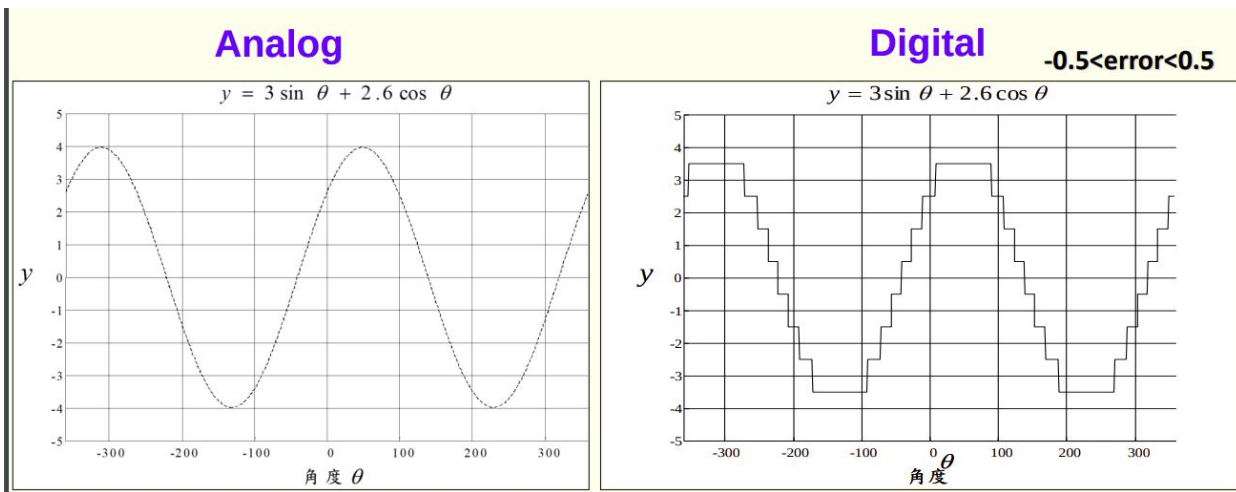


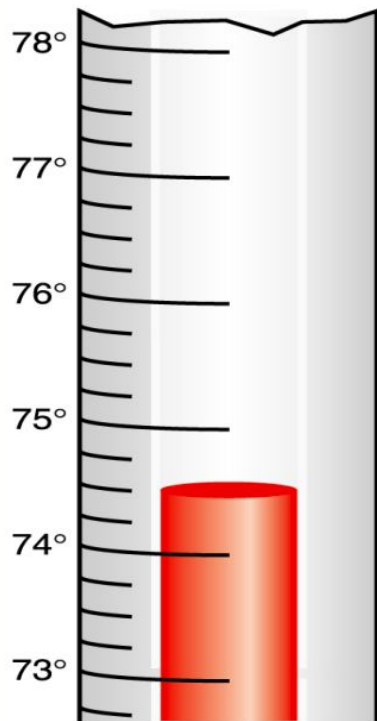
Chapter3-基本型別

類比與數位的觀念

- 討論 C 語言的 type 之前先來了解類比與數位的觀念
- 一筆資訊可以用**類比(analog)**或**數位(digital)**的形式來表示
 - 類比資料(analog data): 是一個連續性的表示法, 是資料所真正呈現的本質
 - 數位資料(digital data): 是一個離散的表示法, 把資訊用一個個分開的元素來表示



類比與數位的例子



溫度計

← Analog device(類比裝置)

74.568.. degree Fahrenheit

A mercury thermometer continually rises in direct proportion to the temperature

由於硬體資源是有限的(有限的記憶體、CPU), 因此用電腦硬體所表示的數值可能都會有誤差

耳溫槍

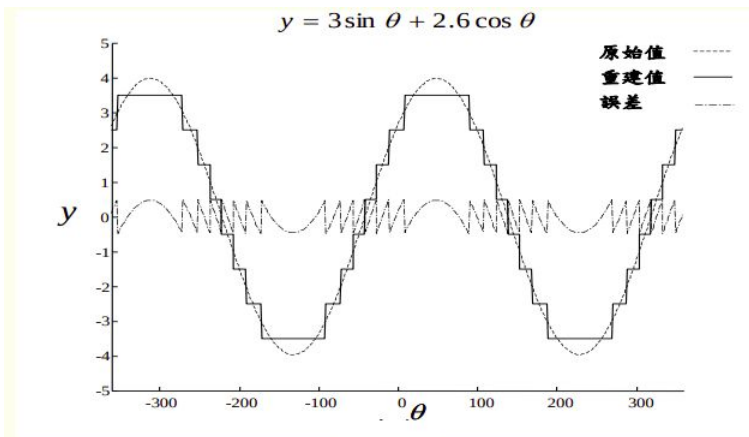
Digitize(數字化): The act of breaking information down into discrete pieces

Digital device: (數位裝置)

If the resolution is 0.01 degree, the answer is 74.56 or 74.57

If the resolution is 0.1 degree, the answer is 74.5 or 74.6

增加硬體的資源以減少誤差



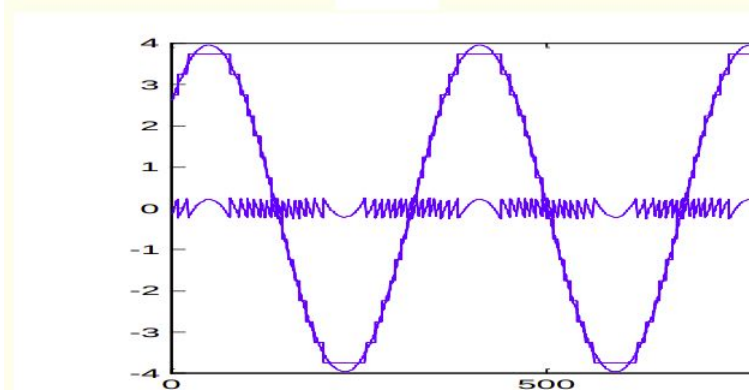
Digitize y with 8 discrete steps.

-3.5, -2.5, -1.5, -0.5, 3.5, 2.5, 1.5, 0.5

$-0.5 < \text{error} < 0.5$

3.99 or 3.01 \rightarrow 3.5

2.99 \rightarrow 2.5



Digitize y with 16 discrete steps.

-3.75, -3.25, -2.75, -2.25, -1.75, -1.25,
-0.75, -0.25, 0.25, 0.75, 1.25, 1.75
2.25, 2.75, 3.25, 3.75

$-0.25 < \text{error} < 0.25$


3.99 or 3.5 \rightarrow 3.75

二進位表示法

- 二進位表示法可以表示為: $a_{n-1} a_{n-2} \dots a_1 a_0 . a_{-1} a_{-2} \dots a_{-m+1} a_{-m}$ $a_i \in \{0, 1\}$

$$a_{n-1} \times 2^{n-1} + a_{n-2} \times 2^{n-2} + \dots + a_1 \times 2^1 + a_0 \times 2^0 \\ + a_{-1} \times 2^{-1} + a_{-2} \times 2^{-2} + \dots + a_{-m} \times 2^{-m}$$

$$(10110.101)_2 = 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^{-1} + 1 \times 2^{-3} \\ = (16 + 4 + 2 + 0.5 + 0.125) = (22.625)_{10}$$

 1 bit

同時我們稱在二進位中用來表示一個 0 或 1 的單位稱為一個 bit (通常單位簡寫為 b)

二進位表示法

- 8 個 bit 的單位稱為一個 byte (縮寫為 B)
- 在以10進位的現實生活中, $K = 10^3$, $M = 10^6$, $G = 10^9$
- 在以2進為的電腦中
 - 2^{10} (1024) 是 Kilo, 縮寫成 “K”
 - 2^{20} (1,048,576) 是 Mega, 縮寫成 “M”
 - 2^{30} (1,073, 741,824) 是 Giga, 縮寫成 “G”
 - 2^{40} 是 Tera, 縮寫成 “T”

因此在看到電腦記憶體中規格中的 $8GB = 8 * 2^{30} \text{ bytes} = 8 * 2^{30} * 8 \text{ bits}$

變數 (variable) 與常數 (constant)

- 當我們在程式中宣告一個變數，程式在執行時便會根據 type 保留一塊記憶體空間給變數，不管變數裡面的值如何改變，變數所站的記憶體空間都是一樣
- 常數顧名思義，它的值並不會像變數一樣會改變

int **a** = 123;

variable constant

基本資料型態

C 常見的內建資料型態

可以使用 `sizeof(...)` 來查詢各個 type 在系統內佔多少空間

資料型態	名稱	大小 (bytes)	範例
短整數 (Short Integer)	short int	2	32
整數 (Integer)	int	4	32
長整數 (Long Integer)	long int	4	32
字元 (Character)	char	1	'3'
單精度浮點數 (Single Precision Floating Point)	float	4	3.2
雙精度浮點數 (Double Precision Floating Point)	double	8	3.2
無	void	(無)	(無)

可表示範圍

-32768 ~ 32767

-2147483648 ~ 2147483647

-2147483648 ~ 2147483647

0~255

1.2e-38 ~ 3.4e38

2.2e-308~1.8e308

《實作相依》：意指語言標準內容並沒有強制的規定，在使用不同編譯器或設定的情況下，可能會不一樣

大小是《實作相依》

整數型別 (integer type)

- 整數型別大小可分為 short, int, long, long long (需根據作業系統以及編譯器來判斷實際大小), 但 $\text{short} \leq \text{int} \leq \text{long} \leq \text{long long}$
- 整數型別又可以依據有沒有負號分為: signed(有號) integer 跟 unsigned(無號) integer
- 因此 signed integer 的範圍計算:
 - 大小為 n bits 則範圍為 $-2^{n-1} \sim 2^{n-1} - 1$
- 而 unsigned integer 的範圍計算:
 - 大小為 n bits 則範圍為 $0 \sim 2^n - 1$

Overflow 溢位

- 由於用來紀錄變數值的 bit 個數有限(不管是 integer、floating number、char), 所以變數值都會有能夠儲存值的範圍上下限
- 若超過該上下限, 該變數儲存的值就會因為儲存空間不足而不可預期
- 這種情況就稱作 Overflow 溢位
- 阿如果連 unsigned long long 都滿足不了要怎麼辦? Answer: 可以使用大數運算 (之後會介紹到)

字元型態 char

- 一個 char 型態佔有一個 byte (8 個 bits) 的空間, 可以用來儲存字元
- 字元符號會被編碼成 1 個 byte 可以儲存的格式, 最常見的編碼格式是 ASCII, C 語言裡面的字元操作也是根據 ASCII
- 實際上 ASCII 只使用 7 個 bit (因為ASCII制定的時間早於大家把 8bit 當作一個 byte)

<Notice!>

- 字元常數必須要放在單引號 ('), 雙引號是用來表示字串 (string) 講到array 會細談

ASCII Table

Control Characters				Graphic Symbols											
Name	Dec	Binary	Hex	Symbol	Dec	Binary	Hex	Symbol	Dec	Binary	Hex	Symbol	Dec	Binary	Hex
NUL	0	0000000	00	space	32	0100000	20	@	64	1000000	40	'	96	1100000	60
SOH	1	0000001	01	!	33	0100001	21	A	65	1000001	41	a	97	1100001	61
STX	2	0000010	02	"	34	0100010	22	B	66	1000010	42	b	98	1100010	62
ETX	3	0000011	03	#	35	0100011	23	C	67	1000011	43	c	99	1100011	63
EOT	4	0000100	04	\$	36	0100100	24	D	68	1000100	44	d	100	1100100	64
ENQ	5	0000101	05	%	37	0100101	25	E	69	1000101	45	e	101	1100101	65
ACK	6	0000110	06	&	38	0100110	26	F	70	1000110	46	f	102	1100110	66
BEL	7	0000111	07	'	39	0100111	27	G	71	1000111	47	g	103	1100111	67
BS	8	0001000	08	(40	0101000	28	H	72	1001000	48	h	104	1101000	68
HT	9	0001001	09)	41	0101001	29	I	73	1001001	49	i	105	1101001	69
LF	10	0001010	0A	*	42	0101010	2A	J	74	1001010	4A	j	106	1101010	6A
VT	11	0001011	0B	+	43	0101011	2B	K	75	1001011	4B	k	107	1101011	6B
FF	12	0001100	0C	,	44	0101100	2C	L	76	1001100	4C	l	108	1101100	6C
CR	13	0001101	0D	-	45	0101101	2D	M	77	1001101	4D	m	109	1101101	6D
SO	14	0001110	0E	.	46	0101110	2E	N	78	1001110	4E	n	110	1101110	6E
SI	15	0001111	0F	/	47	0101111	2F	O	79	1001111	4F	o	111	1101111	6F
DLE	16	0010000	10	0	48	0110000	30	P	80	1010000	50	p	112	1110000	70
DC1	17	0010001	11	1	49	0110001	31	Q	81	1010001	51	q	113	1110001	71
DC2	18	0010010	12	2	50	0110010	32	R	82	1010010	52	r	114	1110010	72
DC3	19	0010011	13	3	51	0110011	33	S	83	1010011	53	s	115	1110011	73
DC4	20	0010100	14	4	52	0110100	34	T	84	1010100	54	t	116	1110100	74
NAK	21	0010101	15	5	53	0110101	35	U	85	1010101	55	u	117	1110101	75
SYN	22	0010110	16	6	54	0110110	36	V	86	1010110	56	v	118	1110110	76
ETB	23	0010111	17	7	55	0110111	37	W	87	1010111	57	w	119	1110111	77
CAN	24	0011000	18	8	56	0111000	38	X	88	1011000	58	x	120	1111000	78
EM	25	0011001	19	9	57	0111001	39	Y	89	1011001	59	y	121	1111001	79
SUB	26	0011010	1A	:	58	0111010	3A	Z	90	1011010	5A	z	122	1111010	7A
ESC	27	0011011	1B	;	59	0111011	3B	[91	1011011	5B	{	123	1111011	7B
FS	28	0011100	1C	<	60	0111100	3C	\	92	1011100	5C		124	1111100	7C
GS	29	0011101	1D	=	61	0111101	3D]	93	1011101	5D	}	125	1111101	7D
RS	30	0011110	1E	>	62	0111110	3E	^	94	1011110	5E	~	126	1111110	7E
US	31	0011111	1F	?	63	0111111	3F	_	95	1011111	5F	Del	127	1111111	7F

ASCII 的小細節

- 若想顯示ASCII所代表的字元(如果該字元可顯示的話), 在 printf 中的格式碼要放 %c, e.g. printf(“%c”, ‘a’)
- ASCII 中的數字字元並不等於實際的數字
- 可以透過對 char 進行算術運算來達到想要的操作, e.g. 小寫換大寫、凱薩加密法.....等等

跳脫字元與跳脫序列

- 當我們想輸入一些無法用鍵盤輸入的字元 (BEL)、或想輸入的字元會跟C語言原本的語法有衝突(e.g.在 printf 中輸入雙引號)
- 在C語言裡可以使用反斜符號『\』並加上一個控制碼來輸入這些特殊的字元
- 『\』讓後面的控制碼跳脫原本字元的定義, 所以稱作跳脫字元
- 『\』與後面的控制碼合稱跳脫序列

跳脫字元與跳脫序列

Escape sequence ↕	Hex value in ASCII ↕	Character represented
\a	07	Alert (Beep, Bell) (added in C89) ^[1]
\b	08	Backspace
\e ^{note 1}	1B	Escape character
\f	0C	Formfeed Page Break
\n	0A	Newline (Line Feed); see notes below
\r	0D	Carriage Return
\t	09	Horizontal Tab
\v	0B	Vertical Tab
\\	5C	Backslash
\'	27	Apostrophe or single quotation mark
\"	22	Double quotation mark
\?	3F	Question mark (used to avoid trigraphs)

浮點數 floating number

- 若程式中想儲存帶有小數點的數字，可以用浮點數來代表
- 其中 C 語言主要有兩種分別為: float、double
- float 所使用的空間是 4 bytes 範圍: $1.2e-38 \sim 3.4e38$
- double 所使用的空間是 8 bytes 範圍: $2.2e-308 \sim 1.8e308$
- 在 printf 裡使用 %f 格式碼輸出 float 變數, %lf 格式法輸出 double 變數

常數 constant

- C 語言裡常數可以用許多表示法表示
- 整數常數
 - 預設就是十進位：15, 255, 32767
 - 可以在開頭加個0變成八進制表示法：017, 0377, 077777
 - 可以在開頭加0x變成十六進位表示法：0xf, 0xff, 0x7fff (f大小寫皆可)
 - 可以在常數後方加 L (大小寫皆可) 告編譯器此常數為 long (預設整數常數為 int)
 - 可以在常數後方加 LL (大小寫皆可, 要一致) 告編譯器此常數為 long long
 - 可以在常數後方加 U (大小寫皆可) 告編譯器此常數為 unsigned (預設整數常數為 signed)
 - e.g., 15L, 0xffffffffUL, 0377U

常數 constant

- 浮點數常數
 - 預設就是使用 double 型別, 57.0
 - 可以在尾巴加個 F (大小寫皆可) 告編譯器此常數為 float, 57.0F
 - 可以在尾巴加個 L (大小寫皆可) 告編譯器此常數為 long double, 57.0L
 - 可以在中間加 e 變成指數表示法, 其中 e 代表 10 的某次方: 0.57e2, 570.0e-1

型別轉換 type casting

- 若我們在程式中想轉換型別，例如 int 換成 double，我們可以使用 type casting 來達成
- 語法： (欲轉換成的資料型態) 變數名稱;
- 要注意的是**有時候型別轉換會改變變數原有的值**，如浮點數轉換成整數時會捨棄小數點，留下整數部份
- 或者兩整數相除想得到小數點的值，可先加兩整數先換成浮點數，這樣兩數相除即可得到小數點的值