



# C/C++基礎程式設計

字元與字串

講師：張傑帆  
CSIE, NTU

人的出身並不重要，你拿時間來做什麼才重要。

*It's not who you were at birth that matters, but what you do with the time you are given. -Steve Jobs*

# 課程大綱

- 字元
- 字串
- 作業



# 字元

- 在電腦的世界裡，所有的一切都是以**0**與**1**之數位訊號來表示
- 一個**字元(char)**可存0~255 (**1Byte**)
- 如何表示一個符號、數字或英文字母？
  - 給一個符號、數字或英文字母一個編號 (ASCII字碼)
  - EX:
    - $A \rightarrow 65, B \rightarrow 66, \dots, Z \rightarrow 90$
    - $a \rightarrow 97, b \rightarrow 98, \dots, z \rightarrow 122$
    - $0 \rightarrow 48, 1 \rightarrow 49, \dots, 9 \rightarrow 57$
    - $+ \rightarrow 43, - \rightarrow 45$

# 字元表

Ctrl	Dec	Hex	Char	Code	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
^@	0	00		NUL	32	20	!	64	40	@	96	60	'	128	80	Ç	160	A0	á	192	C0	Ł	224	E0	α			
^A	1	01		SOH	33	21	!	65	41	A	97	61	a	129	81	ü	161	A1	í	193	C1	ł	225	E1	β			
^B	2	02		STX	34	22	"	66	42	B	98	62	b	130	82	ë	162	A2	ó	194	C2	Ł	226	E2	Γ			
^C	3	03		ETX	35	23	#	67	43	C	99	63	c	131	83	â	163	A3	ô	195	C3	ł	227	E3	Π			
^D	4	04		EOT	36	24	\$	68	44	D	100	64	d	132	84	ä	164	A4	û	196	C4	Ł	228	E4	Σ			
^E	5	05		ENQ	37	25	%	69	45	E	101	65	e	133	85	å	165	A5	ñ	197	C5	ł	229	E5	σ			
^F	6	06		ACK	38	26	&	70	46	F	102	66	f	134	86	ä	166	A6	ä	198	C6	Ł	230	E6	μ			
^G	7	07		BEL	39	27	'	71	47	G	103	67	g	135	87	ç	167	A7	ö	199	C7	ł	231	E7	Υ			
^H	8	08		BS	40	28	(	72	48	H	104	68	h	136	88	ê	168	A8	ë	200	C8	Ł	232	E8	ϕ			
^I	9	09		HT	41	29	)	73	49	I	105	69	i	137	89	è	169	A9	ı	201	C9	ł	233	E9	θ			
^J	10	0A		LF	42	2A	*	74	4A	J	106	6A	j	138	8A	ë	170	AA	½	202	CA	Ł	234	EA	Ω			
^K	11	0B		VT	43	2B	+	75	4B	K	107	6B	k	139	8B	ï	171	AB	¼	203	CB	ł	235	EB	δ			
^L	12	0C		FF	44	2C	,	76	4C	L	108	6C	l	140	8C	î	172	AC	¼	204	CC	Ł	236	EC	ε			
^M	13	0D		CR	45	2D	-	77	4D	M	109	6D	m	141	8D	ï	173	AD	ı	205	CD	ł	237	ED	ϵ			
^N	14	0E		SO	46	2E	.	78	4E	N	110	6E	n	142	8E	Ï	174	AE	»	206	CE	Ł	238	EE	ϵ			
^O	15	0F		SI	47	2F	/	79	4F	O	111	6F	o	143	8F	Ä	175	AF	»	207	CF	ł	239	EF	ϵ			
^P	16	10		DLE	48	30	0	80	50	P	112	70	p	144	90	Å	176	B0	ı	208	D0	Ł	240	F0	≡			
^Q	17	11		DC1	49	31	1	81	51	Q	113	71	q	145	91	æ	177	B1	ı	209	D1	ł	241	F1	±			
^R	18	12		DC2	50	32	2	82	52	R	114	72	r	146	92	Æ	178	B2	ı	210	D2	Ł	242	F2	≤			
^S	19	13		DC3	51	33	3	83	53	S	115	73	s	147	93	ø	179	B3	ı	211	D3	ł	243	F3	≤			
^T	20	14		DC4	52	34	4	84	54	T	116	74	t	148	94	ö	180	B4	ı	212	D4	Ł	244	F4	≤			
^U	21	15		NAK	53	35	5	85	55	U	117	75	u	149	95	ö	181	B5	ı	213	D5	ł	245	F5	≤			
^V	22	16		SYN	54	36	6	86	56	V	118	76	v	150	96	û	182	B6	ı	214	D6	Ł	246	F6	÷			
^W	23	17		ETB	55	37	7	87	57	W	119	77	w	151	97	ù	183	B7	ı	215	D7	ł	247	F7	≈			
^X	24	18		CAN	56	38	8	88	58	X	120	78	x	152	98	ÿ	184	B8	ı	216	D8	Ł	248	F8	◊			
^Y	25	19		EM	57	39	9	89	59	Y	121	79	y	153	99	ÿ	185	B9	ı	217	D9	ł	249	F9	◊			
^Z	26	1A		SUB	58	3A	:	90	5A	Z	122	7A	z	154	9A	Ü	186	BA	ı	218	DA	Ł	250	FA	◊			
^[	27	1B		ESC	59	3B	;	91	5B	[	123	7B	{	155	9B	Ÿ	187	BB	ı	219	DB	ł	251	FB	◊			
^\	28	1C		FS	60	3C	<	92	5C	\	124	7C		156	9C	Ÿ	188	BC	ı	220	DC	Ł	252	FC	◊			
^]	29	1D		GS	61	3D	=	93	5D	]	125	7D	}	157	9D	Ÿ	189	BD	ı	221	DD	ł	253	FD	◊			
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~	158	9E	Ÿ	190	BE	ı	222	DE	Ł	254	FE	◊			
^-	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	Δ*	159	9F	f	191	BF	ı	223	DF	ł	255	FF	◊			

\* ASCII 碼 127 具有代碼 DEL。在 MS-DOS 下，這個代碼與 ASCII 8 (BS) 的效果相同。DEL 代碼可以由 CTRL + BKSP 鍵產生。

# 字元

- **getchar()**
  - 輸入字元
- **putchar()**
  - 輸出字元

```
#include <stdio.h>
int main()
{
    char ch;

    ch = getchar();
    putchar(ch);
    putchar('\n');

    return 0;
}
```

# 字元

- **getch()**
  - 輸入字元, 不需按enter, 輸入內容不顯示於螢幕
- **getche()**
  - 輸入字元, 不需按enter, 輸入內容顯示於螢幕

```
#include <conio.h>
int main()
{
    char ch;
    //ch = getche();
    ch = getch(); //比較使用getche()的差異

    return 0;
}
```

其中ch為字元變數。

下表為此三種輸入函式的比較表：

字元輸入函式	<Enter>鍵	螢幕上該字元	標頭檔
getchar()	需要	會顯示	stdio.h
getche()	不用	會顯示	conio.h
getch()	不用	不顯示	conio.h

# Visual C++ 的問題

```
'strcpy': This function or variable may be unsafe.  
Consider using strcpy_s instead. To disable deprecation,  
use _CRT_SECURE_NO_WARNINGS. See online help for details.
```

- 在 Project Properties -> Configuration Properties -> C/C++ -> Preprocessor -> Preprocessor Definitions
- 加入這兩行

**\_CRT\_SECURE\_NO\_DEPRECATE**

**\_CRT\_NONSTDC\_NO\_DEPRECATE**



組態(C): 作用中 (Debug)

平台(P): 作用中 (Win32)

組態管理員(O)...

## 通用屬性

架構和參考

## 組態屬性

一般

偵錯

VC++ 目錄

## C/C++

一般

最佳化

前置處理器

程式碼產生

語言

先行編譯標頭檔

輸出檔

瀏覽資訊

進階

所有選項

命令列

連結器

資訊清單工具

XML 文件產生器

瀏覽資訊

建置事件

自訂建置步驟

程式碼分析

## 前置處理器定義

WIN32;\_DEBUG;\_CONSOLE;%(PreprocessorDefinitions)

取消前置處理器的定義

取消所有前置處理器的定義

否

忽略標準的 Include 路徑

否

## 前置處理器定義

```
WIN32
_DEBUG
_CONSOLE
_CRT_SECURE_NO_DEPRECATED
_CRT_NONSTDC_NO_DEPRECATED
```

繼承值:

```
_UNICODE
UNICODE
```

☒ 從父代或專案預設值繼承(I)

巨集(M) &gt;&gt;

確定

取消

## 前置處理器定義

為原始程式檔定義前置處理符號。

確定

取消

套用(A)

# 字元應用

- 判斷輸入字元是大寫英文，小寫英文，數字，或其他

```
#include <stdio.h>

int main()
{
    char ch;
    ch = getchar();

    if(ch>='0' && ch<='9')
        printf("你輸入了數字\n");
    else if(ch>='A' && ch<='Z')
        printf("你輸入了大寫英文\n");
    else if(ch>='a' && ch<='z')
        printf("你輸入了小寫英文\n");
    else
        printf("你輸入了其他字元\n");

    return 0;
}
```

# 字元應用

- 將輸入的小寫英文轉成大寫英文

```
#include <stdio.h>

int main()
{
    char ch;
    ch = getchar();

    if(ch>='a' && ch<='z') {
        ch-=32; //或 ch-=('a'-'A');
        printf("%c\n", ch);
    }
    else if(ch>='A' && ch<='Z')
        printf("%c\n", ch);
    else
        printf("你輸入的不是英文字母\n");

    return 0;
}
```

# 課程大綱

- 字元
- 字串
- 作業



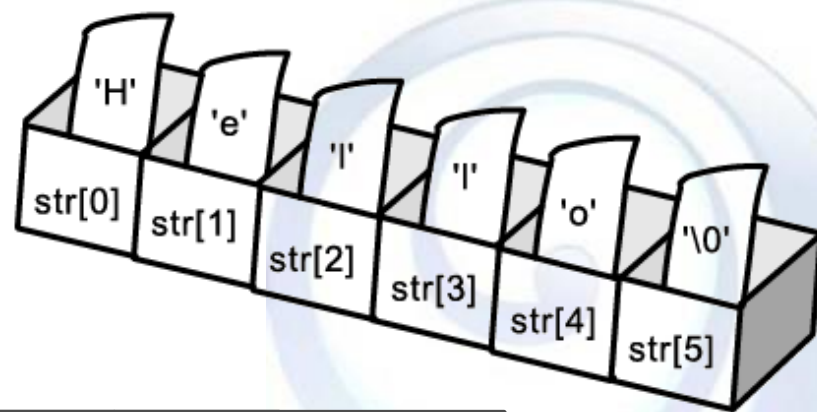
# 字串

- 在程式語言中，一個英文單字，一個句子，都可以當成一個字串
- 在C語言中，一個一維的的字元陣列可以當成一個字串
- 在C語言中，可用雙引號"文字"表示一個字串
- 字串 = 一堆字元 + 字串結束字元'\0'
  - `char a[]={'H', 'e', 'l', 'l', 'o', '\0'};`
  - `char a[]="Hello";`

位置: a



	'H'	'e'	'l'	'l'	'o'	'\0'	
--	-----	-----	-----	-----	-----	------	--



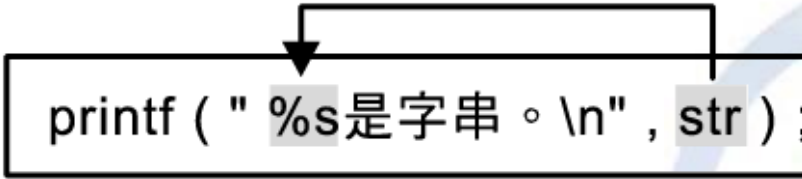
# 字串

- 以下a,b,c皆可表示Hello字串

```
#include <stdio.h>
int main()
{
    char a[]="Hello"; //字串初始化
    char b[]={'H','e','l','l','o','\0'}; //字元陣列初始化

    char c[6]; //字元陣列宣告
    c[0]='H';
    c[1]='e';
    c[2]='l';
    c[3]='l';
    c[4]='o';
    c[5]='\0'; // '\0'是字串的結束符號

    printf("%s \n",a); //字串a輸出
    printf("%s \n",b); //字串b輸出
    printf("%s \n",c); //字串c輸出
    return 0;
}
```



# 字串輸入輸出

- 利用scanf來讀取一個字串在用printf印出
- 比較: 陣列長度 vs. 字串長度
  - 陣列長度: 陣列中可儲存幾個資料
  - 字串長度: 字串結束字元前有多少個字元

```
#include <stdio.h>
int main()
{
    char a[80];
    scanf("%s", &a);    //或用a
    printf("%s \n",a);
    return 0;
}
```

# 字串輸入輸出

- 用scanf來讀字串，字串中**不能有空白**，若有**空白**會被當成兩個不同的字串。
- gets(a)
  - 輸入字串a，以**enter**鍵做為字串結束
- puts(a)
  - 輸出字串a

```
#include <stdio.h>
int main()
{
    char a[80];

    gets(a);
    puts(a);

    return 0;
}
```



# 字串輸入輸出

- 當scanf遇上gets
  - scanf輸入完後所按下之enter鍵會殘留在輸入緩衝區stdin(或稱標準輸入檔) 將影響到下一次的gets輸入
- 解法: 使用fflush(stdin);

```
#include <stdio.h>
int main()
{
    char a[80];
    char b[80];

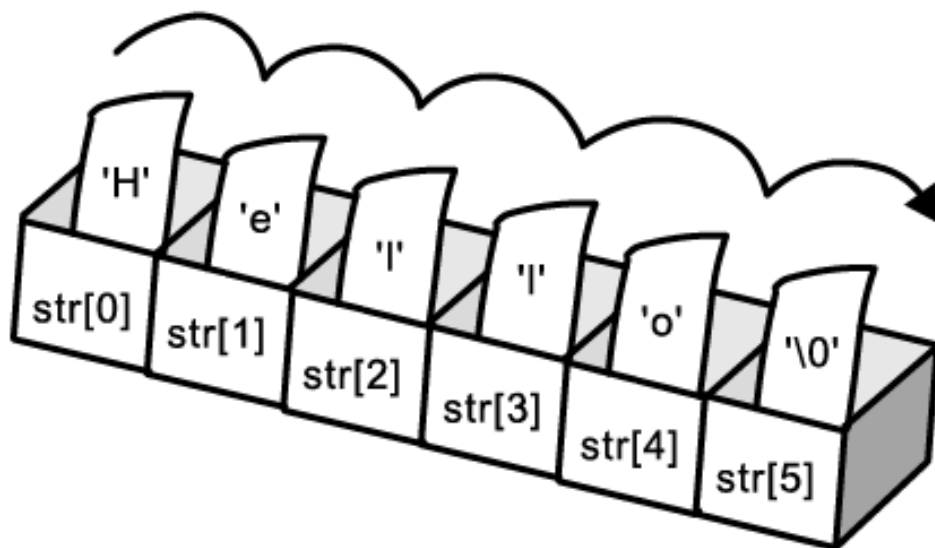
    scanf("%s", a);
    gets(b); //此gets無法正常輸入

    printf("%s\n", a);
    printf("%s\n", b);

    return 0;
}
```

## 練習

- 輸入一字串將其中小寫字元轉成大寫字元
- 例如：輸入abCdE123 → 輸出ABCDE123



# 使用二維陣列存放字串

- 當有多個字串要儲存  
可以使用字元二維陣列

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int i;
```

```
    char name[3][80];
```

```
    for(i=0; i<3; i++) { //輸入三個人姓名
```

```
        printf("姓名%d: ", i+1);
```

```
        gets(name[i]);
```

```
    }
```

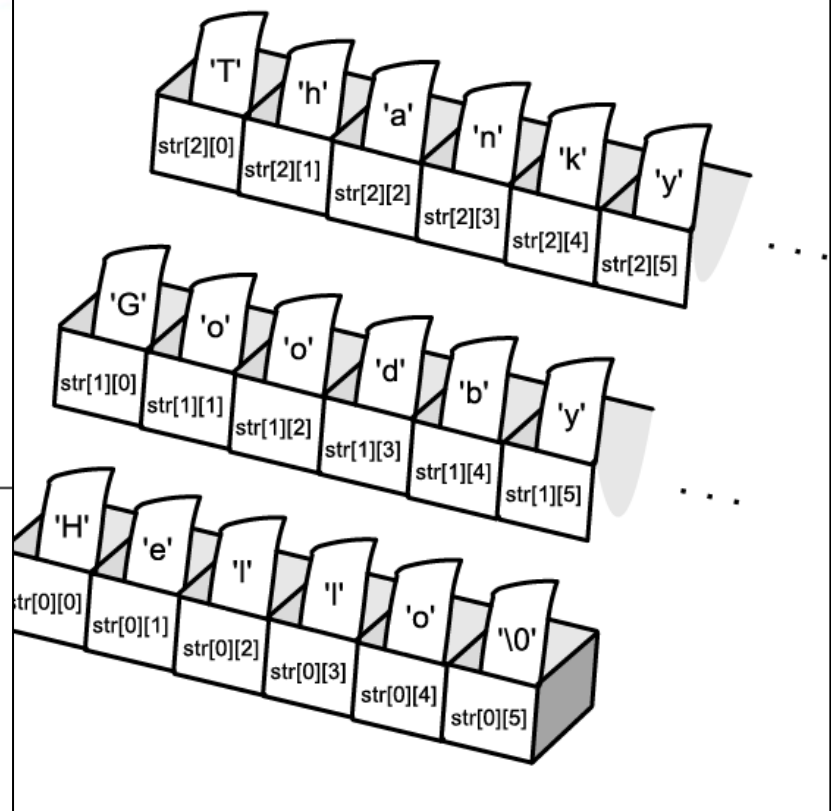
```
    for(i=0; i<3; i++) { //輸出三個人姓名
```

```
        printf("姓名%d: %s\n", i+1, name[i]);
```

```
    }
```

```
    return 0;
```

```
}
```



# 字串應用

- C語言中提供許多字串相關函式可以使用, 以下介紹常用的幾個函式:
  - 字串比對: strcmp
  - 字串複製: strcpy
  - 字串連接: strcat
  - 計算字串長度: strlen
- C語言字元字串函式參考網站
  - <http://www.cppreference.com/wiki/c/string/start>
  - <http://www.cplusplus.com/reference/clibrary/cstring/>

# 字串應用

- 字串原理: 字串用字元陣列儲存
  - 陣列名稱: 陣列開頭之記憶體位置
  - 因此，字串處理 = 陣列處理
- 
- 字串處理相當複雜, 但又太常用了, 因此C語言提供了一些函式讓我們可以方便使用

# 字串比對錯誤用法!

- 下列程式可以執行，但沒有意義

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    char a[]="Hello";
```

```
    char b[80];
```

```
    scanf("%s", &b);
```

```
    if ( a==b )
```

```
    {
```

```
        printf("輸入字串正確\n");
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("輸入字串錯誤\n");
```

```
    }
```

```
    return 0;
```

```
}
```

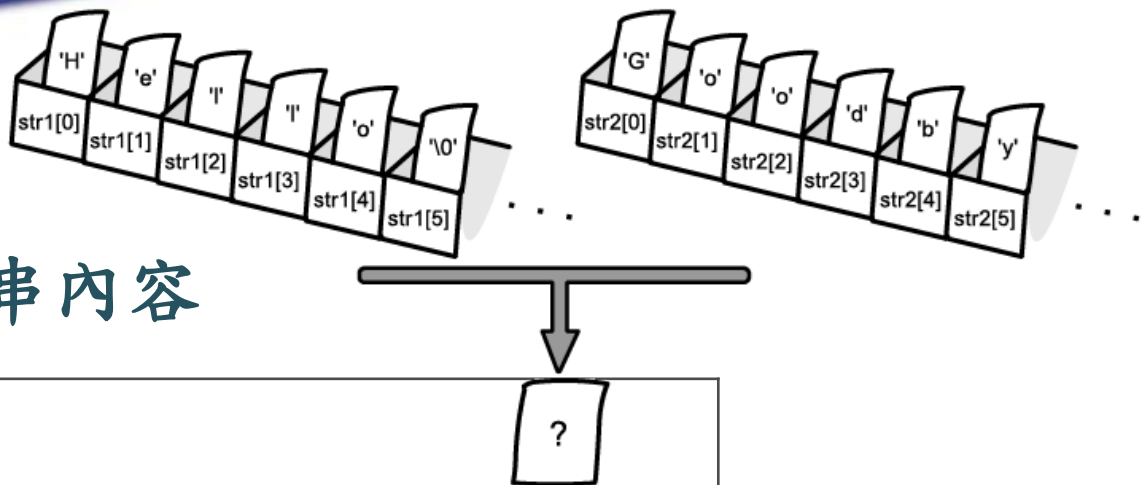
# 字串比對: strcmp

- strcmp(a, b): 比對a,b字串之內容
  - strcmp(a,b) > 0 : a字串字母順序較大
  - strcmp(a,b) < 0 : b字串字母順序較大
  - strcmp(a,b) == 0 : a,b兩字串內容一樣

```
#include <stdio.h>
#include <string.h>
int main()
{
    char a[80]="Hello";
    char b[80];

    scanf("%s", &b);
    if ( strcmp(a,b)==0 ) {
        printf("輸入字串正確\n");
    }
    else {
        printf("輸入字串錯誤\n");
    }
    return 0;
}
```

# 字串比對: **strcmp**



- 範例: 比較兩個字串內容

```
#include <stdio.h>
#include <string.h>
int main()
{
    char a[80];
    char b[80];
    printf("輸入第一個字串: ");
    scanf("%s", &a);
    printf("輸入第二個字串: ");
    scanf("%s", &b);
    if ( strcmp(a,b) > 0 ) {
        printf("輸入第一個字串字母順序較大\n");
    }
    else if ( strcmp(a,b) < 0 ) {
        printf("輸入第二個字串字母順序較大\n");
    }
    else {
        printf("二個字串內容相同\n");
    }
    return 0;
}
```



# 字串複製錯誤用法!

- 下列程式不可執行

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    char a[80];
```

```
    char b[80]="How are you";
```

```
    a = b; // 錯誤! a為記憶體位置, 為一個常數, 不可存放資料
```

```
    printf("%s\n", a);
```

```
    return 0;
```

```
}
```

# 字串複製: strcpy

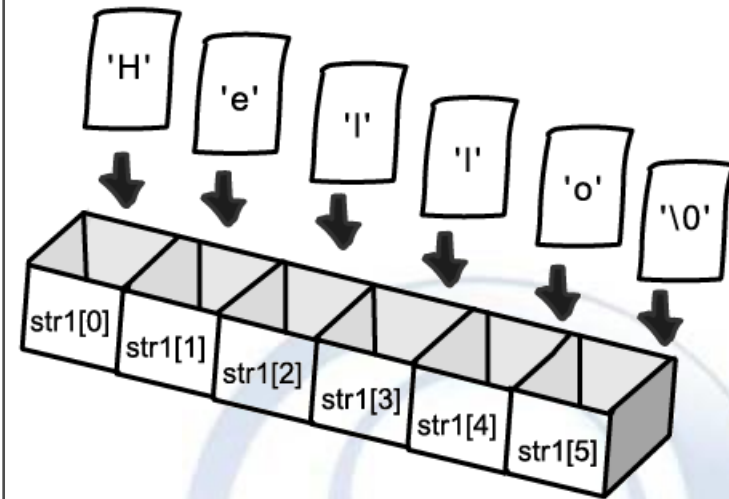
- **strcpy(a, b):** 將b字串內容複製到a字串

```
#include <stdio.h>
#include <string.h>

int main()
{
    char a[80];
    char b[80]="How are you";

    strcpy(a, b);

    printf("%s\n", a);
    return 0;
}
```



# 字串連接錯誤用法!

- 下列程式不可執行

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    char a[80]="Hi, ";
```

```
    char b[80]="how are you";
```

```
    a+=b; // 錯誤! a為記憶體位置, 為一個常數, 不可存放資料
```

```
    printf("%s\n", a);
```

```
    return 0;
```

```
}
```

## 字串連接: **strcat**

- **strcat(a, b)**: 將b字串內容加到a字串後面

```
#include <stdio.h>
#include <string.h>

int main()
{
    char a[80]="Hi, ";
    char b[80]="how are you";

    strcat(a, b);

    printf("%s\n", a);

    return 0;
}
```

```
strcpy (str1, "Hello") ;
```

str1     

H	e	l	l	o	\0						
---	---	---	---	---	----	--	--	--	--	--	--

```
strcpy (str0, "Goodbye") ;
```

str2             

G	o	o	d	b	y	e	\0		
---	---	---	---	---	---	---	----	--	--

```
strcpy (str0, str1) ;
```

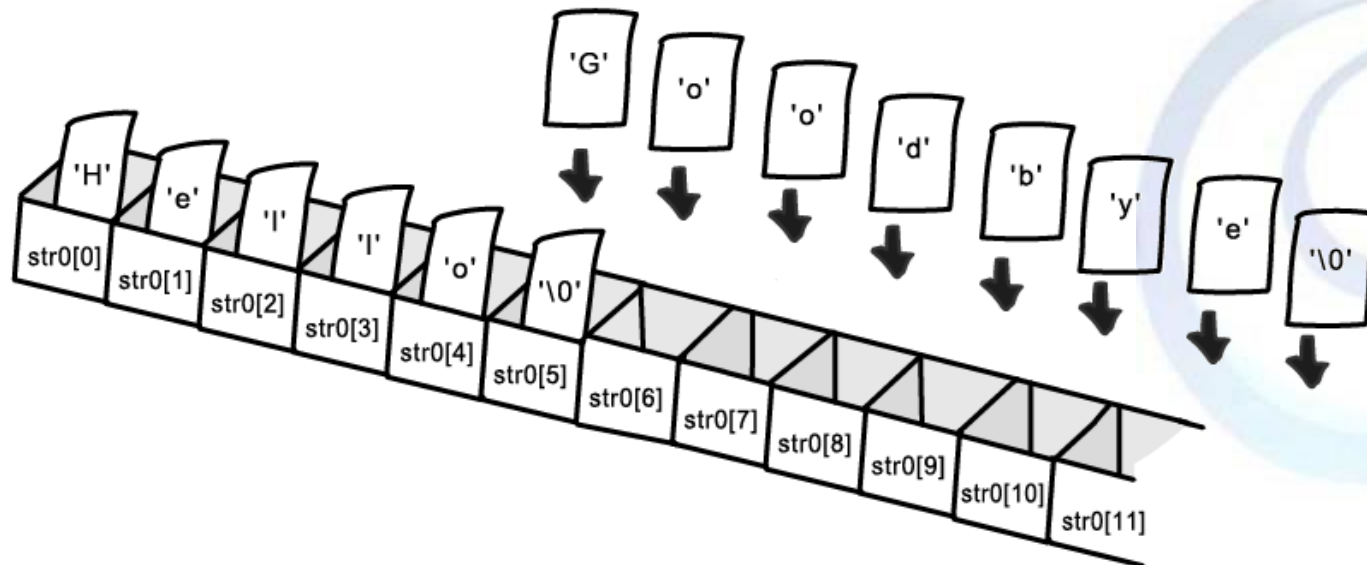
str0     

H	e	l	l	o	\0						
---	---	---	---	---	----	--	--	--	--	--	--

```
strcat (str0, str2) ;
```

str0     

H	e	l	l	o	G	o	o	d	b	y	e	\0		
---	---	---	---	---	---	---	---	---	---	---	---	----	--	--



- 注意陣列的大小

- 在把字串複製到陣列的情況下，要注意不可以超出陣列的大小。

- 範例： ← 不可以使用大小不足的陣列

```
int main(void)
```

```
{
```

```
    char str0[10];
```

```
    char str1[10];
```

```
    char str2[10];
```

```
    str2
```

G	o	o	d	b	y	e	\0		
---	---	---	---	---	---	---	----	--	--

```
    strcat (str0, str2) ;
```

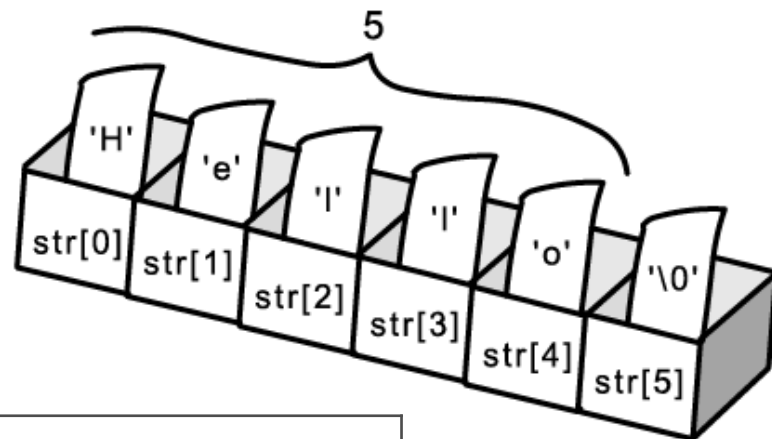
```
    str0
```

H	e	l	l	o	G	o	o	d	b	y	e	\0		
---	---	---	---	---	---	---	---	---	---	---	---	----	--	--

這部分之記憶體範圍的資料會遭到破壞

# 計算字串長度: **strlen**

- **strlen(a)**: 計算a字串長度



```
#include <stdio.h>
#include <string.h>

int main()
{
    char a[80];
    int n;

    gets(a);
    n = strlen(a);

    printf("字串長度 = %d\n", n);

    return 0;
}
```

# 課程大綱

- 字元
- 字串
- 作業





# 猜數字遊戲

- 輸入四位不重覆數字 (0~9), 做為電腦的題目
- 輸入四位不重覆數字 (0~9), 做為您猜的答案
- 不需檢查使用者輸入之格式
- 當輸入之答案與題目相同, 程式結束
- **遊戲規則**
  - 電腦的題目: 1234  
您猜的數字: 5283
  - 結果為 1A1B  
表示您共猜對了兩位數，其中有一個字位置對，另一個字位置不對
  - 其中，A 表示猜對一個字且位置也對，B 表示猜對一個字但是位置不對。

## 範例

```
Enter the question: 1234
guess: 5283
1A1B
guess: 5290
1A0B
guess: 7890
0A0B
guess: 1256
2A0B
guess: 5634
2A0B
guess: 1234
4A0B
you win! bye!
Press any key to continue
```

```
Please enter anser:
1234
Please enter guess:
1324
2A2B
Please enter guess:
1564
2A0B
Please enter guess:
1111
1A3B
Please enter guess:
8146
0A2B
Please enter guess:
9999
0A0B
Please enter guess:
1234
4A0B
You Win!
```

```
-----
Process exited after :
請按任意鍵繼續 . . .
```

## 附錄：輸出入函式參考表

- ANSI C所提供的標準輸出入函式如下表所示：

類型	函式名稱	功能	資料型別	標頭檔
標準輸出函式	printf()	格式化輸出函式	不拘型別	stdio.h
	putchar	字元顯示函式	字元	stdio.h
	putch	字元顯示函式	字元	conio.h
	puts	字串顯示函式	字元	stdio.h
標準輸入函式	scanf	格式化輸入函式	不拘型別	stdio.h
	getchar	單一字元輸入函式	字元	stdio.h
	getche	單一字元輸入函式	字元	conio.h
	getch	單一字元輸入函式	字元	conio.h
	gets	字串輸入函式	字元	stdio.h