CH1 Fundamentals

1. scanf("%1d", &num); 一次只讀一個數字

2. printf("%.15f", a*b*c*d*e); //印小數點後15位

CH2 Control Structures

continue: 直接到迴圈最下面 goto: 後面接標籤(類似 case)

CH3 DataTypes

型態	Byte數	範圍(值域)	
bool	1	true - false	http://notepad.yehyeh.net
char	1	-128 ~ 127	
signed char	1	-128 ~ 127	
unsigned char	1	0 ~ 255	
wchar_t	2	0 ~ 65,535	
short (= short int - signed short int)	2	-32,768 ~ 32,767	
unsigned short (=unsigned shot int)	2	0 ~ 65,535	
int (= signed - signed int)	4	-2,147,483,648 ~ 2,147,483,64	47
unsigned int (= unsigned)	4	0 ~ 4,294,967,295	
long (=long int - signed long int)	4	-2,147,483,648 ~ 2,147,483,64	47
unsigned long (= unsigned long int)	4	0 ~ 4,294,967,295	
float	4	±3.4 * 10 ^{±38} (32位有效位	數,精確至小數點後7位)
double	8	±1.7 * 10 ±308(48位有效位	立數,精確至小數點後15位)
long double	8	±1.7 * 10 ^{±308} (至少48位列	有效位數,精確至小數點後19位)

long long int 8Btyes --[-9,223,372,036,854,775,807, +9,223,372,036,854,775,807] unsigned long long 8bytes --[0, 18,446,744,073,709,551,615]

Integer

1. 強制指定成某 datatype, 在尾巴加東西

ex. 15L, 0x7fffffffULL, 0.53f(整數默認int, 浮點數默認是雙精度)

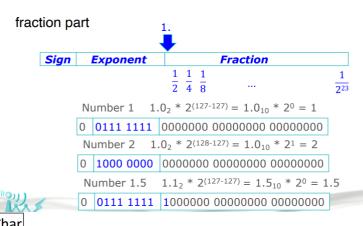
2. Reading and Writing

%d, %lld(long long), %hd(short int), %f or %e or %g(float and double), 但 scanf 時%lf or %le or %lg(double 要加 l) for unsigned int: %u(unsigned decimal), %o(unsigned oct), %x(unsigned hex)

for (unsigned) short int: %hu, %hx for (unsigned) char: %hhx, %hhd

Floating Num

3. IEEE754 floating-point standard sign(1 bit), exponent(8bits), fraction(23bits)



Name	Char	Oct	Hex	Dec
Alert (bell)	\a	\7	\x07	7
Backspace	\b	\10	\x08	8
Form feed	\f	\14	\x0c	12
New line	\n	\12	\x0a	10
Carriage return	\r	\15	\x0d	13
Horizontal tab	\t	\11	\x09	9
Vertical tab	\v	\13	\x0b	11
Backslash	11	\134	\x27	92
, Question mark	/3	\77	\x22	63
Single quote	1	\47	\x5c	39
Double quote	/ _n	\42	\x3f	34

4. ASCII、跳脫字元

(10)\n, (32)space,

(37)%, (42)*, (43)+, (45)-, (47)/,

(48~57)0~9,

(65~90)A~Z,

(97~122)a~z

hex	oct	char	dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char
0	000	NULL	32	20	040	space	64	40	100	@	96	60	140	*
1	001	SOH	33	21	041	1	65	41	101	Α	97	61	141	a
2	002	STX	34	22	042		66	42	102	В	98	62	142	b
3	003	ETX	35	23	043	#	67	43	103	C	99	63	143	C
4	004	EOT	36	24	044	\$	68	44	104	D	100	64	144	d
5	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	007	BEL	39	27	047	1.	71	47	107	G	103	67	147	g
8	010	BS	40	28	050	(72	48	110	н	104	68	150	h
9	011	TAB	41	29	051)	73	49	111	1	105	69	151	i
а	012	LF	42	2a	052	*	74	4a	112	J	106	6a	152	j
b	013	VT	43	2b	053	+	75	4b	113	K	107	6b	153	k
С	014	FF	44	2c	054	,	76	4c	114	L	108	6c	154	1
d	015	CR	45	2d	055	2	77	4d	115	M	109	6d	155	m
е	016	SO	46	2e	056		78	4e	116	N	110	6e	156	n
f	017	SI	47	2f	057	/	79	4f	117	0	111	6f	157	0
10	020	DLE	48	30	060	0	80	50	120	P	112	70	160	р
11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	S
14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
19	031	EM	57	39	071	9	89	59	131	Y	121	79	171	У
1a	032	SUB	58	3a	072	:	90	5a	132	Z	122	7a	172	z
1b	033	ESC	59	3b	073	;	91	5b	133	Į.	123	7b	173	{
1c	034	FS	60	3c	074	<	92	5c	134	١	124	7c	174	1
1d	035	GS	61	3d	075	=	93	5d	135]	125	7d	175	}
1e	036	RS	62	3e	076	>	94	5e	136	۸	126	7e	176	~
1f	037	US	63	3f	077	?	95	5f	137		127	7f	177	DEL
	0 1 2 3 4 5 6 7 8 9 a b c d e f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e	0 000 1 001 2 002 3 003 4 004 5 005 6 006 7 007 8 010 9 011 a 012 b 013 c 014 d 015 e 016 f 017 10 020 11 021 12 022 13 023 14 024 15 025 16 026 17 027 18 030 19 031 1a 032 1b 033 1c 034 1d 035 1e 036	0 000 NULL 1 001 SOH 2 002 STX 3 003 ETX 4 004 EOT 5 005 ENQ 6 006 ACK 7 007 BEL 8 010 BS 9 011 TAB a 012 LF b 013 VT c 014 FF d 015 CR e 016 SO f 017 SI 10 020 DLE 11 021 DC1 12 022 DC2 13 023 DC3 14 024 DC4 15 025 NAK 16 026 SYN 17 027 ETB 18 030 CAN 19 031 EM 1a 032 SUB 1b 033 ESC 1c 034 FS 1d 035 GS 1e 036 RS	0 000 NULL 32 1 001 SOH 33 2 002 STX 34 3 003 ETX 35 4 004 EOT 36 5 005 ENQ 37 6 006 ACK 38 7 007 BEL 39 8 010 BS 40 9 011 TAB 41 a 012 LF 42 b 013 VT 43 c 014 FF 44 d 015 CR 45 e 016 SO 46 f 017 SI 47 10 020 DLE 48 11 021 DC1 49 12 022 DC2 50 13 023 DC3 51 14 024 </td <td>0 000 NULL 32 20 1 001 SOH 33 21 2 002 STX 34 22 3 003 ETX 35 23 4 004 EOT 36 24 5 005 ENQ 37 25 6 006 ACK 38 26 7 007 BEL 39 27 8 010 BS 40 28 9 011 TAB 41 29 a 012 LF 42 2a b 013 VT 43 2b c 014 FF 44 2c d 015 CR 45 2d e 016 SO 46 2e f 017 SI 47 2f 10 020 DLE 48 30</td> <td>0 000 NULL 32 20 040 1 001 SOH 33 21 041 2 002 STX 34 22 042 3 003 ETX 35 23 043 4 004 EOT 36 24 044 5 005 ENQ 37 25 045 6 006 ACK 38 26 046 7 007 BEL 39 27 047 8 010 BS 40 28 050 9 011 TAB 41 29 051 a 012 LF 42 2a 052 b 013 VT 43 2b 053 c 014 FF 44 2c 054 d 015 CR 45 2d 055 e 016 SO</td> <td>0 000 NULL 32 20 040 space 1 001 SOH 33 21 041 ! 2 002 STX 34 22 042 " 3 003 ETX 35 23 043 # 4 004 EOT 36 24 044 \$ 5 005 ENQ 37 25 045 % 6 006 ACK 38 26 046 & 7 007 BEL 39 27 047 ' 8 010 BS 40 28 050 (9 011 TAB 41 29 051) a 012 LF 42 2a 052 * b 013 VT 43 2b 053 + c 014 FF 44 2c 0</td> <td>0 000 NULL 32 20 040 space 64 1 001 SOH 33 21 041 ! 65 2 002 STX 34 22 042 " 66 3 003 ETX 35 23 043 # 67 4 004 EOT 36 24 044 \$ 68 5 005 ENQ 37 25 045 % 69 6 006 ACK 38 26 046 & 70 7 007 BEL 39 27 047 ' 71 8 010 BS 40 28 050 (72 9 011 TAB 41 29 051) 73 a 012 LF 42 2a 052 * 74 b 013</td> <td>0 000 NULL 32 20 040 space 64 40 1 001 SOH 33 21 041 1 65 41 2 002 STX 34 22 042 " 66 42 3 003 ETX 35 23 043 # 67 43 4 004 EOT 36 24 044 \$ 68 44 5 005 ENQ 37 25 045 % 69 45 6 006 ACK 38 26 046 & 70 46 7 007 BEL 39 27 047 ' 71 47 8 010 BS 40 28 050 (72 48 9 011 TAB 41 29 051) 73 49 a 012 LF 42 2a 052 * 74 4a b 013 VT 43 2b 053 + 75 4b c 014 FF 44 2c 054 , 76 4c d 015 CR 45 2d 055 - 77 4d e 016 SO 46 2e 056 . 78 4e f 017 SI 47 2f 057 / 79 4f 10 020 DLE 48 30 060 0 80 50 11 021 DC1 49 31 061 1 81 51 12 022 DC2 50 32 062 2 82 52 13 023 DC3 51 33 063 3 83 53 14 024 DC4 52 34 064 4 84 54 15 025 NAK 53 35 065 5 85 55 16 026 SYN 54 36 066 6 86 56 17 027 ETB 55 37 067 7 87 57 18 030 CAN 56 38 070 8 88 58 19 031 EM 57 39 071 9 89 59 1a 032 SUB 58 3a 072 : 90 5a 1b 033 ESC 59 3b 073 ; 91 5b 1c 034 FS 60 3c 074 < 92 5c 1d 035 GS 61 3d 075 = 93 5d 1e 036 RS 62 3e 076 > 94 5e</td> <td>0 000 NULL 32 20 040 space 64 40 100 1 001 SOH 33 21 041 ! 65 41 101 2 002 STX 34 22 042 " 66 42 102 3 003 ETX 35 23 043 # 67 43 103 4 004 EOT 36 24 044 \$ 68 44 104 5 005 ENQ 37 25 045 % 69 45 105 6 006 ACK 38 26 046 & 70 46 106 7 007 BEL 39 27 047 ' 71 47 107 8 010 BS 40 28 050 (72 48 110 9 011</td> <td>0 000 NULL 32 20 040 space 64 40 100 @ 1 001 SOH 33 21 041 ! 65 41 101 A 2 002 STX 34 22 042 " 66 42 102 B 3 003 ETX 35 23 043 # 67 43 103 C 4 004 EOT 36 24 044 \$ 68 44 104 D 5 005 ENQ 37 25 045 % 69 45 105 E 6 006 ACK 38 26 046 & 70 46 106 F 7 007 BEL 39 27 047 ' 71 47 107 G 8 010 BS 40 28 050 (72 48 110 H 9 011 TAB 41 29 051) 73 49 111 I a 012 LF 42 2a 052 * 74 4a 112 J b 013 VT 43 2b 053 + 75 4b 113 K c 014 FF 44 2c 054 , 76 4c 114 L d 015 CR 45 2d 055 - 77 4d 115 M e 016 SO 46 2e 056 . 78 4e 116 N f 017 SI 47 2f 057 / 79 4f 117 O 10 020 DLE 48 30 060 0 80 50 120 P 11 021 DC1 49 31 061 1 81 51 121 Q 12 022 DC2 50 32 062 2 82 52 122 R 13 023 DC3 51 33 063 3 83 53 123 S 14 024 DC4 52 34 066 6 6 86 56 126 V 17 027 ETB 55 37 067 7 87 87 57 127 W 18 033 ESC 59 3b 073 ; 91 5b 133 [1c 034 FS 60 3c 074 < 92 5c 134 \ 1d 035 GS 61 3d 075 = 93 5d 135] 1e 036 RS 62 8e 076 > 94 5e 136 ^</td> <td>0 000 NULL 32 20 040 space 64 40 100</td> <td>0 000 NULL 32 20 040 space 64 40 100 @ 96 60 1 1 001 SOH 33 21 041 I 65 41 101 A 97 61 2 002 STX 34 22 042 " 66 42 102 B 98 62 3 003 ETX 35 23 043 # 67 43 103 C 99 63 4 004 EOT 36 24 044 \$ 68 44 104 D 100 64 5 005 ENQ 37 25 045 % 69 45 105 E 101 65 6 006 ACK 38 26 046 & 70 46 106 F 102 66 7 007 BEL 39 27 047 ' 71 47 107 G 103 67 8 011 TAB 41 29 051) 73 49 111 I 105 69 a 012 LF 42 2a 052 * 74 4a 112 J 106 6a b 013 VT 43 2b 053 + 75 4b 113 K 107 6b c 014 FF 44 2c 054 , 76 4c 114 L 108 6c d 015 CR 45 2d 055 - 77 4d 115 M 109 6d e 016 SO 46 2e 056 . 78 4e 116 N 110 6e f 017 SI 48 30 060 0 88 50 120 P 111 OF 111 6f 10 020 DLE 48 30 060 0 88 50 120 P 112 70 111 6f 10 020 DLE 48 30 060 0 88 55 122 R 114 72 127 13 147 127 147 157 157 158 159 159 159 159 159 159 159 159 159 159</td> <td>0 000 NULL 32 20 040 space 64 40 100</td>	0 000 NULL 32 20 1 001 SOH 33 21 2 002 STX 34 22 3 003 ETX 35 23 4 004 EOT 36 24 5 005 ENQ 37 25 6 006 ACK 38 26 7 007 BEL 39 27 8 010 BS 40 28 9 011 TAB 41 29 a 012 LF 42 2a b 013 VT 43 2b c 014 FF 44 2c d 015 CR 45 2d e 016 SO 46 2e f 017 SI 47 2f 10 020 DLE 48 30	0 000 NULL 32 20 040 1 001 SOH 33 21 041 2 002 STX 34 22 042 3 003 ETX 35 23 043 4 004 EOT 36 24 044 5 005 ENQ 37 25 045 6 006 ACK 38 26 046 7 007 BEL 39 27 047 8 010 BS 40 28 050 9 011 TAB 41 29 051 a 012 LF 42 2a 052 b 013 VT 43 2b 053 c 014 FF 44 2c 054 d 015 CR 45 2d 055 e 016 SO	0 000 NULL 32 20 040 space 1 001 SOH 33 21 041 ! 2 002 STX 34 22 042 " 3 003 ETX 35 23 043 # 4 004 EOT 36 24 044 \$ 5 005 ENQ 37 25 045 % 6 006 ACK 38 26 046 & 7 007 BEL 39 27 047 ' 8 010 BS 40 28 050 (9 011 TAB 41 29 051) a 012 LF 42 2a 052 * b 013 VT 43 2b 053 + c 014 FF 44 2c 0	0 000 NULL 32 20 040 space 64 1 001 SOH 33 21 041 ! 65 2 002 STX 34 22 042 " 66 3 003 ETX 35 23 043 # 67 4 004 EOT 36 24 044 \$ 68 5 005 ENQ 37 25 045 % 69 6 006 ACK 38 26 046 & 70 7 007 BEL 39 27 047 ' 71 8 010 BS 40 28 050 (72 9 011 TAB 41 29 051) 73 a 012 LF 42 2a 052 * 74 b 013	0 000 NULL 32 20 040 space 64 40 1 001 SOH 33 21 041 1 65 41 2 002 STX 34 22 042 " 66 42 3 003 ETX 35 23 043 # 67 43 4 004 EOT 36 24 044 \$ 68 44 5 005 ENQ 37 25 045 % 69 45 6 006 ACK 38 26 046 & 70 46 7 007 BEL 39 27 047 ' 71 47 8 010 BS 40 28 050 (72 48 9 011 TAB 41 29 051) 73 49 a 012 LF 42 2a 052 * 74 4a b 013 VT 43 2b 053 + 75 4b c 014 FF 44 2c 054 , 76 4c d 015 CR 45 2d 055 - 77 4d e 016 SO 46 2e 056 . 78 4e f 017 SI 47 2f 057 / 79 4f 10 020 DLE 48 30 060 0 80 50 11 021 DC1 49 31 061 1 81 51 12 022 DC2 50 32 062 2 82 52 13 023 DC3 51 33 063 3 83 53 14 024 DC4 52 34 064 4 84 54 15 025 NAK 53 35 065 5 85 55 16 026 SYN 54 36 066 6 86 56 17 027 ETB 55 37 067 7 87 57 18 030 CAN 56 38 070 8 88 58 19 031 EM 57 39 071 9 89 59 1a 032 SUB 58 3a 072 : 90 5a 1b 033 ESC 59 3b 073 ; 91 5b 1c 034 FS 60 3c 074 < 92 5c 1d 035 GS 61 3d 075 = 93 5d 1e 036 RS 62 3e 076 > 94 5e	0 000 NULL 32 20 040 space 64 40 100 1 001 SOH 33 21 041 ! 65 41 101 2 002 STX 34 22 042 " 66 42 102 3 003 ETX 35 23 043 # 67 43 103 4 004 EOT 36 24 044 \$ 68 44 104 5 005 ENQ 37 25 045 % 69 45 105 6 006 ACK 38 26 046 & 70 46 106 7 007 BEL 39 27 047 ' 71 47 107 8 010 BS 40 28 050 (72 48 110 9 011	0 000 NULL 32 20 040 space 64 40 100 @ 1 001 SOH 33 21 041 ! 65 41 101 A 2 002 STX 34 22 042 " 66 42 102 B 3 003 ETX 35 23 043 # 67 43 103 C 4 004 EOT 36 24 044 \$ 68 44 104 D 5 005 ENQ 37 25 045 % 69 45 105 E 6 006 ACK 38 26 046 & 70 46 106 F 7 007 BEL 39 27 047 ' 71 47 107 G 8 010 BS 40 28 050 (72 48 110 H 9 011 TAB 41 29 051) 73 49 111 I a 012 LF 42 2a 052 * 74 4a 112 J b 013 VT 43 2b 053 + 75 4b 113 K c 014 FF 44 2c 054 , 76 4c 114 L d 015 CR 45 2d 055 - 77 4d 115 M e 016 SO 46 2e 056 . 78 4e 116 N f 017 SI 47 2f 057 / 79 4f 117 O 10 020 DLE 48 30 060 0 80 50 120 P 11 021 DC1 49 31 061 1 81 51 121 Q 12 022 DC2 50 32 062 2 82 52 122 R 13 023 DC3 51 33 063 3 83 53 123 S 14 024 DC4 52 34 066 6 6 86 56 126 V 17 027 ETB 55 37 067 7 87 87 57 127 W 18 033 ESC 59 3b 073 ; 91 5b 133 [1c 034 FS 60 3c 074 < 92 5c 134 \ 1d 035 GS 61 3d 075 = 93 5d 135] 1e 036 RS 62 8e 076 > 94 5e 136 ^	0 000 NULL 32 20 040 space 64 40 100	0 000 NULL 32 20 040 space 64 40 100 @ 96 60 1 1 001 SOH 33 21 041 I 65 41 101 A 97 61 2 002 STX 34 22 042 " 66 42 102 B 98 62 3 003 ETX 35 23 043 # 67 43 103 C 99 63 4 004 EOT 36 24 044 \$ 68 44 104 D 100 64 5 005 ENQ 37 25 045 % 69 45 105 E 101 65 6 006 ACK 38 26 046 & 70 46 106 F 102 66 7 007 BEL 39 27 047 ' 71 47 107 G 103 67 8 011 TAB 41 29 051) 73 49 111 I 105 69 a 012 LF 42 2a 052 * 74 4a 112 J 106 6a b 013 VT 43 2b 053 + 75 4b 113 K 107 6b c 014 FF 44 2c 054 , 76 4c 114 L 108 6c d 015 CR 45 2d 055 - 77 4d 115 M 109 6d e 016 SO 46 2e 056 . 78 4e 116 N 110 6e f 017 SI 48 30 060 0 88 50 120 P 111 OF 111 6f 10 020 DLE 48 30 060 0 88 50 120 P 112 70 111 6f 10 020 DLE 48 30 060 0 88 55 122 R 114 72 127 13 147 127 147 157 157 158 159 159 159 159 159 159 159 159 159 159	0 000 NULL 32 20 040 space 64 40 100

5. <ctype.h> 的 toupper(ch)

6. 讀取字元(scanf, getchar 不會跳過空白)

(1)scanf("%c", &ch); 前面加空白,可跳過

型別轉換

- 1. int 轉 float(1bit + 23 fraction bits)可能會有精度失誤(當 int 的使用的 bit 大於 float 限制)
- 2. casting(強制轉換) 是一種 unary operator

Bitwise Operation

```
1.若都是正整數,使用unsigned type
2.operators: &, |, ~(not), <<, >>, ^(xor)
3.Setting a bit by 目標處 or 1
i = 0x000 = 00000000000
i = 0x010 = 00000010000
 i | j =
           00000010000
unsigned long long temp1 = (0x0);
    unsigned long long temp2 = temp1 | (1<<4);</pre>
    num = num | temp2;
00000111(7) \rightarrow 00010111(5^{th} bit turned to 1)
4.Clearing a bit by 目標處 and 0
 i = 0x1111
j=0x0010(想要把第二個 16 位元-bit 都換成 0)
 i \& (\sim j) = 0x1101
unsigned long long temp1 = \sim(0x0);
    unsigned long long temp2 = temp1 & ~(1<<1);
    num = num & temp2;
00000111(7) \rightarrow 00000101(2^{nd} \text{ bit turned to } 0)
5. Testing a bit
 I & (1 << j) testing bit j
int num = 7;
    int test;
    test = (num & (1<<i))>>i;
00000111(7)→測試7的(i+1)th bit是 1 or 0
6. 印出 8bit 2進制
for(int z = 7; z >= 0; z -- ){
        printf("%d", (num>>z)&1);
7. 注意: 常數默認是int(32 bits)、
 CH5 Array
 1. int a[15] = \{[14] = 48, [9] = 7, [2] = 29\};
 2. array length = sizeof(array) / sizeof(a[0])
```

CH6 Function and Scope

- 1. void 也可以 return
- 2. %p 印指標

CH7 Pointer & Array

- 1. 函式回傳時不可將內部pointer傳出去,否則runtime error(需另外設global pointer)
- 2. p=&a[5]; q=&a[1]; i = p q; l = 4
- 3. 優先級

Expression	Meaning
*p++ or * (p++)	Value of expression is *p before increment;
	increment p later
(*p)++	Value of expression is *p before increment;
	increment *p <mark>later</mark>
*++p or * (++p)	Increment p first;
	value of expression is *p after increment
++*p or ++ (*p)	Increment *p first;
	value of expression is *p after increment

4.

- (1) 一維: a[i] == *(a+i) , 但不能變更a的值,可以p = a; p += 1;
- (2) 二維:

- 5. 傳陣列進Function
- 一維: void store(int a[], int n){} or 改成 int* a
- 二維: void store(int a[][NUM_COLS], int n){} or 改成 int** a

CH8 String(char array - char*)

- 1.不能直接用=修改string,要用string函式庫
- 2. char ch; ch = "abc"[1]; //ch = 'b'
- 3. 宣告一個過長的char[],如果字元沒填滿,字串後面都用'\0'寒滿
- 4. I/O
- (1) printf("%m.ps", str); //印出str前p個字元。m: 創造field,置右; m前加負號
- (2) puts(str); //自動在結尾加\n
- (3) scanf("%s", str); //string是指標不用加&, 會跳過space scanf("%ns", str); //避免讀超過char[]大小
- (4) gets(str); //不會跳過space, 直到\n(把\n换成\0儲存結果)
- 5. #include <string.h>
- (1) strncpy(str1, str2, n); 限制複製多少字元//可用於清空字串
- (2) len = strlen("abc"); //len==3
- (3) strncat(str1, str2, n);
- (4) strcmp(str1, str2); //str1>str2: 回傳正值; str1=str2:回傳0; str1<str2:回傳負值(by ASCII)

```
char string[] = "a string, of ,, tokens";
char *token;
int main(){
               char string[] = "a string, of ,, tokens";
              char* token = strtok(string," ,"); /*There are two delimiters here*/
              while (token != NULL){
                              printf("The token is: %s\n", token);
                             token = strtok(NULL, " , ");
               }
   輸出結果:
   The token is: a
   The token is: string
   The token is: of
   The token is: tokens
      |a| |s|t|r|i|n|g|,|o|f| |,|,|t|o|k|e|n|s|
      This is the original string before the first call to strtok().
      |a| \ 0| s| t| r| i| n| g|, |o| f| |, |, |t| o| k| e| n| s|
        ^---- Token will point here on the first call.
      |a| \ |c| 
                          ^---- Token will point here on the second call.
      |a| \ 0| \ |t| \ |r| \ |i| \ |g| \ 0| \ |f| \ 0|, \ |, \ |t| \ |o| \ |k| \ |e| \ |n| \ |s|
                                                                                              ^---- Token will point here on
                                                                                                                the third call.
                                                                                                (and so on)
   (6)字串是否含有字元
```

char* search = strchr(source, char); 回傳char在s中第一次出現位置的指標

(7)字串是否含有子字串

```
char* source = "a string, of ,, tokens";
    char* search = "tokens";
    char* result = strstr(source, search);
    if(result != NULL)
       printf("Found!");
        printf("Not found!");
```

以下函數作為字串的搜尋處理之用

(5) strtok:

函數名稱	功能	函數原型
strchr	回傳在字串 s 中,字元 c 第一次出現位置的指標	<pre>char *strchr(const char *s, int c);</pre>
strcspn	計算經過幾個字元會在字串 s1 中遇到屬於 s2 中的字元	<pre>size_t strcspn(const char *s1, const char *s2);</pre>
strspn	計算經過幾個字元會在字串 s1 中遇到不屬於 s2 中的字元	<pre>size_t strspn(const char *s1, const char *s2);</pre>
strpbrk	回傳在字串 s2 中的任何字元在 s1 第一次出現位置的指標	<pre>char *strpbrk(const char *s1, const char *s2);</pre>
strrchr	回傳在字串 s 中,字元 c 最後一次出現位置的指標	<pre>char *strrchr(const char *s, int c);</pre>
strstr	回傳在字串 s2 在 s1 第一次出現位置的指標	<pre>char *strstr(const char *s1, const char *s2);</pre>
strtok	以字串 s2 的內容切割 s1	<pre>char *strtok(char *s1, const char *s2);</pre>

6. string array

```
char *planets[] = {"Mercury", "Venus", "Earth",
                     "Mars", "Jupiter", "Saturn",
                     "Uranus", "Neptune", "Pluto"};
7. char*相關runtime error,試試看malloc記憶題給它
CH9 Structure & Union & Enum
1. Structure中的Array可以直接複製到另一個Structure
2. 沒有寫typedef的話,structure型別名稱前要加struct
3. typedef struct (struct type Name){···};
4. 給初始值 struct Point point1 = {.x = 10, .y = 18};
5. enum colors{BLACK, LT GRAY = 7, DK GRAY, WHITE = 15};
從0開始,其他依序: BLACk = 0, DK GRAY = 8
CH10 Dynamic
1. p = (char^*)malloc(n+1); //for string's '\0'
strcpy(p, "abc");
p = strcat("abc", "def");
2. allocate memory for array
int*p;
p = malloc(n*sizeof(int));
```

- 3. calloc function: create an array \cdot initalize the merroy by setting all bits = 0 a = calloc(n, sizeof(int));
- 4. realloc function: resize a memory block call by malloc, calloc, realloc q = realloc(q, 20000) //會修改地址,要重新指一次
- 5. free(p); //記得釋放記憶體,釋放後不可再附值給p, memory block已清空

注意事項:

1. 運算中有沒有可能溢位(特別是變數範圍給得很接近 limit)

ASCII code 也會(%26 處理)

- 2. 注意不必要的符號、空格、換行被讀入
- 多利用 scanf("%d%c%d", &a, &useless, &b); or scanf("%d\n")
- 3. 使用指標指向structure時,不能讀取NULL的member,否則RE

其他函式庫

1. #include <stdlib.h>

```
(1) qsort(): sort Array
int cmp(const void *a, const void *b) {
    //要傳進去qsort的function
    //處理傳進來的兩個係數,再根據其level比較大小(>: return>0; <: return <0)
```

```
struct ESPer esper1 = *(struct ESPer*)a;
struct ESPer esper2 = *(struct ESPer*)b;

if(esper1.level == esper2.level)
    return 0;
else
    return (esper1.level > esper2.level) ? 1 : -1;
}

void sort_level(struct ESPer *arr, int length) {
    //qsort(array, arraySize, elementSize, compare function-需自已去定義)
    qsort(arr, length, sizeof(struct ESPer), cmp);
}
```

型別轉換

```
(2)
```

```
//int atoi( const char *str ); (字串有效部分轉 int)
//double atof( const char* str ); (字串轉 double)

const char *str = " 12.34e5trash";
  int i = atoi(str); //12
  double d = atof(str); //11274336.000000
  printf("%d %f", i, d);
```

2. <math.h>

函數名稱	功能	函數原型
fabs	求絕對值	<pre>double fabs(double);</pre>
fmax	求 x 與 y 之中的最大值	<pre>double fmax(double, double);</pre>
fmin	求 x 與 y 之中的最小值	double fmin(double, double);
remainder	求浮點餘數	<pre>double remainder(double, double);</pre>
fma	求 (x * y) + z	<pre>double fma(double, double, double);</pre>
round	四捨五入到整數位	<pre>double round(double);</pre>

函數名稱	功能	函數原型
sqrt	求平方根	<pre>double sqrt(double);</pre>
cbrt	求立方根	<pre>double cbrt(double);</pre>
ром	求 x 的 y 次方	<pre>double pow(double, double);</pre>

函數名稱	功能	函數原型
hypot	求 $x^2 + y^2$ 的平方根	<pre>double hypot(double, double);</pre>
sin	求三角函數的正弦	<pre>double sin(double);</pre>
cos	求三角函數的餘弦	<pre>double cos(double);</pre>
tan	求三角函數的正切	<pre>double tan(double);</pre>

函數名稱	功能	函數原型
log	求自然對數	<pre>double log(double);</pre>
log2	求以 2 為底的對數	<pre>double log2(double);</pre>
log10	求以 10 為底的對數	double log10(double);

printf

轉換說明符

轉換說明符	意義	實例	輸出
%a	浮點數、十六進位數字和p-記數法(C99)		
%A	浮點數、十六進位數字和P-記數法(C99)		
%c	單個字符		
%d	有符號十進位數		
%e	浮點數、e-記數法		
%E	浮點數、E-記數法		
%f	浮點數、十進位記數法		
%g	根據數值不同自動選擇%f或%e;%e格式在指數小於-4或者大於等於精度時使用		
%G	根據數值不同自動選擇%f或%E; %E格式在指數小於-4或者大於等於精度時使用		
%i	有符號十進位整數(與%d相同)		
%o	無符號八進位整數		
%p	指針		
%s	字符串		
%u	無符號十進位整數		
%x	使用十六進位數字(字母小寫)的無符號十六進位整數		
%X	使用十六進位數字(字母大寫)的無符號十六進位整數		
%%	列印百分號		

修飾符	意義	實例	輸出
+/-	修飾數字修飾符,表示正負		
空格			
#			
數字	整數部分表示最小欄位寬度,小數部分表示精度		
h	表示short,修飾整數表示有符號或無符號短整型		
hh	表示short short,修飾整數表示有符號或無符號char型		
j	修飾整數表示intmax_t或uintmax_t值		
I	表示long,修飾整數表示有符號或無符號長整型		
II	表示long long,修飾整數表示有符號或無符號長長整型		
L	表示long,修飾浮點數表示long double值		
t	修飾整數表示一個ptrdiff_t值(C99)		
Z	修飾整數表示一個size_t值(C99)		