string.h

strcpy vs. strncpy

以為是一個安全玩具，怎麼用都安全!?

真相是：會的人「小心使用」才安全

參數越多，會出錯的地方越多

一定要那麼辛苦嗎? 有不用動腦可以完成的嗎?

往好的地方想，你會了以後競爭的人會比較少

strcpy

- char ***strcpy**(char *destination, const char *source);

char src[10] = “hello”;

‘h’	‘e’	‘l’	‘l’	‘o’	0	7	‘x’	‘e’	‘r’

strcpy(dest, src);

char dest[8];

3	‘s’	0	-1	‘A’	12	‘d’	9

strcpy

- char ***strcpy**(char *destination, const char *source);

char src[10] = “hello”;

‘h’	‘e’	‘l’	‘l’	‘o’	0	7	‘x’	‘e’	‘r’

strcpy(dest, src);

char dest[4];

3	‘s’	0	-1		

如果這裡原先有重要東西，就被覆蓋掉了

是什麼東西會被覆蓋掉?
不一定耶，不同程式
不同環境都不一樣

程式表現無法預期

```
while (source[i]!=0) {
   destination[i] = source[i];
   i = i + 1;
}
destination[i] = 0;
```

為什麼不檢查一下 destination 陣列的大小??

strncpy

- char ***strncpy**(char *destination, const char *source, size_t num);

這樣子行了吧，str**n**cpy 自己可以檢查一下目標的大小

char src[10] = “hello”;

‘h’	‘e’	‘l’	‘l’	‘o’	0	7	‘x’	‘e’	‘r’

strncpy(dest, src, sizeof(dest));

sizeof(char[8])

char dest[8];

3	‘s’	0	-1	‘A’	12	**‘0’**	**0**

最後這兩個 0 是讓它更安全嗎?

strncpy: error 1

- char ***strncpy**(char *destination, const char *source, size_t num);

請注意： char src[10] = “hello”;

‘h’	‘e’	‘l’	‘l’	‘o’	0	7	‘x’	‘e’	‘r’

strncpy(dest, src, sizeof(dest));

char dest[4];

3	‘s’	0	-1

需要加上 dest[3] = 0; 才能當作字串使用

或是
char dest[4] = {**0**};
…
strncpy(dest, src, sizeof(dest)-**1**)

糟糕!! 這樣子少了結束字元，不是合法的字串了

strncpy: error 2

- char ***strncpy**(char *destination, const char *source, size_t num);

char src[10] = “hello”;

‘h’	‘e’	‘l’	‘l’	‘o’	0	7	‘x’	‘e’	‘r’

不小心給錯了

strncpy(dest, src, 10);

char dest[8];

3	‘s’	0	-1	‘A’	12	‘o’	**0**	**0**	**0**

```
while (i<num&&source[i]!=0) {
   destination[i] = source[i];
   i = i + 1;
}
while (i<num) destination[i] = 0;
```

最後這兩個 0 是根據你給的 10 這個參數做的, 有雷, 程式表現無法預期!