344-111 ชุดวิชาการโปรแกรมและขั้นตอนวิธี

องค์ประกอบของภาษาซี

Outline

- ชุดอักขระภาษาซี
- ประเภทข้อมูล (Data type)
  - จำนวนเต็ม (Integer)
  - ตัวอักขระ (Character)
  - จำนวนทศนิยม (Floating-point)
- ค่าคงที่
  - ค่าคงที่จำนวนเต็ม
  - ค่าคงที่เลขทศนิยม
  - ค่าคงที่อักขระ
  - ค่าคงที่สตริง
- ตัวแปร
  - การประกาศตัวแปร
  - การตั้งชื่อตัวแปร
  - การกำหนดค่าเริ่มต้นให้กับตัวแปร
- นิพจน์ภาษาซี
  - ประโยคนิพจน์ (Expression statements)
  - ประโยคเชิงซ้อน (Compound statements)
  - ประโยคควบคุม (Control statements)
- ประโยค (Statements)
- การแสดงผลและการรับข้อมูล

ชุดอักขระภาษาซี

- ตัวอักษร a-z และ A-Z
- ตัวเลข 0-9
- ตัวอักขระพิเศษ ได้แก่

+	-	*	/	=	%	&	#	!	?	^	
"	'		□	|	<	>	(	)	[	]
{	}	:	;	.	,	_	(blank)				

หมายเหตุ: อนุญาตให้ใช้ @ และ $ ในส่วนที่เป็นสตริง และคอมเมนต์

ประเภทข้อมูล (Data type)

- จำนวนเต็ม (Integer)
- ตัวอักขระ (Character)
- จำนวนทศนิยม (Floating-point)

จำนวนเต็ม (Integer)

ประเภทข้อมูล	หน่วยความจำที่ใช้(ไบต์)	ช่วงของค่า
char	1	-128 to 127
unsigned char	1	0 to 255
int	2	-32,768 to 32,767
short int	2	-32,768 to 32,767
unsigned int	2	0 to 65,535
long	4	-2,147,483,648 to 2,147,483,647
unsigned long	4	0 to 4,294,967,295

หมายเหตุ: จำนวนไบต์ของหน่วยความจำที่ใช้เก็บข้อมูลในแต่ละประเภทเปลี่ยนแปลงได้ ขึ้นอยู่กับตัวแปลภาษา นั่นหมายความว่า ช่วงค่าจะเปลี่ยนไปด้วย

ตัวอักขระ (Character)

- เนื่องจากคอมพิวเตอร์จัดเก็บตัวอักขระในรูปแบบของตัวเลขที่เป็นรหัส ASCII ซึ่งในภาษาซีจะจัดเก็บตัวอักขระในรูปแบบของข้อมูลจำนวนเต็มประเภท “char”

จำนวนทศนิยม (Floating-point)

ประเภทข้อมูล	หน่วยความจำที่ใช้ (ไบต์)	ความถูกต้อง (ตำแหน่ง)	ช่วงของค่า
float	4	7	3.4×10^{-38} to 3.4×10^{38}
double	8	15	1.7×10^{-308} to 1.7×10^{308}
long double	10	19	3.4×10^{-4932} to 3.4×10^{4932}

ค่าคงที่

- ค่าคงที่จำนวนเต็ม
- ค่าคงที่เลขทศนิยม
- ค่าคงที่อักขระ
- ค่าคงที่สตริง

กฎสำหรับค่าคงที่จำนวนเต็ม และค่าคงที่เลขทศนิยม

- จะต้องไม่มีเครื่องหมายคอมมา (,) หรือช่องว่าง
- สามารถใช้เครื่องหมายลบ (-) นำหน้าได้
- ไม่สามารถกำหนดค่าของค่าคงที่ให้เกินค่าสูงสุดและต่ำสุดของช่วงได้

ค่าคงที่จำนวนเต็มฐานสิบ

- ในแต่ละหลักประกอบด้วยตัวเลข 0 ถึง 9 ซึ่งถ้ามีมากกว่าหนึ่งหลักแล้วหลักแรกต้องไม่เป็น 0
- พิจารณาค่าคงที่จำนวนเต็มฐานสิบต่อไปนี้ว่าถูกต้องหรือไม่
  - 0
  - 1
  - 743
  - 2580
  - 12,245
  - 36.0
  - 10 20
  - 123-45
  - 0900

ค่าคงที่จำนวนเต็มฐานแปด

- ในแต่ละหลักประกอบด้วยตัวเลข 0 ถึง 7 และตัวเลขหลักแรกต้องเป็น 0 เท่านั้น
- พิจารณาตัวอย่างต่อไปนี้
  - 0
  - 01
  - 0743
  - 07777
  - 345
  - 05280
  - 07777.77

ค่าคงที่จำนวนเต็มฐานสิบหก

- ต้องขึ้นต้นด้วย 0x หรือ 0X ตามด้วยตัวเลข 0 ถึง 9 และตัวอักษร a ถึง f (หรือ A ถึง F) ซึ่งใช้แทนเลขฐานสิบจาก 10 ถึง 15 ตามลำดับ
- พิจารณาตัวอย่างต่อไปนี้
  - 0x
  - 0X1
  - oX7FFF
  - 0xabcd
  - 0X12.34
  - 0BE80
  - 0XDEFG

ค่าคงที่จำนวนเต็มแบบ unsigned และ long

- กำหนดค่าคงที่จำนวนเต็มแบบ unsigned ให้ใส่ u หรือ U ไว้ที่ท้ายสุด
- กำหนดค่าคงที่จำนวนเต็มแบบ long ให้ใส่ l หรือ L ไว้ที่ท้ายสุด
- กำหนดค่าคงที่จำนวนเต็มแบบ unsigned long ให้ส่ ul หรือ UL ไว้ที่ท้ายสุด

- ตัวอย่าง

```
50000U        => ฐานสิบ (unsigned)
123456789L    => ฐานสิบ (long)
123456789LU   => ฐานสิบ (unsigned long)
0123456L      => ฐานแปด (long)
0777777U      => ฐานแปด (unsigned)
0XFFFFFUL     => ฐานสิบหก (unsigned long)
```

ค่าคงที่ทศนิยม

- เป็นเลขฐานสิบซึ่งประกอบด้วยจุดทศนิยม หรือเลขชี้กำลัง (หรือทั้งสองอย่าง)

ตัวอย่างที่ถูกต้อง

0.	1.	0.2	827.602
50000.	0.00743	12.3	315.0066
2E-8	0.006e-3	1.67E+8	.1212e12

ตัวอย่างที่ผิด

1	ต้องมีจุดทศนิยมหรือเลขชี้กำลัง
1,000.0	มีตัวอักขระที่ไม่ถูกต้อง (,)
2E+10.2	เลขชี้กำลังต้องเป็นเลขจำนวนเต็ม (มีจุดทศนิยมไม่ได้)
3E 10	มีตัวอักขระที่ไม่ถูกต้อง (ช่องว่าง)

ค่าคงที่เลขทศนิยม

การแปลความหมายของเลขทศนิยมที่มีเลขชี้กำลัง

จะเหมือนกับการแปลความหมายของสัญลักษณ์ทางวิทยาศาสตร์ เพียงแต่เลขฐาน 10 จะถูกแทนที่โดยตัวอักษร E (หรือ e)

ดังนั้น ตัวเลข 3×10^5 สามารถเขียนแทนโดยใช้ค่าคงที่ทศนิยมของภาษาชี้ได้หลายวิธี ดังนี้

300000.	3e5	3e+5	3E5	3.0e+5
.3e6	0.3E6	30E4	30.E+4	300e3

ค่าคงที่อักขระ

- ใช้เก็บอักษรเพียงหนึ่งตัว ซึ่งจะต้องอยู่ภายในเครื่องหมาย ‘’
- คอมพิวเตอร์โดยส่วนมากใช้แทนชุดรหัสแอสกี (ASCII) ซึ่งอักขระแต่ละตัวจะใช้ 7 บิต ซึ่งอักขระแต่ละตัวจะใช้ 7 บิต ดังนั้นจะอ้างถึงได้ 2^7 = 128 ตัว

ค่าคงที่	ค่า
'A'	65
'x'	120
'3'	51
'$'	36
' '	32

หมายเหตุ ค่าคงที่ตัวสุดท้าย คือ ช่องว่าง

ค่าคงที่อักขระ

ค่า ASCII	ตัวอักขระ	ค่า ASCII	ตัวอักขระ	ค่า ASCII	ตัวอักขระ	ค่า ASCII	ตัวอักขระ
000	NUL	032	blank	064	@	096	`
001	SOH	033	!	065	A	097	a
002	STX	034	"	066	B	098	b
003	ETX	035	#	067	C	099	c
004	EOT	036	$	068	D	100	d
005	ENQ	037	%	069	E	101	e
006	ACK	038	&	070	F	102	f
007	BEL	039	'	071	G	103	g
008	BS	040	(	072	H	104	h
009	HT	041	)	073	I	105	i
010	LF	042	*	074	J	106	j
011	VT	043	+	075	K	107	k
012	FF	044	,	076	L	108	l
013	CR	045	-	077	M	109	m
014	SO	046	.	078	N	110	n
015	SI	047	/	079	O	111	o
016	DLE	048	0	080	P	112	p
017	DC1	049	1	081	Q	113	q
018	DC2	050	2	082	R	114	r
019	DC3	051	3	083	S	115	s
020	DC4	052	4	084	T	116	t
021	NAK	053	5	085	U	117	u
022	SYN	054	6	086	V	118	v
023	ETB	055	7	087	W	119	w
024	CAN	056	8	088	X	120	x
025	EM	057	9	089	Y	121	y
026	SUB	058	:	090	Z	122	z
027	ESC	059	;	091	[	123	{
028	FS	060	<	092	\	124	|
029	GS	061	=	093	]	125	}
030	RS	062	>	094	↑	126	~
031	US	063	?	095	_	127	DEL

หมายเหตุ: อักขระ 32 ตัวแรกและอักขระสุดท้ายเป็นอักขระควบคุม ไม่สามารถพิมพ์ออกได้

ค่าคงที่อักขระ

- ตัวอักษรที่ไม่สามารถพิมพ์ได้ รวมถึง อัญประกาศ (“) และอะโพสโซพี (‘) เครื่องหมายคำถาม (?) และแบ็กสแลช (/) จะแสดงในลำดับหลีก
- ลำดับหลีก
  - จะขึ้นต้นด้วยแบ็กสแลช และตามด้วยอักขระพิเศษอย่างน้อยหนึ่งตัว

อักขระ	*ลำดับหลีก*	*ค่า ASCII*
bell	\a	007
backspace	\b	008
horizontal tab	\t	009
vertical tab	\v	011
newline	\n	010
form feed	\f	012
carriage return	\r	013
quotation mark (")	\"	034
apostrophe (')	\'	039
question mark (?)	\?	063
backslash (\)	\\	092
null	\0	000

ค่าคงที่สตริง

- ประกอบด้วยอักขระที่เรียงต่อกัน หรือ อาจไม่มีอักขระเลยก็ได้ อยู่ภายใต้เครื่องหมายอัญประกาศ
- ตัวอย่าง

```
"green"    "Washington, D.C. 2005"          "270-32-3456"
"$19.95"   "THE CORRECT ANSWER IS:"         "2*(I+3)/J"
" "        "Line 1\nLine 2\nLine 3"         ""
```

- พิจารณาข้อความต่อไปนี้ จะแสดงผลลัพธ์อย่างไร
  - “Line1\nLine2\nline3\n”
- ตัวแปลโปรแกรมภาษาซีจะเติมอักขระนัล (\0) เป็นอักขระสุดท้ายของค่าคงที่สตริงทุกตัวโดยอัตโนมัติ

ตัวแปร (Variable)

- เป็นชื่อที่ใช้แทนการอ้างอิงข้อมูลที่เก็บใน Memory
  - ตัวแปร ที่เก็บค่าต่างๆในโปรแกรม เช่น (x, y, sum, …) จะถูกจองเนื้อที่ไว้ใน Address หนึ่งๆใน Memory
  - ค่าของตัวแปร เปลี่ยนได้ขณะประมวลผล

การประกาศตัวแปร

- รูปแบบ

<ประเภทข้อมูล> <ชื่อตัวแปร>;

- ตัวอย่าง

```
int k;
int num1, num2, sum;
float mean;
char grade;
char grade[15];
```

Note: ตัวแปร grade สามารถเก็บตัวอักษรได้เพียงตัวเดียวเท่านั้น ในกรณีที่จะเก็บข้อมูลเป็นคำหรือเป็นประโยค จะต้องนำตัวอักษรหลายตัวมารวมกัน เรียกว่า string จะต้องใช้โครงสร้างข้อมูลแบบ array ดังตัวอย่าง สามารถเก็บ string ที่มีความยาว 14 ตัว (รายละเอียดเกี่ยวกับ array จะกล่าวหลังกลางภาค)

การตั้งชื่อตัวแปร

- ประกอบด้วย ตัวอักษร ตัวเลข และเครื่องหมาย underscore (_)
- ตัวแรกจะต้องเป็น ตัวอักษร หรือเครื่องหมาย underscore (_)
  หมายเหตุ: เครื่องหมาย underscore (_) ไม่เป็นที่นิยมใช้เป็นตัวแรกในทางปฏิบัติ
- ควรตั้งชื่อให้สื่อความหมาย ส่วนมากตัวแปรภาษา C จะกำหนดให้การตั้งชื่อมีความยาวไม่เกิน 32 ตัวอักขระเท่านั้น
- ห้ามเว้นวรรคตัวอักขระในการตั้งชื่อ ถ้าต้องการนำหลายคำมาใช้ในการตั้งชื่อควรคั่นด้วย (_) เช่น first_name และ student_id เป็นต้น
- ชื่อที่ตั้งเป็นตัวพิมพ์ใหญ่และตัวพิมพ์เล็ก ถือว่าเป็นคนละชื่อกัน
- ห้ามตั้งชื่อซ้ำกับคำสงวน

คำสงวนในภาษาซี

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

Note: คำในสองบรรทัดสุดท้ายเป็นคำสงวนในบางคอมไพเลอร์

การกำหนดค่าเริ่มต้นให้กับตัวแปร

- เมื่อมีการกำหนดตัวแปรขึ้น ตัวแปลภาษาจะกำหนดตำแหน่งที่อยู่ในหน่วยความจำให้กับตัวแปร และจะมีการกำหนดค่าให้กับตัวแปรนั้นเสนอ โดยค่านี้จะถูกสุ่มขึ้นมาเมื่อเปิดเครื่องคอมพิวเตอร์ หรือเป็นค่าข้อมูลที่เกิดจากโปรแกรมที่มีการทำงานก่อนหน้านี้

- ดังนั้น การกำหนดค่าเริ่มต้นให้กับตัวแปรจึงเป็นการเก็บค่าเริ่มต้นที่ผู้พัฒนาโปรแกรมกำหนดขึ้นไว้ในตำแหน่งที่อยู่ในหน่วยความจำที่มีการอ้างอิงตัวแปรนั้น เราเรียกกระบวนการนี้ว่า การกำหนดค่าเริ่มต้น (Initializing)

การกำหนดค่าเริ่มต้นให้กับตัวแปร

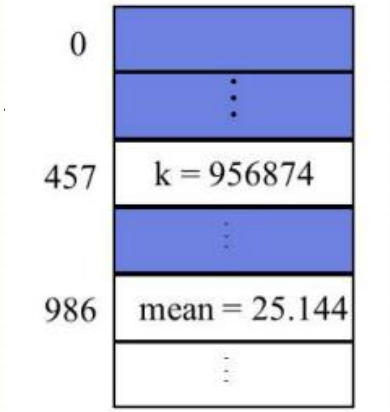
- รูปแบบการกำหนดค่าเริ่มต้น

<ชื่อตัวแปร> = <ค่าที่ต้องการกำหนด>

- ตัวอย่าง

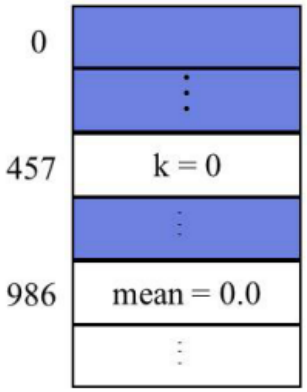
1. ประกาศตัวแปร k และ mean

```
int k;
float mean;
```



2. กำหนดค่าเริ่มต้น

```
K=0;
Mean=0;
```



Exercise

- จากการประกาศตัวแปรพร้อมกำหนดค่าดังแสดงต่อไปนี้ ข้อใดถูกและข้อใดผิดโดยให้บอกส่วนที่ผิดมาด้วย
  - int a – ผิดเพราะไม่มี ;
  - int main =10;
  - int num = 25,00;
  - float x = 45.1;
  - char string = “Hello”;

นิพจน์

- การนำตัวแปร ค่าคงที่ มาสัมพันธ์กันโดยใช้เครื่องหมายอย่างใดอย่างหนึ่งเป็นตัวเชื่อม เช่น

นิพจน์คณิตศาสตร์	นิพจน์ภาษาซี
$\frac{a+b}{c \times d}$	(a+b)/(c*d)
$10x - 3xy + 10y^2$	10*x - 3*x*y + 10*y*y

- ข้อสังเกต ห้ามเขียนตัวแปรสองตัวติดกันโดยไม่มีเครื่องหมาย เช่น ab ในภาษาซี ต้องเขียน a*b (More details in next chapter)

ประโยค (Statements)

- ประโยคเป็นตัวสั่งงานให้คอมพิวเตอร์ทำงาน ในภาษาซีมีประโยคอยู่ 3 แบบ คือ
  - ประโยคนิพจน์ (Expression statements)
  - ประโยคเชิงซ้อน (Compound statements)
  - ประโยคควบคุม (Control statements)

ประโยคนิพจน์ (Expression statements)

- ประกอบด้วย นิพจน์และปิดท้ายด้วยเครื่องหมายอัฒภาค (;) เมื่อมีการอ่านประโยคนิพจน์เข้าไปประมวลผล โปรแกรมจะทำงานตามนิพจน์นั้น ดังตัวอย่าง เช่น

ตัวอย่างที่ 1

```
a = 3;
c = a + b;
```

เป็นการกำหนดค่า โดยนำค่าจากนิพจน์ที่อยู่ทางขวาของเครื่องหมายเท่ากับไปกำหนดให้กับตัวแปรที่อยู่ทางซ้าย

ตัวอย่างที่ 2

Printf(“The value of a is %d”, a);

เป็นการใช้งานฟังก์ชัน **printf** เพื่อแสดงข้อความ **The value of a is** ตามด้วยค่าที่เก็บอยู่ในตัวแปร **a** นั่นคือ **a** มีค่า 3

ดังนั้นข้อความที่แสดงออกมาคือ

The value of a is 3

ประโยคเชิงซ้อน (Compound statements)

- ประกอบด้วยประโยคหลายๆ ประโยคซึ่งอยู่ภายใต้วงเล็บปีกกา ({ }) ซึ่งแต่ละประโยคเหล่านี้อาจเป็นประโยคนิพจน์ ประโยคเชิงซ้อน หรือประโยคควบคุมก็ได้ ตัวอย่างเช่น

```
{
        pi = 3.141593;
        circumference = 2*pi*radius;
        are = pi*radius*radius;
}
```

- ข้อสังเกต ท้ายเครื่องหมาย } ห้ามใส่ ;

ประโยคควบคุม (Control statements)

- ใช้เพื่อสร้างคุณลักษณะพิเศษของโปรแกรม เช่น ทดสอบตรรกะ การวน (loop) การกระโดด (branch) ประโยคควบคุมหลายๆ ประโยคต้องมีประโยคอื่นๆ ฝังอยู่ในตัวมัน เช่น

- ตัวอย่าง

```
while (count <= n)
{
      printf("x = ");
      scanf("%f",&x);
      sum += x;
      count++;
}
```

ประโยคควบคุมในตัวอย่างนี้ประกอบด้วยประโยคเชิงซ้อน 1 ประโยค ซึ่งประกอบด้วยประโยคนิพจน์ 4 ประโยค โปรแกรมจะทำงานตามประโยคเชิงซ้อนนี้ไปเรื่อย ๆ ตราบใดที่ค่าของ count ยังไม่เกินค่า n ให้สังเกตว่า count จะมีค่าเพิ่มขึ้นในแต่ละรอบของการทำงาน

การแสดงผลและการรับข้อมูล

- การแสดงผล
  - printf
  - putchar
  - puts
- การรับข้อมูล
  - scanf
  - getchar และ getch
  - gets

printf

- เป็นฟังก์ชันที่ใช้ในการแสดงผลข้อมูลออกทางจอภาพ

- รูปแบบ

```
printf(control string, argment list);
```

  - argument list : เป็นค่าตัวแปร ค่าคงที่ หรือ นิพจน์ที่ต้องการนำมาแสดงผล ถ้ามีมากกว่า 1 ค่าจะแยกกันโดยใช้เครื่องหมายคอมม่า (,)
  - control string: จะต้องเขียนอยู่ภายใต้เครื่องหมาย “ ” ซึ่งสามารถเขียนได้ 2 ลักษณะ คือ
    - เป็บข้อความที่เป็นคำอธิบายที่ต้องการให้แสดงผลออกมา ส่วนมากจะใช้เพื่อแต่งรูปแบบของรายงานให้สวยงาม เช่น printf(“sum of x is”);
    - เป็นรหัสรูปแบบ (format Code) ที่ใช้ในการแสดงผล ซึ่งทุกรหัสรูปแบบจะต้องอยู่ตามหลังเครื่องหมาย

printf

รหัสรูปแบบข้อมูลที่นิยมใช้ แสดงดังต่อไปนี้

รหัสรูปแบบ	ความหมาย
%c	ใช้กับตัวแปรที่เก็บค่าที่เป็นตัวอักษรเพียงตัวเดียว
%s	ใช้กับตัวแปรที่เก็บค่าที่เป็นข้อความที่เก็บเป็นตัวแปรชุด
%d	ใช้กับตัวแปรที่เก็บค่าที่เป็นที่เป็นเลขจำนวนเต็ม
%u	ใช้กับตัวแปรที่เก็บค่าที่เป็นที่เป็นเลขจำนวนเต็มบวก
%f	ใช้กับตัวแปรที่เก็บค่าที่เป็นเลขทศนิยม
%e	ใช้กับตัวแปรที่เก็บค่าที่เป็นเลขทศนิยมในรูป e ยกกำลัง
%x	ใช้กับตัวแปรที่เก็บค่าที่เป็นค่าเลขฐานสิบหก
%o	ใช้กับตัวแปรที่เก็บค่าที่เป็นค่าเลขฐานแปด

printf

- ตัวอย่าง

```
int a, b;
a = 10;
b = 15;
printf("The sum of %d and %d is %d \n", a, b, a+b);
```

- ในที่นี้ผลลัพธ์ที่ได้ คือ

```
The sum of 10 and 15 is 25
```

printf

- ตัวอย่างที่ 2

```
int j = 45, k = -123;
float x = 12.34;
char c = 'w';
char *m = "Hello";
printf("Hello\n");
printf("Tax is 10%%\n);
printf("%d\n", j);
printf("%d\n", k);
printf("%f\n", x);
printf("%c\n", c);
printf("%s\n", m);
```

ผลลัพธ์ที่ได้ **??**

putchar

- แสดงผลตัวอักษรหรืออักขระ
- รูปแบบ
  - putchar(ch);
  - ch คือ ตัวอักษรหรืออักขระเขียนอยู่ภายในเครื่องหมาย ‘’ หรือตัวแปรชนิด char
- ตัวอย่าง
  - putchar(‘w’); → แสดงตัวอักษร w ออกทางหน้าจอ
  - char letter = ‘a’;
  - putchar(letter); → แสดงค่าของตัวแปร letter ออกทางหน้าจอ คือ ตัวอักษร a

puts

- แสดงผลข้อความ

- รูปแบบ
  - puts(string)
  - string คือ ข้อความที่อยู่ภายใต้เครื่องหมาย “” หรือตัวแปรที่เก็บข้อมูลชนิดข้อความ

- ตัวอย่าง
  - puts(“wararat”); → แสดงข้อความ wararat ทางจอภาพ

  - char greeting[10] = “Hello”;
  - puts(greeting); → แสดงผลข้อความในตัวแปร greeting

scanf

- รับข้อมูลจากคีย์บอร์ด

- รูปแบบ
  - scanf(“format”, &variable);
  - format การใช้รหัสควบคุม เพื่อกำหนดชนิดของข้อมูล
  - variable ตัวแปรที่ใช้เก็บค่าที่รับเข้ามาจากทางคีย์บอร์ด ต้องตรงกับรหัสควบคุมที่กำหนดไว้ใน format หน้าชื่อตัวแปรมี & ยกเว้นตัวแปรที่เป็นข้อความ

- ตัวอย่าง

```
int number;
printf("Enter a number: ");
scanf("%d",&number);

int name[10];
printf("Enter a name: ");
scanf("%s",number);
```

```
Enter a number: 10
Enter a name: wararat
```

scanf

int number; → สร้างตัวแปรชนิด int สำหรับเก็บค่าจำนวนเต็ม

printf("Enter a number: "); → แสดงข้อความให้กรอกค่าตัวเลข

scanf("%d",&number); → รับค่าตัวเลขเข้ามาเก็บไว้ในตัวแปร

int name[10]; → สร้างตัวแปรสตริงสำหรับเก็บข้อความ

printf("Enter a name: "); → แสดงข้อความให้ป้อนชื่อ

scanf("%s",number); → รับชื่อเข้ามาเก็บไว้ในตัวแปร name ตัวแปรนี้ไม่ใส่ &

getchar and get

- รับข้อมูลประเภทตัวอักษรหรืออักขระ

- รูปแบบ
  - ch = getchar();
  - ch = getch();
  - ch เป็นตัวแปรชนิด char เพื่อเก็บค่าตัวอักษร

- ตัวอย่าง

```
char anser;
printf("Enter your answer (y or n):");
answer = getchar();
```

gets

- รับข้อความ
- รูปแบบ
  - gets(string);
  - string เป็นตัวแปรสำหรับเก็บข้อความ
- ตัวอย่าง

```
char name[10];
printf("Enter name:");
gets(name);
```

* หากข้อความที่รับเข้ามาเป็นประโยคและมีช่องว่างด้วย ควรใช้ gets