบัน
- outtextxy0; - แสดงข้อความที่กำหนดตามพิกัดที่กำหนด

ฟังก์ชันเกี่ยวกับการแสดงผลทางจอภาพ

การแสดงผลบนจอในโหมดข้อความ (Text Mode) โปรแกรม Turbo C มีขนาดจอมาตรฐาน 80 x 25 คือมี 80 คอลัมน์ และ 25 บรรทัด

textmode(C80);

เป็นฟังก์ชันเข้าสู่โหมดเท็กซ์ (Text mode) หน้าต่างมาตรฐาน 80 x 25, 16 สี

clrscr();

เป็นฟังก์ชันใน TurboC ทำหน้าที่ล้างหน้าจอ (Clear screen) พร้อมนำเคอร์เซอร์ไปยังตำแหน่งซ้ายขนบรรทัดแรกของจอ (คือตำแหน่ง 1,1) ซึ่งฟังก์ชันนี้มีใช้ใน Turbo C เท่านั้น หากใช้กับ C ของค่ายอื่น ให้ใช้คำสั่ง system("cls"); แทนได้

clreol();

เป็นฟังก์ชัน ลบข้อความตั้งแต่ตำแหน่งที่เคอร์เซอร์ปรากฏอยู่ไปจนสุดบรรทัดนั้น ถ้าเคอร์เซอร์อยู่ที่คอลัมน์แรก คำสั่งนี้ก็จะเป็นการลบข้อความในบรรทัดนั้นทั้งบรรทัด

printf()

เป็นฟังก์ชันมาตรฐานในภาษาซีที่ใช้ในการแสดงผลลัพธ์ออกทางจอภาพ ณ ตำแหน่งที่เคอร์เซอร์ปรากฏอยู่

รูปแบบ

printf("ข้อความ");

printf("control_format",exp1 [,exp2[,...expN]]); Control_format อาจประกอบไปด้วย ข้อความ หรือ อักขระควบคุมการแสดงผลลัพธ์ หรือ รหัสรูปแบบ อยู่ภายในเครื่องหมายคำพูด " " แต่รวมความยาวทั้งหมดไม่เกิน 254 อักขระ

exp1, exp2, ... expN คือเป็นค่าข้อมูล หรือตัวแปร หรือนิพจน์ก็ได้

จำนวนของ expression จะต้องสัมพันธ์กับชนิดและจำนวนรหัสรูปแบบที่อยู่ในเครื่องหมายคำพูด เช่น

```
printf("Hello");
printf("\n");
printf("Welcome to Thailand\n");
printf("\nValue  A\t\tValue  B\t\tSum\n");
printf("%3d\t\t%3d\t\t%3d\n",a,b,(a+b));
printf("Score = %3d\t\tGrade = %.2f\n",78,3.5);
printf("Score = %3d\t\tGrade = %.2f\n",score,grade);
```

อักขระควบคุมการแสดงผลลัพธ์ (Escape Control)

- \a Bell
- \b Backspace
- \f Form feed
- \n New line
- \r Carriage return
- \t Horizontal tab
- \\ back slash
- \' Single quote
- \" Double quote
- \0 Null

รหัสรูปแบบข้อมูล (Format code)

หมายถึง รหัสรูปแบบข้อมูล เพื่อกำหนดประเภทข้อมูลที่กำหนด โดยต้องระบุภายในเครื่องหมาย " " (Double Quote)

- %c ใช้กับข้อมูลประเภท char
- %d ใช้กับข้อมูลประเภท int - เลขฐานสิบ
- %u ใช้กับข้อมูลประเภท unsigned int
- %l ใช้กับข้อมูลประเภท long
- %lu ใช้กับข้อมูลประเภท unsigned long
- %e ใช้กับข้อมูลประเภท float ในรูป e ยกกำลัง
- %f ใช้กับข้อมูลประเภท float
- %lf ใช้กับข้อมูลประเภท double
- %g ใช้กับข้อมูลประเภท float
- %h ใช้กับข้อมูลประเภท short int
- %o, %O ใช้กับข้อมูลประเภท - เลขฐานแปด
- %0x,%0X ใช้กับข้อมูลประเภท - เลขฐานสิบหก
- %s ใช้กับข้อมูลประเภท ข้อความ (String)
- %p ใช้กับตัวแปรพอยน์เตอร์ (Pointer)

ฟังก์ชันเกี่ยวกับการแสดงผลทางจอภาพ (ต่อ)

- putchar()

เป็นฟังก์ชันมาตรฐานในภาษาซีที่ใช้ในการแสดงอักขระออกทางจอภาพ

รูปแบบ putchar(ch);

โดย ch เป็นตัวแปรชนิดอักขระ

- puts()

เป็นฟังก์ชันมาตรฐานในภาษาซีที่ใช้ในการแสดงข้อความออกทางจอภาพ

รูปแบบ puts(str);

โดย str เป็นตัวแปรชนิดข้อความ หรือ สายอักขระ

- gotoxy( )

เป็นฟังก์ชันย้ายเคอร์เซอร์ไปตำแหน่ง x และ y

รูปแบบ gotoxy(x,y);

x มีค่าอยู่ระหว่าง 1-80

y มีค่าอยู่ระหว่าง 1-25

window()

เป็นฟังก์ชันกำหนดขนาดของหน้าต่างในโหมดเท็กซ์ ซึ่งมีค่าโดยปริยาย (Default) = 80 x 25 (80 คอลัมน์ 25 บรรทัด)

รูปแบบ

window(left, top, right, bottom);

left ขอบหน้าต่างด้านซ้าย มีค่าอยู่ระหว่าง 1-80

right ขอบหน้าต่างด้านขวา มีค่าอยู่ระหว่าง 1-80 แต่ต้องมากกว่า left

top ขอบหน้าต่างด้านบน มีค่าอยู่ระหว่าง 1-25 แต่ต้องน้อยกว่า bottom

bottom ขอบหน้าต่างด้านล่าง มีค่าอยู่ระหว่าง 1-25 แต่ต้องมากกว่า top

การกำหนดสีตัวอักษรและสีพื้น และคำสั่งแสดงผล

- textcolor(fc);

  fc = ค่าสีตัวอักษร

  สามารถกำหนดได้ 16 สี มีค่าอยู่ระหว่าง 0-15

- textbackground(bc);

  bc = ค่าสีพื้น

  สามารถกำหนดได้ 8 สี มีค่าอยู่ระหว่าง 0-7

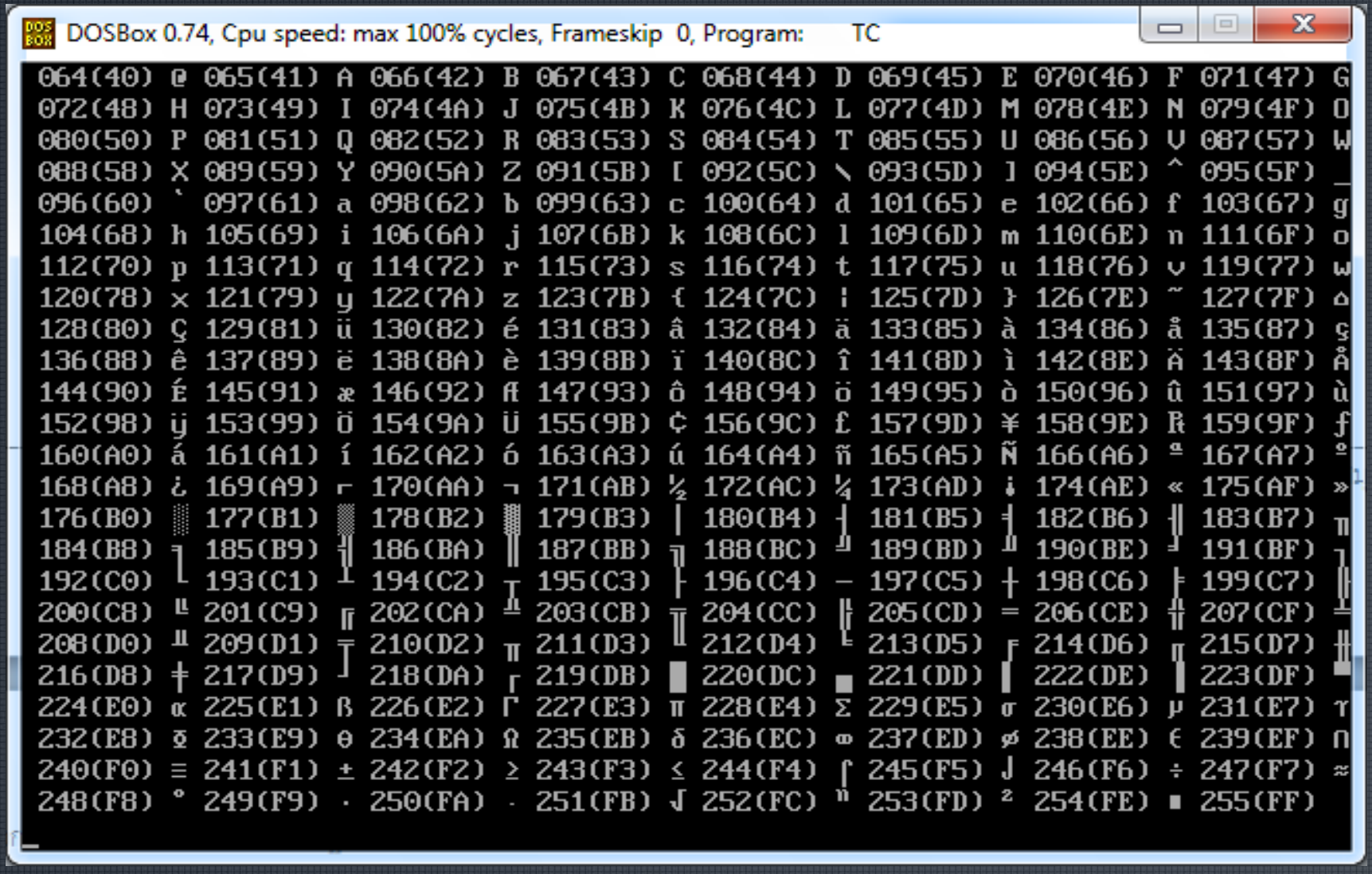
- cprintf()

  เป็นฟังก์ชันสำหรับพิมพ์ข้อความที่เป็นสี ใช้กับ Turbo C เท่านั้น

ตัวอย่าง โปรแกรมแสดงรหัสแอสกี้ (ASCII Code)

```
// by Nuntchayathorn Chatrsuwun
#include <conio.h>
#include <stdio.h>
void main()
{
        clrscr();
        for (int i=32; i<=255; i++)
        {
                if (i == 128) getch();
                printf(" %03d(%02X) %c",i,i,i);
        }
        getch();
```

ผลการรันโปรแกรม



ฟังก์ชันรับค่าข้อมูลทางแป้นพิมพ์

- scanf()

เป็นฟังก์ชันในการรับข้อมูลจากแป้นพิมพ์ (Keyboard) เข้าไปเก็บไว้ในตัวแปรที่กำหนด โดยชนิดข้อมูลสามารถกำหนดได้ ทุกประเภท ดังนั้นฟังก์ชันนี้จึงเป็นฟังก์ชันที่นิยมใช้กันมากในการรับข้อมูล เมื่อคำสั่งนี้ทำงาน จะปรากฏเคอร์เซอร์ (Cursor) กระพริบรอรับการป้อนข้อมูล ให้เราพิมพ์ข้อมูลที่ต้องการแล้วกดคีย์ Enter

รูปแบบ scanf("format", &var);

"format" : รหัสรูปแบบข้อมูล (Format code)

var : ตัวแปรที่จะทำหน้าที่เก็บค่าที่รับเข้ามาจากแป้นพิมพ์

โดยชื่อตัวแปรจะต้องนำด้วยเครื่องหมาย & เสมอ

ยกเว้นถ้าเป็นตัวแปรชนิดข้อความ อาจจะไม่ต้องใส่เครื่องหมาย & ก็ได้

การรับค่าทางแป้นพิมพ์ให้กับตัวแปรเดียว

scanf("%d", &n);

เป็นการรับข้อมูลตัวเลขจำนวนเต็มฐานสิบจากแป้นพิมพ์มาเก็บไว้ที่ตัวแปร n

scanf("%s", str); หรือ

scanf("%s",&str);

เป็นการรับข้อมูลที่เป็นข้อความ มาเก็บไว้ที่ตัวแปร str

scanf("%c", &ch);

เป็นการรับข้อมูลอักขระจากแป้นพิมพ์มาเก็บไว้ในตัวแปร ch

การรับข้อมูลชนิดอักขระ

- แบบต้องกดคีย์ Enter ทุกครั้ง

getchar( )

เป็นฟังก์ชันที่ใช้สำหรับรับข้อมูลประเภทตัวอักขระจากแป้นพิมพ์ โดยรับข้อมูลครั้งละ 1 ตัวอักขระเท่านั้น ดังนั้นตัวแปรที่มารับค่า ก็จะต้องประกาศเป็นชนิด char ด้วย เมื่อป้อนอักขระ 1 ตัวแล้ว ต้องกดคีย์ Enter

รูปแบบ

```
var = getchar();
```

เช่น

```
char ch;
printf("Enter your charactor : ");
ch = getchar();
```

การรับข้อมูลชนิดอักขระ

- แบบไม่ต้องกดคีย์ Enter ทุกครั้ง

getch();

เป็นฟังก์ชันสำหรับรับข้อมูลชนิดอักขระจากแป้นพิมพ์ 1 อักขระ โดยไม่ต้องกดคีย์ Enter จะไม่แสดงค่าอักขระ

getche();

เป็นฟังก์ชันสำหรับรับข้อมูลชนิดอักขระจากแป้นพิมพ์ 1 อักขระ โดยไม่ต้องกดคีย์ Enter จะแสดงอักขระที่กด

การรับข้อมูลชนิดข้อความ

- gets()

เป็นฟังก์ชันที่ใช้สำหรับรับข้อมูลชนิดข้อความจากแป้นพิมพ์ และเมื่อป้อนข้อมูลที่เป็นข้อความเสร็จแล้ว ให้กดคีย์ Enter

รูปแบบ gets(str);

str คือตัวแปรชนิดข้อความไว้สำหรับเก็บข้อมูลชนิดข้อความที่ป้อนจากแป้นพิมพ์

เช่น

```
char str[30];
printf("Enter your name : ");
fflush(stdin); gets(str);
printf("Your name is %s\n",str);
```

จบการนำเสนอ

บทที่ 6 โครงสร้างและไวยากรณ์ภาษาชี