

344-111 ชุดวิชาการโปรแกรมและขั้นตอนวิธี

ตัวดำเนินการและนิพจน์



ตัวดำเนินการและนิพจน์

- ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic operators)
- ตัวดำเนินการยูนารี (Increment and Decrement Operators)
- ตัวดำเนินการเชิงเปรียบเทียบ (Relational operators)
- ตัวดำเนินการเชิงตรรกะ (Logical operators)
- ตัวดำเนินการกำหนดค่า (Assignment operators)
- ตัวดำเนินการเงื่อนไข (Conditional operators)
- นิพจน์ (expression)
- ขั้นตอนการทำงานของนิพจน์
- Examples



ตัวดำเนินการทางคณิตศาสตร์

ตัวดำเนินการ	ความหมาย	ตัวอย่าง
+	บวก	$A+B$
-	ลบ	$A-B$
*	คูณ	$A*B$
/	หาร	A/B
%	การหารเอาแต่เศษไว้ (modulus)	$A\%B$

ตัวอย่าง

นิพจน์คณิตศาสตร์	นิพจน์คณิตศาสตร์ในภาษาซี
------------------	--------------------------

$$\frac{a+b}{cd}$$

$$(a+b)/(c*d)$$

$$10x - 3xy + 10y^2$$

$$10*x - 3*x*y + 10*y*y$$



ตัวดำเนินการทางคณิตศาสตร์

- กำหนดตัวแปร A และ B เป็นจำนวนเต็ม

`int A = 10, B = 3;`

นิพจน์	ค่า
A+B	13
A-B	7
A*B	30
A/B	3
A%B	1



ตัวดำเนินการยูนารี

ตัวดำเนินการ	ความหมาย	ตัวอย่าง
--	การลดค่าลง ครั้งละ 1	A-- (เหมือนกับ A=A-1)
++	การเพิ่มค่าขึ้นครั้งละ 1	A++ (เหมือนกับ A=A+1)

ตัวอย่าง 1

```
int x;  
x = 10;  
printf("x = %d",x);  
x--;  
printf("x = %d",x);  
x++;  
printf("x = %d",x);
```

ผลลัพธ์ ??????



ตัวดำเนินการยูนารี

ตัวอย่าง 2

```
int i = 1;  
printf("i = %d\n",i);  
printf("i = %d\n",++i);  
printf("i = %d\n",i);
```

++ i คือ บวกก่อนแล้วจึงพิมพ์ค่า

ตัวอย่าง 3

```
int i = 1;  
printf("i = %d\n",i);  
printf("i = %d\n", i++);  
printf("i = %d\n",i);
```

i ++ คือ พิมพ์ค่าก่อนแล้วจึงบวก



ตัวดำเนินการเชิงเปรียบเทียบ

ตัวดำเนินการ	ความหมาย	ตัวอย่าง
>	มากกว่า	$A > B$ (A มากกว่า B)
<	น้อยกว่า	$A < B$ (A น้อยกว่า B)
>=	มากกว่า หรือ เท่ากับ	$A \geq B$ (A มากกว่าหรือเท่ากับ B)
<=	น้อยกว่า หรือ เท่ากับ	$A \leq B$ (A น้อยกว่าหรือเท่ากับ B)
==	เท่ากับ	$A == B$ (A เท่ากับ B)
!=	ไม่เท่ากับ	$A != B$ (A ไม่เท่ากับ B)

หมายเหตุ เราใช้ตัวดำเนินการทั้ง 6 ตัวนี้เพื่อสร้างเป็นนิพจน์ทางตรรกะซึ่งให้แทนเงื่อนไขเป็นจริงหรือเท็จ ผลลัพธ์ที่ได้จากนิพจน์เหล่านี้ จะมีประเภทข้อมูลเป็นจำนวนเต็ม โดยจะแทนจริง (true) ด้วย 1 และเท็จ (false) ด้วย 0



ตัวดำเนินการเชิงเปรียบเทียบ

■ `int i=1, j=2, k=3;`

นิพจน์	การแปลความหมาย	ค่า
<code>i < j</code>	จริง	1
<code>(i+j) >= k</code>	จริง	1
<code>(i+k) > (i+5)</code>	เท็จ	0
<code>k!=3</code>	เท็จ	0
<code>j==2</code>	จริง	1



ตัวดำเนินการเชิงตรรกะ

ตัวดำเนินการ

& &

||

!

ความหมาย

และ (and)

หรือ (or)

นิเสธ (not)

P	Q	P & Q	P Q
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

p	!p
1	0
0	1



ตัวดำเนินการเชิงตรรกะ

ตัวอย่าง `int i = 7;`
 `float f = 5.5;`
 `char c = 'w';`

นิพจน์	การแปลความหมาย	ค่า
<code>(i>=6) && (c=='w')</code>	จริง	1
<code>(i>=6) (c==119)</code>	จริง	1
<code>(f<11) && (i>100)</code>	เท็จ	0
<code>(c != 'p') ((i+f)<=10)</code>	จริง	1
<code>!(f<5)</code>	จริง	1

↑
ค่าเหล่านี้เราสามารถนำไปใช้คำนวณต่อได้



ตัวดำเนินการกำหนดค่า

■ คือ เครื่องหมาย =, +=, -=, *=, /=, %=

■ รูปแบบ 1

identifier = expression

โดยที่ identifier โดยทั่วไปจะเป็นตัวแปร

expression เป็นค่าคงที่ ตัวแปรหรือนิพจน์ที่มีความซับซ้อนได้

■ ตัวอย่าง

```
a = 4;  
x = y;  
alpha = 0.001;  
sum = a + b;  
area = length * width;
```

ข้อควรจำ !

= และ ==

มีความแตกต่างกันโดยสิ้นเชิง



ตัวดำเนินการกำหนดค่า

- ในกรณีที่ตัวดำเนินการทั้งสองข้างในนิพจน์มีการกำหนดค่าประเภทข้อมูลที่แตกต่างกัน ค่าของนิพจน์ที่อยู่ทางขวามือ จะถูกแปลงให้เป็นประเภทเดียวกันกับตัวแปรที่อยู่ทางซ้ายมืออัตโนมัติ ซึ่งจะมีผลกับค่าของข้อมูล เช่น เมื่อมีการกำหนดค่าให้กับตัวแปรที่เป็นจำนวนเต็ม จะมีการตัดส่วนที่เป็นทศนิยมของเลขทศนิยมทิ้ง
- ตัวอย่าง

int i;

<u>นิพจน์</u>	<u>ค่าของ i</u>
i = 3.3;	3
i = 3.9;	3
i = -3.9;	-3
i = 9/2;	4



ตัวดำเนินการกำหนดค่า

■ ตัวอย่าง

```
int i;  
int j = 5;
```

นิพจน์

$i = j;$

$i = j / 2;$

$i = 2 * j / 2;$

$i = 2 * (j / 2);$

ค่าของ i

5

2

5 (ทำงานจากซ้ายไปขวา)

4 (หารตัดเศษทิ้ง แล้วจึงคูณ)



ตัวดำเนินการกำหนดค่า

■ ตัวอย่าง

```
float i;  
float j=2;  
int k = 5;
```

!!! การประกาศชนิดของตัวแปรอย่างไม่ระมัดระวัง
มักจะเป็นต้นเหตุของข้อผิดพลาดสำหรับนักเขียน
โปรแกรมระดับเริ่มต้น

<u>นิพจน์</u>	<u>ค่าของ i</u>
i = k;	5.000000
i = 15 / j;	7.500000
i = k / j;	2.500000
i = k / 2;	2.000000
i = k / 2.0;	2.500000



ตัวดำเนินการกำหนดค่า

ตัวอย่าง

สมมติ `i` เป็นจำนวนเต็ม และใช้กับชุดอักขระแอสกี

```
int i;
```

<u>นิพจน์</u>	<u>ค่าของ i</u>
<code>i = 'x';</code>	120
<code>i = 'A';</code>	65
<code>i = ('x' - 'A')/8;</code>	6
<code>i = 'x' - 'A'/5;</code>	107

ลองดู !!!

```
char i, j, k;  
i = 'x';  
j = 'A';  
k = 'x' + 'A';  
ค่าของ k = ?
```



ตัวดำเนินการกำหนดค่า

■ รูปแบบ 2

`identifier1 = identifier2 = ... = expression`

- ซึ่งการกำหนดค่าลักษณะนี้จะทำงานจากขวาไปซ้าย
- ดังนั้นการกำหนด `identifier1 = identifier2 = expression` จึงมีค่าเท่ากับ `identifier1 = (identifier2 = expression)`

ตัวอย่าง

`i=j=5;`



ตัวดำเนินการกำหนดค่า

■ รูปแบบ 3

`expression1 += expression2`

■ การกำหนดค่าเช่นนี้มีค่าเท่ากับ `expression1 = expression2 + expression3`

■ ตัวอย่าง

`a = 4;`

`a += 5;`

หลังจากทำงานเสร็จจะได้ a มีค่าเท่ากับ 9;

หมายเหตุ โดยทั่วไป `expression1` จะเป็นตัวแปร



ตัวดำเนินการกำหนดค่า

- ในทำนองเดียวกัน
- รูปแบบ 4

`expression1 -= expression2`

- จะมีค่าเท่ากับ `expression1 = expression2 - expression3`

ตัวดำเนินการที่เหลืออีก 3 ตัว คือ `*=`, `/=` และ `%=`
ก็มีการทำงานในลักษณะเดียวกัน



ตัวดำเนินการกำหนดค่า

ตัวอย่าง

```
int i=5, j=7;
```

```
float f=5.5, g=-3.25;
```

นิพจน์	นิพจน์ที่เทียบเท่ากัน	ค่า
<code>i += 5</code>	<code>i = i + 5</code>	10
<code>f -= g</code>	<code>f = f - g</code>	8.75
<code>j *= (i-3)</code>	<code>j = j * (i-3)</code>	14
<code>f /= 3</code>	<code>f = f / 3</code>	1.833333
<code>i %= (j-2)</code>	<code>i = i % (j-2)</code>	0



ตัวดำเนินการเงื่อนไข

■ รูปแบบ

expression1 ? expression2 : expression3

■ ตัวอย่าง 1

$(i < 0) ? 0 : 100;$

หมายความว่า ถ้า i น้อยกว่า 0 นิพจน์นี้จะมีค่าเป็น 0

ถ้า i มีค่ามากกว่า หรือ เท่ากับ 0 นิพจน์นี้จะมีค่าเป็น 100

■ ตัวอย่าง 2

$\text{max} = (a \geq b) ? a : b;$

หมายความว่า ถ้า a มากกว่าหรือเท่ากับ b แล้ว max มีค่าเท่ากับ a

ถ้า a น้อยกว่า b แล้ว max มีค่าเท่ากับ b



กฎในการเขียนนิพจน์

- ห้ามเขียนตัวแปร 2 ตัวติดกันโดยไม่มีเครื่องหมาย เช่น
 - `ab` ในภาษาซีต้องเขียนเป็น `a*b`
 - การเขียน `ab` ถือว่าเป็นตัวแปรเดียว คือชื่อตัวแปร `ab` ไม่ใช่ค่า `a` คูณ `b`
- ถ้าเขียนนิพจน์โดยมีค่าของตัวแปรหรือค่าคงที่ต่างชนิดกันในนิพจน์เดียวกันภาษาซีจะเปลี่ยนชนิดของข้อมูลที่มีขนาดเล็กให้เป็นชนิดของข้อมูลที่ใหญ่ขึ้น เช่น
 - ถ้าใช้ `char` กับ `int` จะเปลี่ยนเป็น `int`
 - ถ้าใช้ `int` กับ `float` จะเปลี่ยนเป็น `float`
 - ถ้าใช้ `float` กับ `double` จะเปลี่ยนเป็น `double`

ตัวอย่าง

```
int a = 15;
float b = 2;
sum = a+b;
mul = a*b;
```

คำถาม: เราควรประกาศตัวแปร `sum` และ `mul` เป็นตัวแปรชนิดใด



กฎในการเขียนนิพจน์

การหาร

$\frac{\text{Integer}}{\text{Integer}} = \text{Integer}$

$\frac{\text{Float}}{\text{Float}} = \text{Float}$

$\frac{\text{Integer}}{\text{Float}} = \text{Float}$

$\frac{\text{Float}}{\text{Integer}} = \text{Float}$

ตัวอย่าง

int a = 5;

int b = 2;

float c = 2;

ถ้า a/b จะได้ผลลัพธ์เป็น 2

ถ้า a/c จะได้ผลลัพธ์เป็น 2.500000



ขั้นตอนการทำงานของนิพจน์

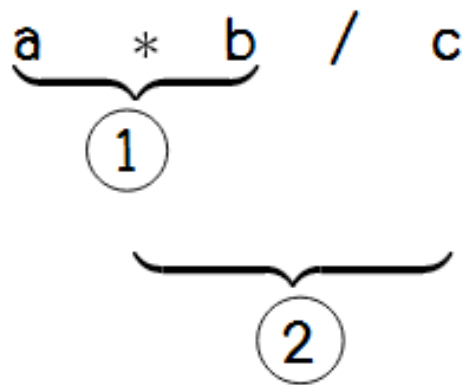
- นิพจน์ในภาษาซี จะทำงานตามลำดับการทำงานของตัวดำเนินการต่อไปนี้
- โดยเรียงจากการทำงานอันดับที่ 1 ลงไปถึงอันดับสุดท้าย คือ

(1)	()	ซ้ายไปขวา
(2)	!, ++, --	ขวาไปซ้าย
(3)	*, /, %	ซ้ายไปขวา
(4)	+, -	ซ้ายไปขวา
(5)	<, <=, >, >=	ซ้ายไปขวา
(6)	==, !=	ซ้ายไปขวา
(7)	&&	ซ้ายไปขวา
(8)		ซ้ายไปขวา
(9)	? :	ขวาไปซ้าย
(10)	=, +=, -=, *=, /=, %=	ขวาไปซ้าย

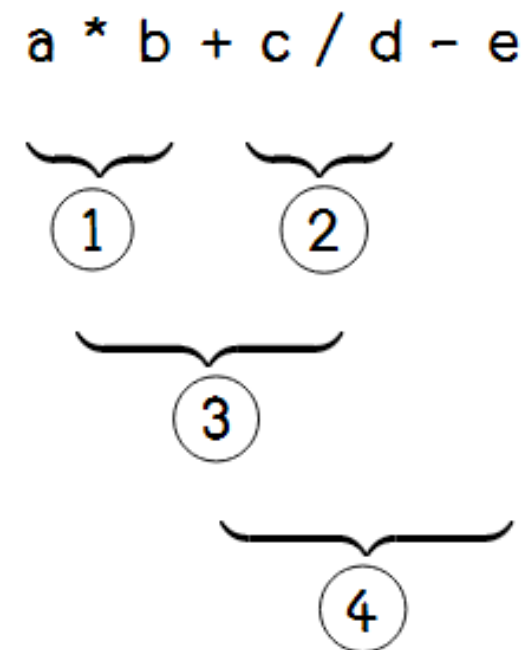


ขั้นตอนการทำงานของนิพจน์

ตัวอย่าง 1



ตัวอย่าง 2

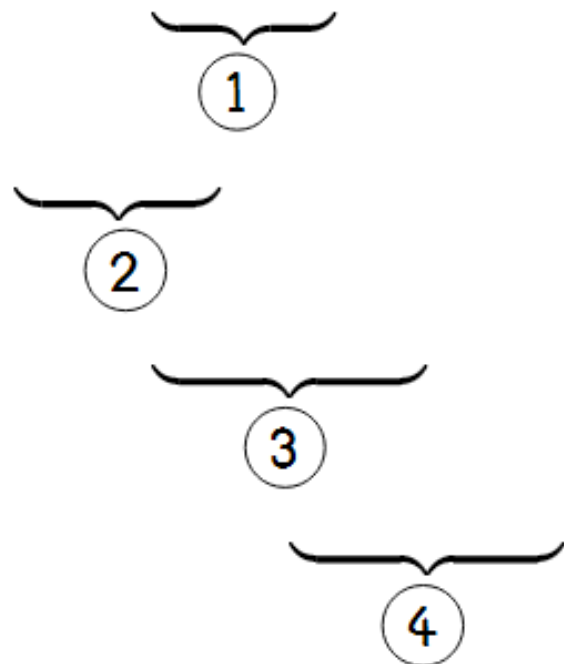




ขั้นตอนการทำงานของนิพจน์

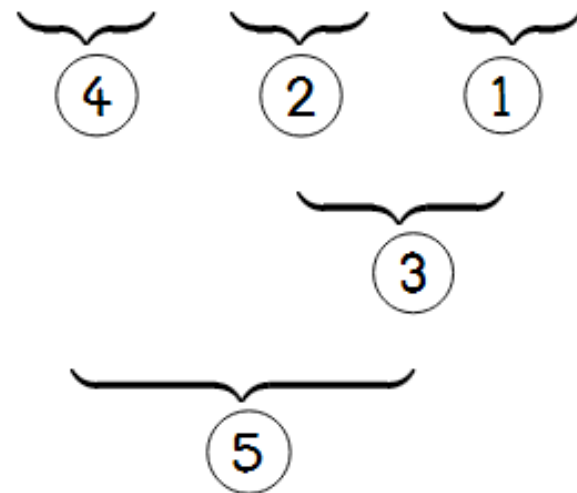
ตัวอย่าง 3

$$a * (b + c) / d - e$$



ตัวอย่าง 4

$$a - b + c * d / e --$$

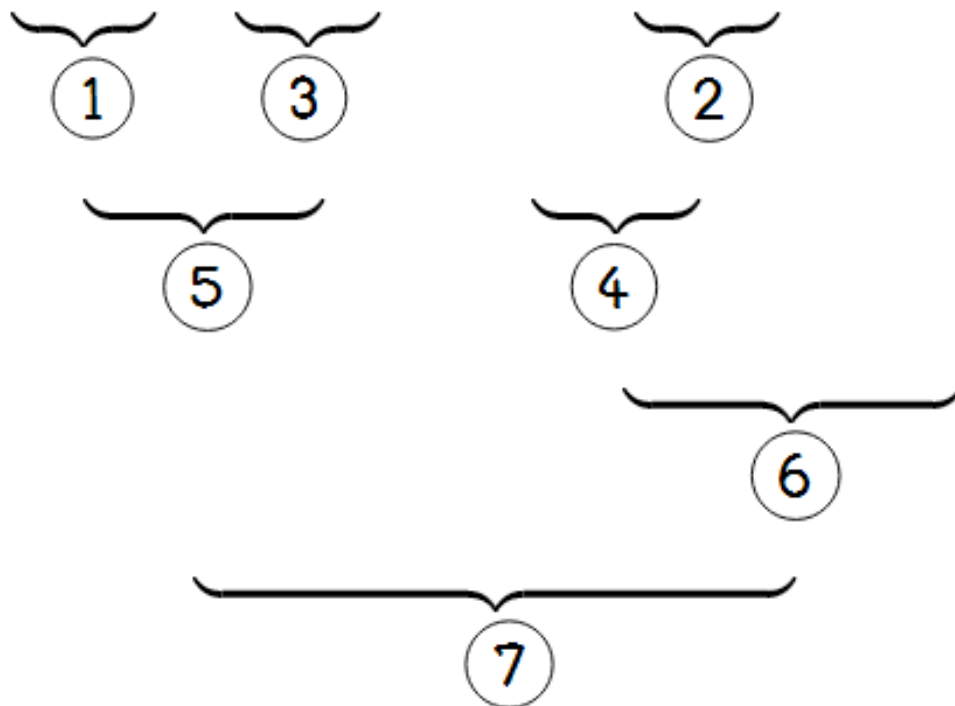




ขั้นตอนการทำงานของนิพจน์

ตัวอย่าง 5

$b * c > 5 - d \ \&\& \ e - f * g == 10$

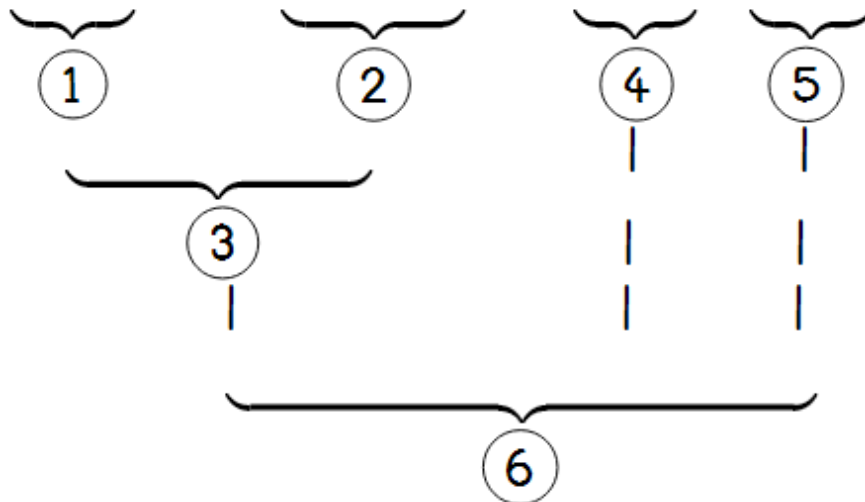




ขั้นตอนการทำงานของนิพจน์

ตัวอย่าง 6

`c += (a > 0 && a <= 10) ? ++ a : a / b`



ถ้าสมมติให้

`a = 1, b = 2, c = 3`

หลังจบการทำงานของนิพจน์นี้แล้วจะได้

`c = ???`

ถ้าสมมติให้

`a = 50, b = 10, c = 20`

หลังจบการทำงานของนิพจน์นี้แล้วจะได้

`c = ???`



Exercise 1

- What is the output

```
#include<stdio.h>
int main()
{
    int i = 5;
    int a = ++i + ++i;
    printf("%d",a);
    return 0;
}
```



Exercise 2

- What is a output

```
#include<stdio.h>
int main(){
    int i = 16;
    i =! i > 15;
    printf("i = %d",i);
    return 0;
}
```