บทที่ 4

ประเภทของข้อมูลและตัวดำเนินการ

Data Types and Operator

สื่อสไลด์นี้ใช้รูปแบบฟอนต์ สารบรรณ รุ่นปรับปรุงใหม่ “Sarabun New”
สามารถดาวน์โหลดฟอนต์ได้ที่ URL : https://www.f0nt.com/release/th-sarabun-new/

1. ประเภทของข้อมูล

ข้อมูลในภาษาซี แบ่งได้เป็น 4 กลุ่ม

- ข้อมูลชนิดอย่างง่าย (Simple Type)
- ข้อมูลประเภทแถวอักขระ (String Type)
- ข้อมูลชนิดโครงสร้าง (Structure Type)
- ข้อมูลชนิดศรชี้ (Pointer Type)

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลตัวเลขจำนวนเต็ม
  - ข้อมูลอักขระ
  - ข้อมูลตรรกะ
- ข้อมูลประเภทจำนวนจริง (Real Type)
  - ข้อมูลตัวเลขทศนิยม

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลจำนวนเต็ม สามารถแสดงผลในระบบเลขฐานได้ 4 รูปแบบ

ชนิด	ตัวอย่าง
integer	..., -3, -2, -1, 0, 1, 2, 3, ...
binary	0, 1
octal	0124, 076, 04
hexadecimal	0x17, 0xd ,0x5f

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลตัวเลขจำนวนเต็ม

ตัวอย่างโค้ดโปรแกรมเพื่อแสดงการใช้เลขฐานรูปแบบต่าง ๆ

main.c saved

```c
#include <stdio.h>
int x= 0b1011;
int y= 0x10;
int z= 10;
int main(void) {
  printf("x in Decimal     : %d\n",x);
  printf("x in Octal       : %o\n",x);
  printf("x in Hexadecimal : %x\n\n",x);

  printf("y in Decimal     : %d\n",y);
  printf("y in Octal       : %o\n",y);
  printf("y in Hexadecimal : %x\n\n",y);

  printf("z in Decimal     : %d\n",z);
  printf("z in Octal       : %o\n",z);
  printf("z in Hexadecimal : %x\n",z);
  return 0;
}
```

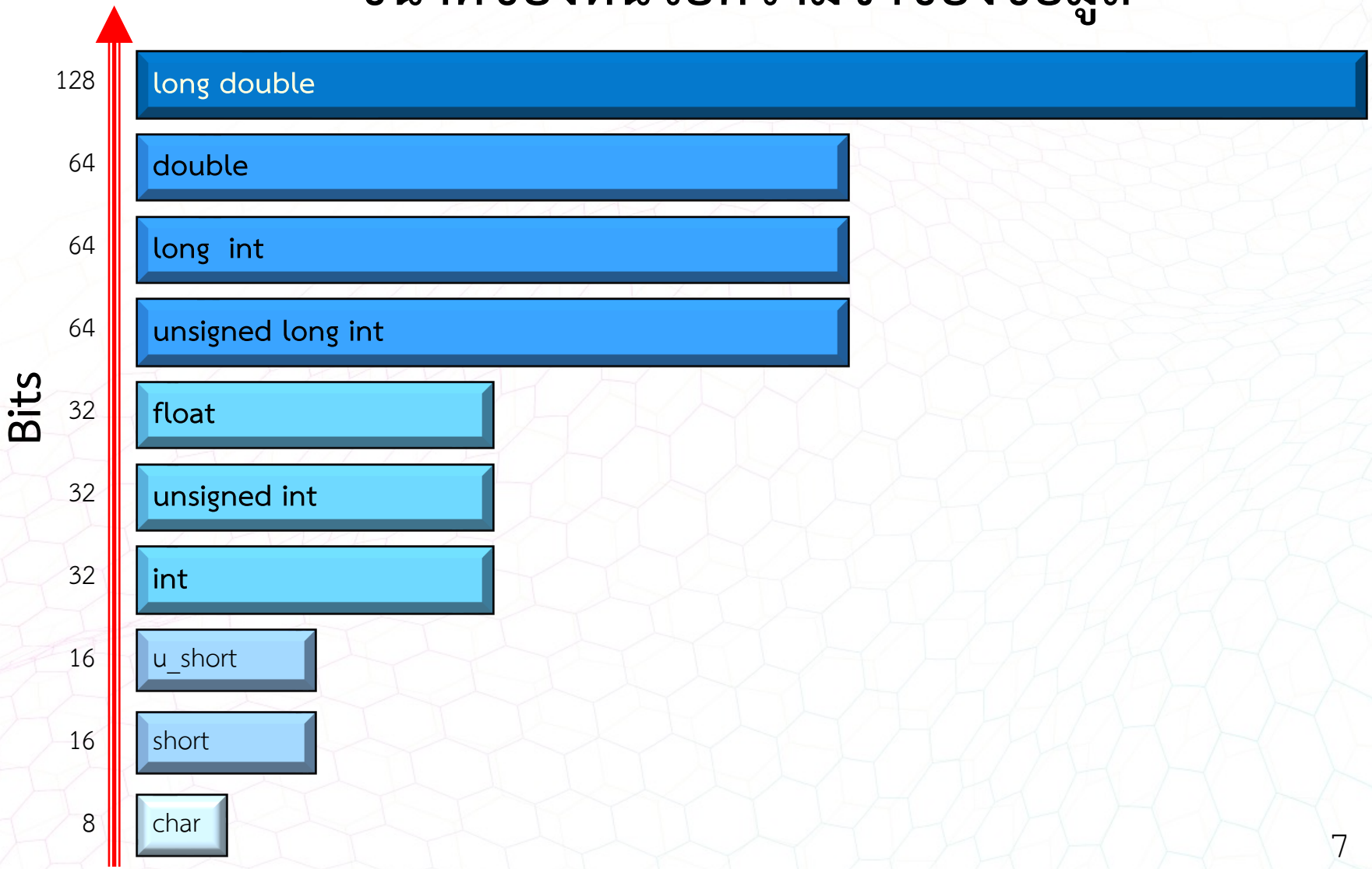
```
clang-7 -pthread -lm -o main main.c
./main
x in Decimal     : 11
x in Octal       : 13
x in Hexadecimal : b

y in Decimal     : 16
y in Octal       : 20
y in Hexadecimal : 10

z in Decimal     : 10
z in Octal       : 12
z in Hexadecimal : a
```

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลจำนวนเต็ม มีช่วงขอบเขตที่ใช้งานได้ ดังนี้

ชนิดข้อมูล	ช่วงของข้อมูลที่เก็บไว้
(signed) char	-128...+127
unsighted char	0...255
(signed) short int	-32768...32767
unsigned short int	0...65535
(signed) int	- 2,147,483,648 ...+ 2,147,483,647
unsigned int	0...4,294,967,295
(signed) long int	-9,223,372,036,854,775,808 ...+ 9,223,372,036,854,775,807
unsigned long int	0...18,446,744,073,709,551,615

ชนิดข้อมูลและ
ขนาดของหน่วยความจำของข้อมูล



ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลตัวเลขจำนวนเต็ม

ตัวอย่างโค้ดโปรแกรมเพื่อแสดงขอบเขตข้อมูลแต่ละชนิด

main.c saved

```c
#include <stdio.h>
#include <limits.h>
short int  x;
int main(void) {
  printf("The minimum value of (SIGNED) CHAR        = %d\n", CHAR_MIN);
  printf("The maximum value of (SIGNED) CHAR        = %d\n", CHAR_MAX);
  printf("The maximum value of UNSIGNED CHAR        = %d\n", 0);
  printf("The maximum value of UNSIGNED CHAR        = %d\n\n", UCHAR_MAX);

  printf("The minimum value of (SIGNED) SHORT INT   = %d\n", SHRT_MIN);
  printf("The maximum value of (SIGNED) SHORT INT   = %d\n", SHRT_MAX);
  printf("The minimum value of UNSIGNED SHORT INT   = %d\n", 0);
  printf("The maximum value of UNSIGNED SHORT INT   = %d\n\n", USHRT_MAX);

  printf("The minimum value of (SIGNED) INT         = %d\n", INT_MIN);
  printf("The maximum value of (SIGNED) INT         = %d\n", INT_MAX);
  printf("The minimum value of UNSIGNED INT         = %d\n", 0);
  printf("The maximum value of UNSIGNED INT         = %u\n\n", UINT_MAX);

  printf("The minimum value of (SIGNED) LONG        = %ld\n", LONG_MIN);
  printf("The maximum value of (SIGNED) LONG        = %ld\n", LONG_MAX);
  printf("The minimum value of UNSIGNED LONG        = %d\n", 0);
  printf("The maximum value of UNSIGNED LONG        = %lu\n", ULONG_MAX);
  return 0;
}
```

```
clang-7 -pthread -lm -o main main.c
clang-7 -pthread -lm -o main main.c
clang-7 -pthread -lm -o main main.c
clang-7 -pthread -lm -o main main.c
./main
The minimum value of (SIGNED) CHAR        = -128
The maximum value of (SIGNED) CHAR        = 127
The maximum value of UNSIGNED CHAR        = 0
The maximum value of UNSIGNED CHAR        = 255

The minimum value of (SIGNED) SHORT INT   = -32768
The maximum value of (SIGNED) SHORT INT   = 32767
The minimum value of UNSIGNED SHORT INT   = 0
The maximum value of UNSIGNED SHORT INT   = 65535

The minimum value of (SIGNED) INT         = -2147483648
The maximum value of (SIGNED) INT         = 2147483647
The minimum value of UNSIGNED INT         = 0
The maximum value of UNSIGNED INT         = 4294967295

The minimum value of (SIGNED) LONG        = -9223372036854775808
The maximum value of (SIGNED) LONG        = 9223372036854775807
The minimum value of UNSIGNED LONG        = 0
The maximum value of UNSIGNED LONG        = 18446744073709551615
```

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลอักขระ (Character Data Type)

สามารถแสดงค่าได้ 2 รูปแบบ ดังนี้

1. ตัวเลข

ชนิดข้อมูล	ช่วงของข้อมูลที่เก็บไว้
char	-128...+127 หรือ 0...255

2. ตัวอักขระหนึ่งตัว ซึ่งเป็นไปตามตารางรหัส ASCII ซึ่งประกอบไปด้วย ตัวอักษร ตัวเลข อักขระพิเศษ

ชนิดข้อมูล	ตัวอย่าง
char	'C' , 'a' , '\n' , '#' , '@', '{' , '0' , '$'

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลอักขระ (Character Data Type)

ตัวอย่างโค้ดโปรแกรมเพื่อแสดงข้อมูลอักขระ ทั้งสองรูปแบบ

main.c saved

```c
#include <stdio.h>
int x= 65;
int y= 33;
int main(void) {
  printf("x in Decimal     : %d\n",x);
  printf("x in Octal       : %o\n",x);
  printf("x in Hexadecimal : %x\n",x);
  printf("x in character   : %c\n\n",x);

  printf("y in Decimal     : %d\n",y);
  printf("y in Octal       : %o\n",y);
  printf("y in Hexadecimal : %x\n",y);
  printf("y in character   : %c\n",y);
  return 0;
}
```

```
clang-7 -pthread -lm -o main main.c
./main
x in Decimal     : 65
x in Octal       : 101
x in Hexadecimal : 41
x in character   : A

y in Decimal     : 33
y in Octal       : 41
y in Hexadecimal : 21
y in character   : !
```

ข้อมูลชนิดอย่างง่าย (Simple Type)

- ข้อมูลประเภทลำดับ (Ordinal Type)
  - ข้อมูลตรรกะ (Boolean Data Type)

    จะเป็นค่าทางลอจิก
    - ค่าเท็จ (False) แทนค่าด้วยเลข 0
    - ค่าจริง (True) แทนค่าด้วยเลข 1

หมายเหตุ ค่าจริง คือ ค่าที่ไม่เท่ากับ 0

เช่น -1 4 25 356

ข้อมูลชนิดอย่างง่าย (Simple Type)

- **ข้อมูลประเภทจำนวนจริง (Real Data Type)**

  ไม่เป็นข้อมูลชนิดลำดับ เนื่องจากทศนิยมมีได้หลายตำแหน่ง

ชนิดข้อมูล	ช่วงของข้อมูลที่เก็บไว้
float	±1.17549e-38...3.40282e+38
double	±1.79769e-308...2.22507e+308
long double	±3.3621e-4932... 1.18973e+4932

ข้อมูลชนิดอย่างง่าย (Simple Type)

- **ข้อมูลประเภทจำนวนจริง (Real Data Type)**

ตัวอย่างโค้ดโปรแกรมเพื่อแสดงขอบเขตข้อมูลแต่ละชนิด

main.c saved

```c
#include <stdio.h>
#include <float.h>
int main(void) {
    printf("Storage size for float : %lu \n", sizeof(float));
    printf("-Minimum float             : %g\n", (float) -FLT_MIN);
    printf("-Maximum float             : %g\n", (float) -FLT_MAX);
    printf("+Minimum float             : %g\n", (float) FLT_MIN);
    printf("+Maximum float             : %g\n\n", (float) FLT_MAX);

    printf("Storage size for double float : %lu \n", sizeof(double));
    printf("-Minimum double float      : %g\n", (double) -DBL_MIN);
    printf("-Maximum double float      : %g\n", (double) -DBL_MAX);
    printf("+Minimum double float      : %g\n", (double) DBL_MIN);
    printf("+Maximum double float      : %g\n\n", (double) DBL_MAX);

    printf("Storage size for long double float : %lu \n", sizeof(long double));
    printf("-Minimum long double float : %Lg\n", -LDBL_MIN);
    printf("-Maximum long double float : %Lg\n", -LDBL_MAX);
    printf("+Minimum long double float : %Lg\n", LDBL_MIN);
    printf("+Maximum long double float : %Lg\n", LDBL_MAX);
    return 0;
}
```

```
clang-7 -pthread -lm -o main main.c
./main
Storage size for float : 4
-Minimum float             : -1.17549e-38
-Maximum float             : -3.40282e+38
+Minimum float             : 1.17549e-38
+Maximum float             : 3.40282e+38

Storage size for double float : 8
-Minimum double float      : -2.22507e-308
-Maximum double float      : -1.79769e+308
+Minimum double float      : 2.22507e-308
+Maximum double float      : 1.79769e+308

Storage size for long double float : 16
-Minimum long double float : -3.3621e-4932
-Maximum long double float : -1.18973e+4932
+Minimum long double float : 3.3621e-4932
+Maximum long double float : 1.18973e+4932
```

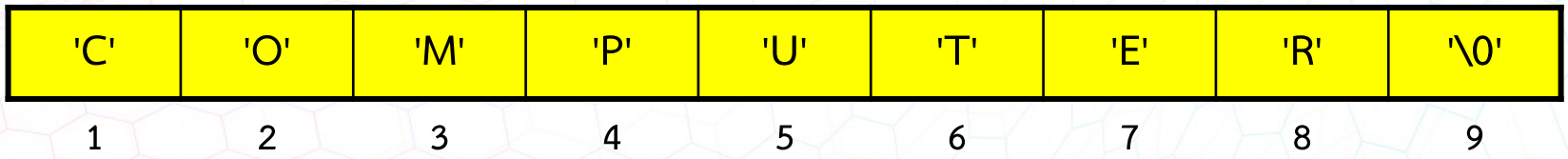
ข้อมูลประเภทแถวอักขระ (String Type)

- เป็นการนำตัวอักขระ (char) มาเรียงต่อกันเป็นข้อความตั้งแต่หนึ่งตัวเป็นต้นไป
- สามารถเก็บตัวอักขระได้ 255 ตัว
- โดยตัวอักขระทั้งหมด จะต้องอยู่ในเครื่องหมาย “ ” (Double Quote)
  - ภาษาซี มีการเติมตัวอักษรว่าง Null character ('\0') เป็นตัวสุดท้ายของสตริง

Example

```
char dept[9] = "COMPUTER";
```

dept



ใช้เนื้อที่ในการเก็บทั้งสิ้น 9 Bytes

การประกาศตัวแปรและค่าคงที่

ตัวแปร (Variables)

- หมายถึง ค่าในหน่วยความจำที่สามารถเปลี่ยนค่าได้
- ชื่อตัวแปรจะเป็นตำแหน่งหน่วยความจำที่เก็บข้อมูลอยู่
- การประกาศตัวแปรสามารถทำได้ดังนี้

```
Type   Variables_list
```

ตัวอย่างเช่น

```
int    count;
```

```
float  data1 , data2=10;
```

การประกาศตัวแปรและค่าคงที่

ค่าคงที่ (Constant)

- หมายถึง ค่าในหน่วยความจำที่มีค่าคงที่ตลอดโปรแกรม
- การประกาศค่าคงที่
  - ประกาศ ให้ค่าคงที่ ชื่อว่า b เป็นชนิด Integer เก็บค่า 12 ไว้

```
const int b = 12 ;
const float pi = 3.14159 ;
const char ch = 'A' ;
```

ตัวอย่างการประกาศตัวแปรและค่าคงที่

```
1   #include <stdio.h>
2   const int taxrate = 7;
3   float itemcost , salestax;
4   int main()
5   {
6       printf("Please  Enter  Cost of  item : ");
7       scanf("%f" , &itemcost);
8       salestax = (taxrate * itemcost) / 100;
9       printf(" item Cost is = %.2f \n" , itemcost);
10      printf(" Sales tax is %.2f\n" , salestax);
11      return(0);
12  }
```

ผลการทำงาน?

```
Output
Please  Enter  Cost of item : 500
item Cost is = 500.00
Sales tax is 35.00
```

2. ตัวดำเนินการ (Operators)

ภาษาซีมีตัวดำเนินการ ดังต่อไปนี้

2.1) ตัวดำเนินการเลขคณิต (Arithmetic Operators)

2.2) ตัวดำเนินการเปรียบเทียบ (Relational Operators)

2.3) ตัวดำเนินการลอจิก (Logical Operators)

2.4) ตัวดำเนินการการกำหนดค่า (Assignment Operators)

ตัวดำเนินการเลขคณิต (Arithmetic Operators)

operator	ความหมาย	ชนิดของข้อมูล	ตัวอย่าง
+	บวก	ตามชนิดของข้อมูล	5 + 2
-	ลบ	ตามชนิดของข้อมูล	5 - 2
*	คูณ	ตามชนิดของข้อมูล	5 * 2
/	หาร (Divide)	ตามชนิดของข้อมูล	5 / 2
%	หารเอาเฉพาะเศษของการหาร (Modulo)	จำนวนเต็ม	5%2
++	การเพิ่มค่าขึ้นหนึ่ง	จำนวนเต็ม	x++ , ++x
--	การลดค่าลงหนึ่ง	จำนวนเต็ม	x-- , --x

ตัวอย่างการใช้ตัวดำเนินการเลขคณิต

ค่าตัวแปร x	ค่าตัวแปร y	การดำเนินการ	ค่าจากการกระทำ	ผลลัพธ์
11	5	x = y + 2	7	x = 7
11	5	x = x / y	2	x = 2
11.0	5	x = x / y	2.2	x = 2.2
9	2	x = x % y	1	x = 1
14	-3	x = x % y	2	x = 2
-14	3	x = x % y	-2	x = -2

ลำดับการดำเนินงานของ Operators

ลำดับ	ชนิด	ตัวดำเนินการ	ลำดับการทำงานในกลุ่มเดียวกัน
1		( )	ทำในวงเล็บก่อน
2	ตัวดำเนินการเอกภาค	! (Not) , ++ , -- , เครื่องหมายติดลบ	ขวาไปซ้าย
3	ตัวดำเนินการเลขคณิต	* , / , %	ซ้ายไปขวา
4		+ , -	ซ้ายไปขวา
5	ตัวดำเนินการเปรียบเทียบ	< , > , <= , >=	ซ้ายไปขวา
6		== , !=	ซ้ายไปขวา
7	ตัวดำเนินการลอจิก	&& (And)	ซ้ายไปขวา
8		|| (Or)	ซ้ายไปขวา
9	ตัวดำเนินการการกำหนดค่า	= , *= , /= , %= , += , -=	ขวาไปซ้าย

⚠ ในกรณีอยู่ในลำดับเดียวกันจะดำเนินการตามลำดับการทำงานในกลุ่มเดียวกัน

ตัวอย่างการใช้ตัวดำเนินการเลขคณิต

1) 13 / 5 * 3

ลำดับ	การดำเนินการ	ผลลัพธ์
1	13 / 5 * 3	2
2	2 * 3	6

ตัวอย่างการใช้ตัวดำเนินการเลขคณิต

2) 6 * 5 / 10 * 2 + 10

ลำดับ	การดำเนินการ	ผลลัพธ์
1	6 * 5 / 10 * 2 + 10	30
2	30 / 10 * 2 + 10	3
3	3 * 2 + 10	6
4	6 + 10	16

ตัวอย่างการใช้ตัวดำเนินการเลขคณิต

	ค่าตัวแปร x	การดำเนินการ	ค่าจากการกระทำ
การกำหนดค่าคงที่	7	x = 7 ;	7
การกำหนดค่าให้ตัวแปรด้วยนิพนธ์	7	x = x+1;	8
	7	x = x-1;	6
การใช้ตัวดำเนินการเพิ่มหรือลดค่าตัวแปร	7	x++;	8
	7	++x;	8
	7	x--;	6
	7	--x;	6
	7	Y = ++x;	Y มีค่าเป็น 8 , x = 8
	7	Y = x++;	Y มีค่าเป็น 7 , x = 8

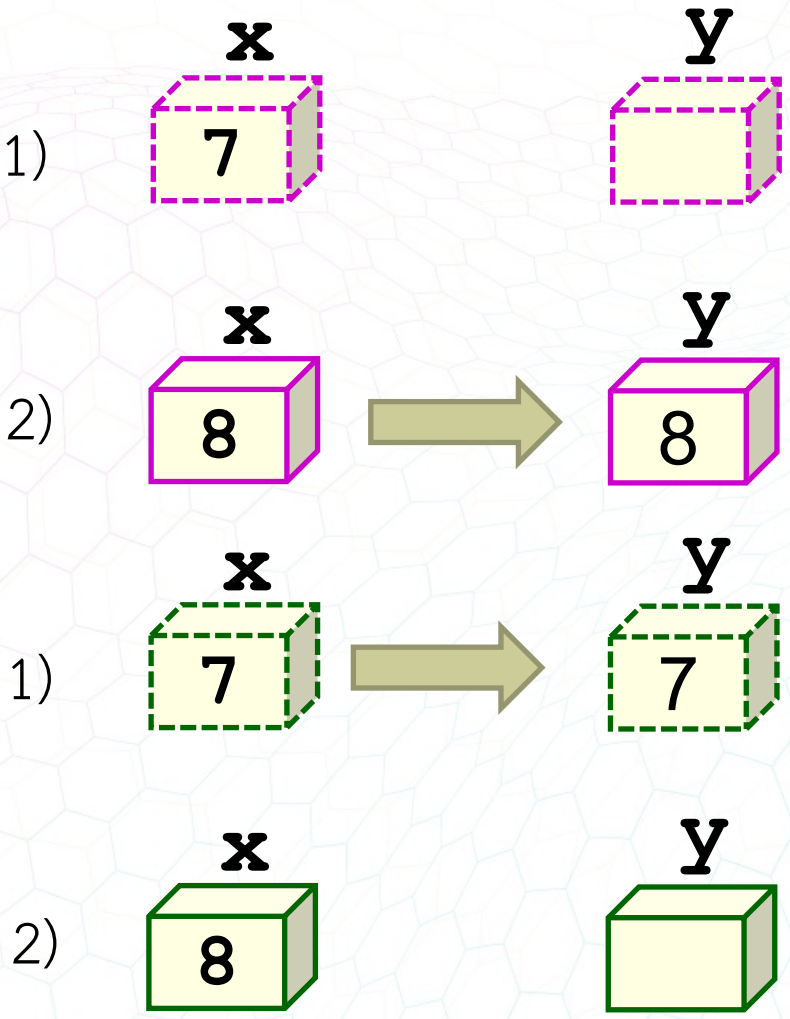
ตัวดำเนินการเพิ่มและตัวดำเนินการลดค่าตัวแปร (Increments and Decrement Operators)

ถ้า x มีค่าเป็น 7

y= + + x ; อธิบายได้ว่า

x มีค่าเป็น 7 ต่อมาเพิ่มค่า x ขึ้นหนึ่ง

แล้วส่งให้ตัวแปร y ทำให้ y มีค่าเป็น 8



ถ้า x มีค่าเป็น 7

y= x + +; อธิบายได้ว่า

x มีค่าเป็น 7 ทำให้ y มีค่าเป็น 7 ด้วย

และเพิ่มค่า x ขึ้นหนึ่ง ส่งผลให้ x มีค่าเป็น 8

ตัวดำเนินการเพิ่มและตัวดำเนินการลดค่าตัวแปร (Increments and Decrement Operators)

สรุปได้ว่า

ถ้า x มีค่าเป็น 7 y= + + x ;

- ถ้าวางตัวดำเนินการไว้หน้าตัวแปร จะทำการเพิ่มค่าตัวแปร x ก่อนแล้วจึงส่งค่าให้กับ y

ผลลัพธ์สุดท้าย ตัวแปร x มีค่าเท่ากับ y

ถ้า x มีค่าเป็น 7 y= x + +;

- ถ้าวางตัวดำเนินการไว้หลังตัวแปร จะทำการส่งค่าปัจจุบันให้กับ y ก่อนแล้วจึงเพิ่มค่า x

ผลลัพธ์สุดท้าย ตัวแปร x มีค่าไม่เท่ากับ y

ตัวดำเนินการการกำหนดค่า (Assignment Operators) และ การกำหนดตัวแปรแบบย่อ (shorthand assignment)

operator	ความหมาย	ชนิดของข้อมูล	ตัวอย่าง
=	Assignment	ตามชนิดของข้อมูล	A= B
+=	Add and Assignment	ตามชนิดของข้อมูล	A+=2
-=	Subtract and Assignment	ตามชนิดของข้อมูล	A-=5
*=	Multiply and Assignment	ตามชนิดของข้อมูล	A*=6
/=	Divide and Assignment	ตามชนิดของข้อมูล	A/=3
%=	Modulus and Assignment	จำนวนเต็ม	A%=2

ตัวดำเนินการกำหนดค่า (Assignment Operators) และ การกำหนดตัวแปรแบบย่อ (shorthand assignment)

Assignment Operator	ความหมายเทียบเคียง
x += y	x = x + y
x -= y	x = x - y
x *= y	x = x * y
x /= y	x = x / y
x %= y	x = x % y

ตัวดำเนินการเปรียบเทียบ (Relational Operators)

operator	ความหมาย	ชนิดของข้อมูล	ตัวอย่าง	ผลลัพธ์
<	น้อยกว่า	boolean	5 < 4	0
<=	น้อยกว่าหรือเท่ากับ	boolean	5 <= 5	1
==	เท่ากับ	boolean	0 == 0	1
>	มากกว่า	boolean	5 > 4	1
>=	มากกว่าหรือเท่ากับ	boolean	5 >= 4	1
!=	ไม่เท่ากับ	boolean	0 != 0	0

⚠ ไม่ควรใช้ ข้อมูลประเภททศนิยม float double

ตัวอย่างการใช้ตัวดำเนินการเปรียบเทียบ

```
1   #include <stdio.h>
2   int main()
3   {
4       int i, j;
5       scanf("%d %d" , &i , &j );
6       printf(" i <  j: %d\n" , i<j );
7       printf(" i <= j: %d\n" , i<=j );
8       printf(" i == j: %d\n" , i==j );
9       printf(" i >  j: %d\n" , i>j );
10      printf(" i >= j: %d\n" , i>=j );
11      return(0);
12  }
```

```
1
6
 i <  j: 1
 i <= j: 1
 i == j: 0
 i >  j: 0
 i >= j: 0
```

```
6  -1
 i <  j: 0
 i <= j: 0
 i == j: 0
 i >  j: 1
 i >= j: 1
```

ตัวดำเนินการลอจิก (Logical Operators)

operator	ความหมาย	ชนิดของข้อมูล	ตัวอย่าง	ผลลัพธ์
&&	AND	boolean	1 && -1	1
||	OR	boolean	1 || 0	1
!	NOT	boolean	!4	0

ตารางค่าความจริง AND

A	B	A && B
1	1	1
1	0	0
0	1	0
0	0	0

ตารางค่าความจริง OR

A	B	A || B
1	1	1
1	0	1
0	1	1
0	0	0

ตารางค่าความจริง NOT

A	!A
1	0
0	1

ตัวอย่างการใช้ตัวดำเนินการลอจิก

```
1   #include <stdio.h>
2   int main()
3   {
4   int A , B;
5   printf("  A    B    A AND B\n");
6   A = 1; B = 1;
7   printf("| %d | %d |     %d    |\n", A, B, A&&B);
8   A = 1; B = 0;
9   printf("| %d | %d |     %d    |\n", A, B, A&&B);
10  A = 0; B = 1;
11  printf("| %d | %d |     %d    |\n", A, B, A&&B);
12  A = 0; B = 0;
13  printf("| %d | %d |     %d    |\n", A, B, A&&B);
14  return(0);
15  }
```

ผลการทำงาน?

```
   A    B    A AND B
| 1  | 1  |     1     |
| 1  | 0  |     0     |
| 0  | 1  |     0     |
| 0  | 0  |     0     |
```

ตัวอย่างการใช้ตัวดำเนินการลอจิก

```
1   #include <stdio.h>
2   int main()
3   {
4   int  i , j;
5   scanf("%d %d" , &i , &j );
6   printf(" i && j : %d\n" , i && j );
7   printf(" i || j : %d\n" , i || j );
8   printf(" !i     : %d\n" , !i  );
9   printf(" !j     : %d\n" , !j ) ;
10  return(0);
11  }
```

ผลการทำงาน

```
2
-1
 i && j : 1
 i || j : 1
 !i     : 0
 !j     : 0
```

```
0  -2
 i && j : 0
 i || j : 1
 !i     : 1
 !j     : 0
```

ตัวอย่างลำดับการดำเนินงานของ Operators

```
1  #include <stdio.h>
2  int main()
3  {
4  int  i , j;
5  printf("2+3 && 1          : %d\n" , 2+3 && 1);
6  printf("2+3 && 0          : %d\n" , 2+3 && 0);
7  printf("-3 < 1 && 1       : %d\n" , -3 < 1 && 1);
8  printf("12/3*5 > 10 < 5: %d\n" , 12/3*5 > 10 < 5);
9  return(0);
10 }
```

ผลการทำงาน ?

```
2+3 && 1          : 1
2+3 && 0          : 0
-3 < 1 && 1       : 1
12/3*5 > 10 < 5: 1
```

ตัวอย่างลำดับการดำเนินงานของ Operators

การดำเนินการทางตรรกะ

2+3 && 1 = ?

ผลลัพธ์ 2 + 3 && 1 = 1

1

5 && 1

ค่าตรรกะ True

ค่าตรรกะ True

การดำเนินการทางตรรกะ

2+3 && 0 = ?

ผลลัพธ์ 2 + 3 && 0 = 0

1

5 && 0

ค่าตรรกะ True

ค่าตรรกะ False

ตัวอย่างลำดับการดำเนินงานของ Operators

การดำเนินการทางตรรกะ

-3 < 1 && 1 = ?

ผลลัพธ์ -3 < 1 && 1 = 1

1

1 && 1

ค่าตรรกะ True

ค่าตรรกะ True

การดำเนินการทางตรรกะ

12/3*5 > 10 < 5 = ?

ผลลัพธ์ 12/3*5 > 10 < 5 = 1

1 4 * 5 > 10 < 5

2 20 > 10 < 5

3 1 < 5

1

การประกาศชนิดข้อมูลของตัวแปร

x	y	นิพจน์	z
int	int	z = x + y	int
float	float	z = x + y	float
int	float	z = x + y	float
int	double	z = x * y	double
char	float	z = x + y	float
short	long	z = (x + y) / 4.6	long then float

ผลลัพธ์จะเก็บในข้อมูลประเภทที่ใหญ่กว่าเสมอ

การเปลี่ยนประเภทของข้อมูล (Casting Type)

ภาษาซีสามารถเปลี่ยนประเภทของข้อมูลให้เป็นไปตามที่ผู้ใช้ต้องการได้ โดยการนำชนิดข้อมูลไว้หน้าข้อมูล เช่น

x	นิพจน์	ผลลัพธ์
int	(float)(x)	float
float	(int)(x + 2.5)	int
int	(float)(x +1)	float

จบบทที่ 4

สื่อสไลด์นี้ใช้รูปแบบฟอนต์ สารบรรณ รุ่นปรับปรุงใหม่ “Sarabun New”
สามารถดาวน์โหลดฟอนต์ได้ที่ URL : https://www.f0nt.com/release/th-sarabun-new/