程式語言一　講義

字串

Hanly《C 語言詳論》6e

§ 1　名詞速查表

中文	English	一句話定義	科目	章節
字串	string	一串字元；C 裡就是 **char 陣列 + 結尾 '\0'**	程式語言一	字串
空字元	null terminator（`'\0'`）	ASCII 0，標記字串「到此結束」	程式語言一	字串
字元	char	單一字元型態，1 byte；`'A'` 用單引號	程式語言一	字串
strlen	strlen	算長度（不含 `'\0'`），需 `<string.h>`	程式語言一	字串
strcpy	strcpy	複製字串（陣列不能用 `=`）	程式語言一	字串
strcmp	strcmp	比較**內容**（不能用 `==`），相等回 0	程式語言一	字串
fgets	fgets	安全讀整行（含空格、含 `'\n'`），限長度	程式語言一	字串
緩衝區溢位	buffer overflow	寫超出陣列大小、覆蓋鄰近記憶體，安全漏洞根源	程式語言一	字串
ctype.h	ctype.h	字元分類／轉換：`isalpha`、`toupper`…	程式語言一	字串

§ 2 核心概念

核心概念

Python 有 `str`、Java 有 `String`，但 **C 沒有字串型態**。C 的字串就是一個 **char 陣列**，結尾放一個 `'\0'`（**空字元，null terminator**）當作「字串到這裡結束」的記號。

```
char str[] = "Hello";   // 編譯器自動在結尾加 '\0'
```

`"Hello"` 有 5 個字元，卻佔 **6 bytes**（多一個 `'\0'`）。所有字串函式（`printf("%s")`、`strlen`…）都靠 `'\0'` 知道字串在哪裡結束。沒有 `'\0'` 就會一路讀到垃圾。承接 [[陣列]]：字串就是 char 陣列，傳函式同樣是傳址。

§ 3 主要內容

3.1 字串的本質：char[] + '\0'

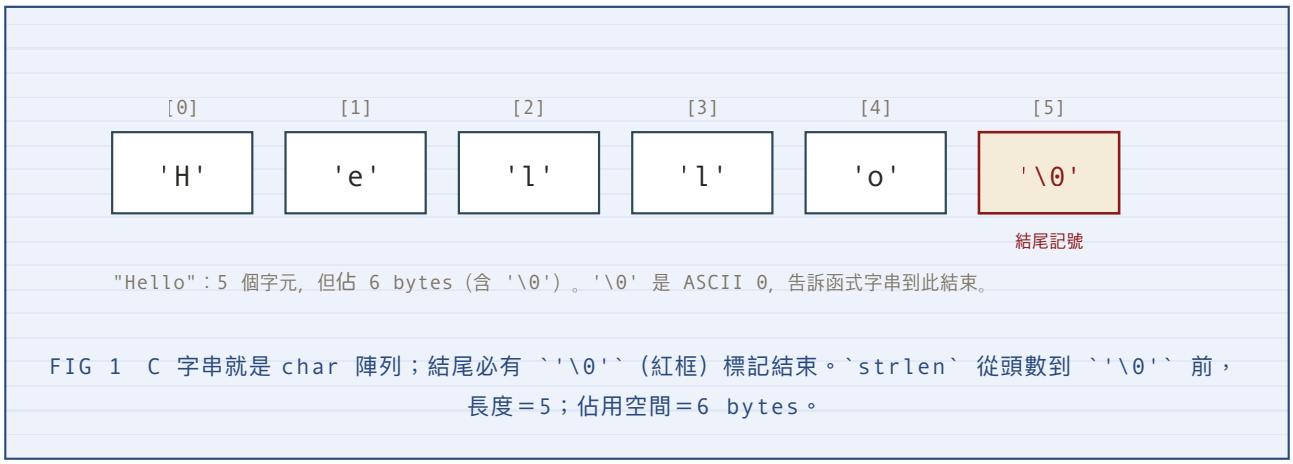

FIG 1 C 字串就是 char 陣列；結尾必有 ``'\0'``（紅框）標記結束。`strlen` 從頭數到 ``'\0'`` 前，長度＝5；佔用空間＝6 bytes。

3.2 宣告與讀取

```
char s1[] = "Hello";        // 編譯器算大小＝6（含 '\0'）
char s2[10] = "Hello";      // 大小 10，後面補 '\0'
char s4[100];               // 未初始化，稍後讀入
// char s[5] = "Hello";     // 錯！沒空間放 '\0'，至少要 6
```

讀字串兩種：`scanf("%s", s)`（**不用 `&`**，讀到空格就停、無長度限制會溢位）；`fgets(s, 100, stdin)`（讀整行含空格、**有長度限制較安全**，但會把換行 `'\n'` 也讀進來）。建議用 `fgets`。

3.3 遍歷與手寫 strlen

```
char s[] = "Hello";
for (int i = 0; s[i] != '\0'; i++) printf("%c", s[i]);  // 遍歷到 '\0' 就停

int len = 0;
while (s[len] != '\0') len++;         // 這就是 strlen 的原理，len = 5
```

`s[i] != '\0'` 是遍歷字串的標準寫法。`'\0'` 不算進長度。

3.4 不能用 `=` 和 `==`：要用函式

```
#include <string.h>
char a[10], b[10];
// a = "Hello";           // 錯！陣列不能整個 =
strcpy(a, "Hello");       // 用 strcpy 複製
// if (a == b) ...         // 錯！== 比的是位址，不是內容
if (strcmp(a, b) == 0) { /* 內容相同 */ }
```

函式	功能	備註
`strlen(s)`	長度（不含 `'\0'`）	—
`strcpy(d, s)`	複製 s 到 d	不檢查 d 大小
`strcat(d, s)`	把 s 接到 d 後面	不檢查 d 大小
`strcmp(a, b)`	比較內容	0＝相等、<0、>0（逐字元比 ASCII）

3.5 緩衝區溢位（安全）

```
char buf[8];
scanf("%s", buf);   // 使用者輸入 "HelloWorld123" → 寫爆 buf、覆蓋鄰近記憶體
```

`strcpy`、`strcat`、`scanf("%s")` 都**不檢查大小**，輸入過長就**緩衝區溢位（buffer overflow）**：可能 crash，也是經典攻擊手法（1988 Morris Worm）。安全做法：`fgets(buf, 8, stdin)` 或 `scanf("%7s", buf)` 限制長度。

3.6 字元處理與應用

`<ctype.h>` 提供 `isalpha`、`isdigit`、`isupper`、`islower`、`isspace`、`toupper`、`tolower`，比自己寫 `c >= 'a' && c <= 'z'` 清楚。常見應用：

```c
// 大小寫互換（大小寫 ASCII 差 32）
for (int i = 0; s[i]; i++) {
    if (s[i] >= 'a' && s[i] <= 'z') s[i] -= 32;
    else if (s[i] >= 'A' && s[i] <= 'Z') s[i] += 32;
}
// 字元頻率：用 c-'a' 當索引（承接陣列「用值當索引」）
int cnt[26] = {0};
for (int i = 0; s[i]; i++) if (s[i] >= 'a' && s[i] <= 'z') cnt[s[i]-'a']++;
// Caesar 位移：(c-'a'+k)%26 + 'a'
```

§ 4 語法與函式速查

```c
char s[] = "Hello";              // char 陣列 + 自動 '\0'，佔 6 bytes
s[i] != '\0'                     // 遍歷到結尾的標準條件
scanf("%s", s);                  // 不用 &；讀到空格停；會溢位
fgets(s, n, stdin);              // 安全讀整行（含 '\n'）
#include <string.h>              // strlen / strcpy / strcat / strcmp
strcmp(a, b) == 0                // 比較內容相等（不可用 a == b）
#include <ctype.h>               // toupper / tolower / isalpha …
```

§ 5 常見錯誤

常見錯誤

- 忘了 `'\0'`：手動建字串沒補結尾，`printf("%s")` 一路讀到垃圾。
- 空間不夠：`"Hello"` 要 6 bytes，`char s[5]` 放不下 `'\0'`。
- 用 `==` 比字串：比的是位址不是內容，要用 `strcmp(a,b)==0`。
- 用 `=` 指派字串：陣列不能整個 `=`，要用 `strcpy`。
- `fgets` 的換行：會讀進 `'\n'`，需要時手動去掉。
- `scanf("%s")` 溢位：不限長度，改 `fgets` 或 `scanf("%99s")`。

§ 6 練習題

練習 1（一般題）：字串追蹤

下列印什麼（兩行）？

```
char s[] = "abc";
s[1] = 'x';
printf("%s\n", s);
printf("%d\n", s[3] == '\0');
```

> **引導步驟**
>
> 1. `s[1]='x'` 把第二個字元改掉。
> 2. `"abc"` 佔幾 bytes？`s[3]` 是什麼？

解答

第一行 `axc`（s[1] 改成 'x'）。第二行 `1`：`"abc"` 佔 4 bytes，`s[3]` 正是結尾的 `'\0'`，`'\0' == '\0'` 為真＝1。

練習 2（一般題）：strcmp 回傳值

下列三個 `strcmp` 各回傳大於 0、小於 0、還是等於 0？

```
strcmp("apple", "banana");
strcmp("hello", "hello");
strcmp("abc", "abd");
```

> **引導步驟**
>
> 1. 逐字元比 ASCII，第一個不同的字元決定大小。

解答

① `< 0`（'a' < 'b'）② `== 0`（完全相同）③ `< 0`（前兩字元同，第三個 'c' < 'd'）。`strcmp` 逐字元比 ASCII，相等回 0、a 在前回負、a 在後回正。

練習 3（重要題）：抓 bug

下列想判斷兩字串內容是否相同，為何永遠印 `diff`？怎麼修？

```
char a[] = "cat", b[] = "cat";
if (a == b) printf("same\n");
else        printf("diff\n");
```

> **引導步驟**
>
> 1. `a` 和 `b` 是兩個不同的陣列，`a`、`b` 各代表什麼？
> 2. `==` 比的是內容還是位址？

解答

`a`、`b` 是兩個獨立陣列，陣列名代表各自的**起始位址**，位址不同，所以 `a == b` 永遠為假、印 `diff`。`==` 比位址不比內容。修：用 `strcmp` 比內容。

```
if (strcmp(a, b) == 0) printf("same\n");
else                   printf("diff\n");
```

> **易錯**
>
> · 以為 `==` 會比字串內容。比較內容一律 `strcmp(a,b)==0`。

§ 7 自我檢核

- [ ] 知道 C 沒有字串型態，字串＝char 陣列 + `'\0'`。
- [ ] 會算 `"Hello"` 佔 6 bytes，能畫出含 `'\0'` 的記憶體圖。
- [ ] 會用 `s[i] != '\0'` 遍歷字串、手寫 strlen。
- [ ] 知道字串不能用 `=`／`==`，要用 `strcpy`／`strcmp`。
- [ ] 懂緩衝區溢位的成因與安全做法（fgets、限長度）。
- [ ] 會用 ctype.h 與「`c-'a'` 當索引」做大小寫、頻率、Caesar。
- [ ] 分得清 `scanf("%s")`（讀到空格停、會溢位）與 `fgets`（整行、含 `'\n'`、安全）。