

บทที่ 2

พื้นฐานภาษา C

(C Fundamentals)

ในบทนี้ เราจะกล่าวถึงรายละเอียดพื้นฐานต่าง ๆ ที่นำมาใช้ในการสร้างประโยคคำสั่งในภาษา C โดยรายละเอียดพื้นฐานดังกล่าว จะประกอบด้วย เซตของอักขระ (Character Set), ชื่อ (Identifiers), คำหลัก (Keyword), ชนิดข้อมูล (Data Types), ค่าคงตัว (Constants), ตัวแปร (Variable)

2.1 เซตของอักขระในภาษา C (The C Character set)

จะประกอบด้วย

1. ตัวอักษร (Letter) ได้แก่ A ถึง Z, a ถึง z
2. ตัวเลข (Digital) ได้แก่ 0 ถึง 9
3. อักขระพิเศษ (Special Character) ต่าง ๆ มีดังรายการต่อไปนี้

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

2.2 ชื่อ (Identifiers)

เป็นเซตของอักขระที่ประกอบด้วยตัวอักษร, ตัวเลข และอักขระขีดล่าง (_) และห้ามขึ้นต้นด้วยตัวเลข

2.3 คำหลัก (Keywords) หรือ คำสงวน (Reserved words)

เป็นคำที่มีความหมายพิเศษให้กับคอมพิวเตอร์ของ C ดังนั้น นักเขียนโปรแกรม (Programmer) จะต้องระมัดระวัง ไม่ควรใช้คำเหล่านี้มาตั้งชื่อเป็นตัวแปรโดยเด็ดขาด

คำหลัก หรือ คำสงวนในภาษา C มีดังรายการต่อไปนี้

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	

ชื่อ (Identifier) เราจะแบ่งออกเป็น 2 แบบ คือ

1. ชื่อมาตรฐาน (Standard identifiers)
2. ชื่อที่ผู้ใช้นิยามขึ้นมาเอง (User-defined identifiers)

2.4 ชื่อมาตรฐาน

เป็นคำที่มีความหมายพิเศษที่ภาษา C ได้กำหนดขึ้นมา เช่น printf และ scanf เป็นชื่อที่ถูกนิยามในไลบรารีแสดงและรับข้อมูล

2.5 ชื่อที่ผู้ใช้นิยามขึ้นมาเอง เพื่อนำมาใช้ในโปรแกรม

2.6 กฎเกณฑ์ในการตั้งชื่อ (Rules for Constructing Identifiers) มีดังนี้

1. ชื่อจะต้องประกอบด้วยอักษร A ถึง Z, a ถึง z, ตัวเลข 0 ถึง 9 และอักขระขีดล่าง (_)
2. อักขระตัวแรก จะต้องเป็นตัวอักษรหรืออักขระขีดล่าง

3. ความยาวของชื่อจะเป็นเท่าไรก็ได้ แต่คอมไพเลอร์ของภาษา C จะแยกความแตกต่างของอักขระ 31 ตัวแรกเท่านั้น ตามมาตรฐานของ ANSI C (คอมไพเลอร์ของ C บางตัว จะแยกความแตกต่างเพียง 8 ตัวอักขระแรกเท่านั้น)

หมายเหตุ ANSI ย่อมาจาก American National Standards Institute

4. ห้ามมีช่องว่าง

5. ห้ามนำคำหลักหรือคำสงวนมาใช้ในการตั้งชื่อ

6. ชื่อที่ตั้งในภาษา C จะมีลักษณะเป็น Case Sensitive หมายความว่า อักษรตัวเล็กกับอักษรตัวใหญ่ จะถือว่ามีความแตกต่างกัน เช่น Tax และ tax ซึ่งชื่อทั้ง 2 นี้ ตั้งได้ถูกต้อง แต่เราจะถือว่ามีความแตกต่างกัน

ตัวอย่างที่ 2.1 ชื่อที่ตั้งได้ถูกต้อง (Valid Identifiers)

x	y12	sum_1	_temperature
names	area	tax_rate	TABLE

แต่ถ้ามีความยาวมาก ซึ่งโดยปกติแล้วกฎเกณฑ์การตั้งชื่อจะไม่ได้กำหนดความยาวที่แน่นอน แต่มาตรฐานของ ANSI C จะพิจารณาแยกความแตกต่างระหว่าง 2 ชื่อ เฉพาะอักขระ 31 ตัวแรกเท่านั้น เช่น

```
per_capita_meat_consumption_in_1980
per_capita_meat_consumption_in_1995
```

ตัวอย่างที่ 2.2 ชื่อที่ตั้งไม่ถูกต้อง (Invalid Identifiers)

4Letter	ผิด เพราะว่า ขึ้นต้นด้วยตัวเลข
double	ผิด เพราะว่า เป็นคำสงวน
int	ผิด เพราะว่า เป็นคำสงวน
TWO*FOUR	ผิด เพราะว่า ไม่อนุญาตให้ใช้ตัวอักขระ *
joe's	ผิด เพราะว่า ไม่อนุญาตให้ใช้ตัวอักขระ '
"x"	ผิด เพราะว่า ไม่อนุญาตให้ใช้ตัวอักขระ "
order-no	ผิด เพราะว่า ไม่อนุญาตให้ใช้ -
error flag	ผิด เพราะว่า มีอักขระว่าง

การตั้งชื่อ ควรจะตั้งชื่อที่ง่ายต่อความเข้าใจ (Easy to Understand) และมีความหมายที่ชัดเจน (Meaning is clear) เช่น ถ้าเราต้องการตั้งชื่อเก็บข้อมูลเกี่ยวกับเงินเดือน เราควรจะตั้งชื่อว่า Salary แทนที่จะใช้ S หรือ Bagel ซึ่งดูแล้วไม่สื่อความหมายเลย ถ้าชื่อประกอบด้วย 2 คำ หรือมากกว่า เราควรจะใช้ขีดล่าง (_) ระหว่างคำ เพื่อปรับปรุงทำให้ชื่อนั้นอ่านง่ายและดูดี เช่น dollars_per_hour แทนที่จะใช้ dollarsperhour แต่พยายามหลีกเลี่ยงการตั้งชื่อที่ยาว เนื่องจากอาจจะมีผลต่อการพิมพ์ชื่อผิดได้

2.7 ค่าคงตัว (Constants)

ในภาษา C จะมีค่าคงตัวพื้นฐานอยู่ 4 แบบ คือ

- 2.7.1 ค่าคงตัวจำนวนเต็ม (Integer Constants)
- 2.7.2 ค่าคงตัวจุดลอยตัว (Floating-point Constants)
- 2.7.3 ค่าคงตัวอักขระ (Character Constants)
- 2.7.4 ค่าคงตัวสายอักขระ (String Constants)

2.7.1. ค่าคงตัวจำนวนเต็ม

ซึ่งในภาษา C จะมีตัวเลขอยู่ 3 แบบ คือ ตัวเลขจำนวนเต็มฐานสิบ, ตัวเลขจำนวนเต็มฐานแปด และตัวเลขจำนวนเต็มฐานสิบหก

กฎเกณฑ์ในการพิจารณาค่าคงตัวจำนวนเต็ม มีดังนี้

- 1) ห้ามมีเครื่องหมายต่าง ๆ และช่องว่างระหว่างตัวเลข
- 2) จำนวนเต็มใด ๆ ถ้าไม่มีเครื่องหมายนำหน้า จะถือว่าเป็นเครื่องหมายบวก
- 3) ถ้ามีตัวเลขมากกว่า 1 หลัก ห้ามใส่เลขศูนย์นำหน้า เนื่องจากภาษา C จะตีความหมายว่าเป็นตัวเลขจำนวนเต็มฐานแปด

ตัวอย่างที่ 2.3 ค่าคงตัวจำนวนเต็มฐานสิบที่ถูกต้อง ได้แก่

0, 320, 3456

ตัวอย่างที่ 2.4 ค่าคงตัวจำนวนเต็มฐานสิบที่ไม่ถูกต้อง ได้แก่

1,345 ผิด เนื่องจากมีเครื่องหมาย Comma (,)

10.1 ผิด เนื่องจากมีจุดทศนิยม

10 20 ผิด เนื่องจากมีช่องว่าง

12-30 ผิด เนื่องจากมีเครื่องหมายลบ (-)

0900 ผิด เนื่องจากมีเลขศูนย์นำหน้า ซึ่งจะหมายถึงตัวเลขจำนวนเต็มฐานแปด

จำนวนเต็มฐานแปด ในภาษา C จะกำหนดให้มีเลขศูนย์นำหน้า

ตัวอย่างที่ 2.5 จำนวนเต็มฐานแปดที่ถูกต้อง ได้แก่ 0, 06, 045

ตัวอย่างที่ 2.6 จำนวนเต็มฐานแปดที่ไม่ถูกต้อง ได้แก่

7 ผิด เนื่องจากไม่มีเลขศูนย์นำหน้า

019 ผิด เนื่องจากระบบฐานแปดจะมีตัวเลขโดดที่ใช้คือ 0 ถึง 7

03,442 ผิด เนื่องจากมีเครื่องหมายจุลภาค (,)

จำนวนเต็มฐานสิบหก ในภาษา C จะกำหนดให้ขึ้นต้นด้วย 0X หรือ 0x โดยตัวเลข ที่ใช้ในระบบฐานสิบหก จะประกอบด้วยตัวเลข 0 ถึง 15 แต่ตัวเลขตั้งแต่ 10 ถึง 15 จะแทนด้วยอักษรภาษาอังกฤษ A ถึง F หรือ a ถึง f

ตัวอย่างที่ 2.7 จำนวนเต็มฐานสิบหกที่ถูกต้อง ได้แก่

0X, 0X123, 0XABC, 0xabc

ตัวอย่างที่ 2.8 จำนวนเต็มฐานสิบหกที่ไม่ถูกต้อง ได้แก่

0X1.3 ผิด เนื่องจากมีจุด

0XEH ผิด เนื่องจากใช้อักษร H

2.7.2. ค่าคงตัวจุดลอยตัว

จะเป็นตัวเลขฐานสิบที่มีจุดทศนิยม หรือเขียนให้อยู่ในรูปของเลขชี้กำลัง ที่เป็นสัญลักษณ์ทางวิทยาศาสตร์ กล่าวคือ จะเขียนจำนวนนั้นให้อยู่ในรูป $A \times 10^n$ เมื่อ $1 \leq A < 10$, n เป็นจำนวนเต็มใด ๆ เช่น $63200 = 6.3200 \times 10^4$

ตัวอย่างที่ 2.9 ค่าคงตัวจุดลอยตัวที่ถูกต้อง ได้แก่

0., 1., 0.2, 827.602, 0.003

2E+8, 0.006e-3, .12e12

ตัวอย่างที่ 2.10 ค่าคงตัวจุดลอยตัวที่ไม่ถูกต้อง ได้แก่

2 ผิด เนื่องจากไม่มีจุดต่อท้าย

2,130.0 ผิด เนื่องจากมีเครื่องหมาย จุดภาค (,)

3E + 1.2 ผิด เนื่องจากส่วนเลขนี้อาจต้องเป็นเลขจำนวนเต็ม

3E 12 ผิด เนื่องจากมีช่องว่าง

หมายเหตุ 1) จำนวน 20.35 สามารถเขียนได้เป็น 0.2035×10^2 , 2.035×10^1 , 20.35×10^0 , 203.5×10^{-1} เป็นต้น

0.2035×10^2 ในภาษา C จะเขียนแทนด้วย 0.2035e2

2.035×10^1 ในภาษา C จะเขียนแทนด้วย 2.035e1

20.35×10^0 ในภาษา C จะเขียนแทนด้วย 20.35e0

203.5×10^{-1} ในภาษา C จะเขียนแทนด้วย 203.5e-1

2) ค่าของ $125.5 = +125.5$

3) ค่าของ $-15.0 = -15.$

4) ค่าของ $0 = 0.0 = .0 = 0.$

2.7.3 ค่าคงตัวอักขระ

จะเป็นอักขระเพียง 1 ตัว ที่ปิดล้อมโดยเครื่องหมาย apostrophes (single quotation marks) (' ')

ตัวอย่างที่ 2.11 ค่าคงตัวอักขระที่ถูกต้อง ได้แก่

'A', ' ', '2', '\$'

ตัวอย่างที่ 2.12 ค่าคงตัวอักขระที่ไม่ถูกต้อง ได้แก่

'AB' ผิด เนื่องจากภายในเครื่องหมาย ' ' มีอักขระมากกว่า 1 ตัว

A ผิด เนื่องจากไม่มีเครื่องหมาย ' ' ปิดล้อม

ค่าคงตัวอักขระจะเป็นรหัสที่ใช้ในคอมพิวเตอร์ที่ถูกเรียกว่า รหัสแอสกี (ASCII) ซึ่งจะมีการกำหนดค่าให้กับรหัสแต่ละตัว

หมายเหตุ 1) ASCII ย่อมาจาก American Standard Code for Information Interchange

2) รหัส ASCII แต่ละตัวจะถูกแทนด้วยตัวเลข 7 บิต ดังนั้น จึงมีรหัสที่เป็นไปได้

$$\text{ทั้งหมด} = 2^7 = 128 \text{ ตัวอักขระ}$$

ตารางที่ 2.1 เซตของอักขระ ASCII (The ASCII Character Set)

ASCII Value	Character	ASCII Value	Character	ASCII Value	Character	ASCII Value	Character
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	100	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

Note : The first 32 characters and the last character are control character; they cannot be printed.

(ข้อสังเกต : อักขระ 32 ตัวแรก และอักขระตัวสุดท้าย จะเป็นตัวอักขระที่ใช้ในการควบคุม ไม่สามารถนำมาพิมพ์ได้)

ตัวอย่างที่ 2.13 จากตาราง 2.1 ค่าคงตัวอักขระของรหัส ASCII จะมีการกำหนดค่าดังนี้

'A'	มีค่า 65
'a'	มีค่า 97
'\$'	มีค่า 36
'3'	มีค่า 51

2.7.4 ค่าคงตัวสายอักขระ

ได้แก่ ชุดของอักขระที่นำมาเขียนให้ติดต่อกัน และปิดล้อมด้วยเครื่องหมาย double quotation marks (“ ”)

ตัวอย่างที่ 2.14 ค่าคงตัวแบบสายอักขระที่ถูกต้อง ได้แก่

“125” , “27-10-99” , “” , “what is it?”

หมายเหตุ 1) ถ้าต้องการพิมพ์ผลลัพธ์ดังนี้

Jim “Mac” MacDonald.

ให้ใส่ลำดับหลักดังนี้

“Jim \“Mac\“MacDonald.”

2) “” เป็นสายอักขระที่ไม่มีตัวอักขระ เราจะเรียกว่า null (empty) string

3) “Line1\n Line2\n line3” จะแสดงผลบนจอภาพดังนี้

Line1

Line2

Line 3

เนื่องจากมีการใช้ลำดับหลัก \n หมายถึงขึ้นบรรทัดใหม่

หมายเหตุ ค่าคงตัวอักขระ 'x' กับค่าคงตัวสายอักขระ “x” จะมีความแตกต่างกัน กล่าวคือ

ค่าคงตัวอักขระ 'x' จะเก็บข้อมูล ดังรูป

x

ค่าคงตัวสายอักขระ “x” จะเก็บข้อมูล ดังรูป

x	\0
---	----

โดย \0 เป็นตัวอักขระที่เรียกว่า อักขระว่าง (null character) ที่ใช้เติมท้ายเพื่อบอกจุดสิ้นสุดของสายอักขระ ภาษา C จะเติมให้โดยอัตโนมัติ

2.8 ลำดับหลัก (Escape Sequences)

เครื่องหมาย backslash (\) จะถูกเรียกว่า อักขระหลัก (Escape Character) ซึ่งจะเป็นตัวที่บอกให้คำสั่ง printf ทำบางสิ่งบางอย่างนอกเหนือจากงานปกติ และเมื่อนำมารวมกับอักขระจะถูกเรียกว่าลำดับหลัก (Escape Sequences) ซึ่งจะมีความหมายพิเศษตามอักขระที่ตามหลังเครื่องหมาย \ โดยเมื่อคอมพิวเตอร์พบเครื่องหมาย \ ในสายอักขระ (string) มันก็จะมองหาอักขระตัวถัดไป เมื่อพบก็จะทำตามคำสั่งที่ได้กำหนดไว้ เช่น \n เป็นลำดับหลัก ซึ่งมีความหมายว่าให้ขึ้นบรรทัดใหม่ (Newline) นั่นคือ เมื่อมีการใช้ลำดับหลัก \n ในคำสั่ง printf ก็จะทำให้มีการขึ้นบรรทัดใหม่ โดยเคอร์เซอร์จะไปอยู่ที่ตำแหน่งแรกของบรรทัดใหม่

หมายเหตุ อักขระที่ตามหลังเครื่องหมาย \ อาจจะมี 1 ตัวหรือมากกว่า 1 ตัว ก็ได้

ตารางที่ 2.2 แสดงลำดับหลัก

ตัวอักขระ (Character)	ลำดับหลัก (Escape Sequence)	ค่ารหัสแอสกี (ASCII Value)
bell (alert)	\a	007
backspace	\b	008
horizontal tab	\t	009
vertical tab	\v	011
newline (line feed)	\n	010
form feed	\f	012
carriage return	\r	013
quotation mark (")	\"	034
apostrophe (')	\'	039
question mark (?)	\?	063
backslash (\)	\\	092
null	\0	000

2.9 ชนิดของข้อมูล (Types of data)

ในภาษา C จะแบ่งประเภทข้อมูลออกเป็น 2 แบบ คือ

1. จำนวน (Numbers) ยังแบ่งออกเป็น 2 แบบ คือ

- จำนวนเต็ม (Integer) และ
- จำนวนที่มีจุดทศนิยม (Float)

2. ตัวอักษร (Characters)

3. สายอักขระ (String)

โดยในภาษา C ได้กำหนดชื่อที่เป็นคำเฉพาะให้กับข้อมูลแต่ละประเภท ดังนี้

- ข้อมูลประเภทชนิดจำนวนเต็ม จะเขียนแทนด้วย int
- ข้อมูลประเภทชนิดจำนวนที่มีจุดทศนิยม จะเขียนแทนด้วย float
- ข้อมูลประเภทชนิดตัวอักษร จะเขียนแทนด้วย char

ตารางที่ 2.3 แสดงชนิดข้อมูล และพิสัยของค่าที่ใช้ใน IBM PC.

ชนิด	ความหมาย	พิสัยของค่า
void	ว่าง (null)	0
char	ตัวอักษร	-128 ถึง 127
int	จำนวนเต็ม	-32,768 ถึง 32,767
short	จำนวนเต็มแบบ short	-32,768 ถึง 32,767
long	จำนวนเต็มแบบ long	-2,147,483,648 ถึง 2,147,483,647
unsigned char	ตัวอักษรไม่มีเครื่องหมาย	0 ถึง 255
unsigned	จำนวนเต็มไม่มีเครื่องหมาย	0 ถึง 65,535
unsigned short	จำนวนเต็มไม่มีเครื่องหมายแบบ short	0 ถึง 65,535
unsigned long	จำนวนเต็มไม่มีเครื่องหมายแบบ long	0 ถึง 4,294,967,295
float	จุดลอยตัว	3.4E +/- 38 (7 digits)
double	จุดลอยตัวแบบ double	1.7E +/- 308 (15 digits)
long double	จุดลอยตัวแบบ long double	1.7E +/- 4932 (15 digits)

ตารางที่ 2.4 แสดงขนาดชนิดข้อมูล

ชนิดข้อมูล	ขนาด (ไบต์)
จำนวนเต็ม (Integer)	
มีเครื่องหมาย (Signed)	
จำนวนเต็ม แบบ short (short, short int, signed short, signed short int)	2
จำนวนเต็ม (int, signed, signed int)	2
จำนวนเต็ม แบบ long (long, long int, signed long, signed long int)	4
ไม่มีเครื่องหมาย (Unsigned)	
จำนวนเต็ม แบบ short (unsigned short, unsigned short int)	2
จำนวนเต็ม (unsigned, unsigned int)	2
จำนวนเต็ม แบบ long (unsigned long, unsigned long int)	4
จุดลอยตัว (Floating-point)	
จุดลอยตัว (float)	4
จุดลอยตัว แบบ double (double)	8
จุดลอยตัว แบบ long double (long double)	16
ตัวอักษร (Character)	
ตัวอักษร (char)	1

2.10 การประกาศตัวแปร (Variable Declarations)

ก่อนที่จะมีการประมวลผล เราจะต้องมีการประกาศตัวแปรทุกตัวให้คอมพิวเตอร์ทราบว่าตัวแปรแต่ละตัวมีข้อมูลเป็นแบบใด เพื่อที่จะได้จองเนื้อที่ให้กับตัวแปรแต่ละตัวได้ถูกต้อง

ตัวอย่างที่ 2.15 การเขียนการประกาศตัวแปรที่เหมาะสมให้กับตัวแปรแต่ละตัว ดังต่อไปนี้

ตัวแปรจำนวนเต็ม : p, q	จะเขียนแทนด้วย int p, q ;
ตัวแปรจำนวนจุดลอยตัว : x, y, z	จะเขียนแทนด้วย float x, y, z ;
ตัวแปรอักขระ : a, b, c	จะเขียนแทนด้วย char a, b, c ;
ตัวแปรจำนวนเต็มแบบ long : l	จะเขียนแทนด้วย long l ;
ตัวแปรจำนวนเต็มแบบ short : t	จะเขียนแทนด้วย short t ;
ตัวแปรจำนวนเต็มแบบ unsigned : n	จะเขียนแทนด้วย unsigned n ;
ตัวแปรแถวลำดับของอักขระที่มีจำนวน 80 ตัว : message	จะเขียนแทนด้วย

char message [80] ;

ตัวอย่างที่ 2.16 ในโปรแกรมภาษา C ที่มีการประกาศ 2 แบบ ดังนี้

แบบที่ 1	แบบที่ 2
int a ;	int a, b, c ;
int b ;	
int c ;	

เราจะถือว่า การประกาศทั้ง 2 แบบนี้เหมือนกัน เพียงแต่ในแบบที่ 1 นั้น สามารถใส่หมายเหตุ (Comment) ให้กับตัวแปรแต่ละตัวเท่านั้น

2.11 การประกาศค่าคงตัวชื่อ (Name Constants Declarations)

จะเป็นกำหนดให้ตัวแปรนั้นคงตัว ไม่สามารถเปลี่ยนแปลงค่าได้

ตัวอย่างที่ 2.17 `Const double Tax = 0.07` จะทำให้ตัวแปร Tax ชนิด Double มีค่าเท่ากับ 0.07

2.12 การประกาศค่าคงตัวสัญลักษณ์ (Symbolic Constants Declarations)

จะมีรูปแบบการกำหนดโดยทั่วไป ดังนี้

```
#define name text
```

โดยที่ `define` เป็นคำสั่งว่ามีเครื่องหมาย `#` เติมหน้า

`name` จะเป็นค่าคงตัวสัญลักษณ์ ซึ่งจะต้องเขียนด้วยอักษรตัวใหญ่ทั้งหมด ซึ่งจะถูกแทนด้วยค่าคงตัวตัวเลข หรือค่าคงตัวอักขระ หรือค่าคงตัวสายอักขระ

ส่วน `text` อาจจะเป็นค่าคงตัวตัวเลข หรือค่าคงตัวอักขระ หรือค่าคงตัวสายอักขระ ที่จะนำไปแทนใน `name`

หมายเหตุ การประกาศค่าคงตัวสัญลักษณ์ ไม่ต้องมีเครื่องหมาย semicolon (;) เติมท้าย text

ตัวอย่างที่ 2.18 จะเป็นการแสดงการประกาศค่าคงตัวสัญลักษณ์ ดังต่อไปนี้

ค่าคงตัว text

FACTOR -18 จะเขียนเป็น `#define FACTOR -18`

ERROR 0.01 จะเขียนเป็น `#define ERROR 0.01`

```

BEGIN      {          จะเขียนเป็น #define BEGIN {
END        }          จะเขียนเป็น #define END }
NAME       "Sharon"   จะเขียนเป็น #define NAME "Sharon"
COST       "$18.95"   จะเขียนเป็น #define COST "$18.75"

```

ตัวอย่างที่ 2.19 จากตัวโปรแกรมที่ 1.6 ในบทที่ 1 จะมีคำสงวน, ชื่อมาตรฐาน และชื่อที่ผู้ใช้นิยามขึ้นมาเอง สรุปได้ดังนี้

คำสงวน	ชื่อมาตรฐาน	ชื่อที่ผู้ใช้นิยามขึ้นมาเอง
int, void, double, return	printf, scanf	KMS_PER_MILE, main, miles, kms