

[C語言_02] 秒懂字串處理函數（上）



Andy Cheng

Follow

Apr 3, 2019 · 7 min read



前情提要

上一篇C語言談到了字元陣列與字串的一些區別，相信大家對字串都有一定的熟悉，因此這篇就來跟大家介紹幾個非常實用，也容易搞混的字串函數吧！

本篇內容包含：

gets, puts, fgets, fputs, strlen, strcpy, strncpy

. . .

（一）gets 函數

函數定義：

```
char *gets(char *s);
```

參數

- 字串 s

功能說明：

讀取字串，依序將每個字元存入char的陣列中，直到出現「**換行符**」或「**文件結尾**」為止。

白話文解釋：

就是將輸入的字串讀到陣列中。

返回值：

- 成功：讀到的字串
- 失敗：NULL

使用方法：

- gets(字串);

使用範例：

```
// 先定義一個字元陣列
char ch[100];

// 讀取輸入的字串
gets(ch); // ex: Hello!!

// 印出字串
printf("%s",ch); ex: Hello!!
```

gets(str) vs scanf("%s",str)

我們最常用讀入字串的函數就是scanf，那麼這兩個函數差在哪裡呢？

- gets(str)允許輸入的字符含有空格
- scanf("%s",str)不允許輸入字符串含有空格

所以我們看下面的程式，會發現scanf("%s",str)只會讀到空格前面的字串：

```
輸入：Hello World

gets(ch);
// Output: Hello World

scanf("%s",str);
// Output: Hello
```

注意事項：

當你編譯包含gets的程式時，可能會出現warning說gets是不安全的，原因是gets() 及 scanf()都無法知道字串s大小，必須遇到換行符或讀到文件結尾為止才接收輸入，因此容易導致字元陣列overflow的情況。

意思就是說，當你輸入的字串數量超過接收字串的陣列大小，就很容易發生overflow。

(二) fgets 函數

函數定義：

```
char *fgets(char *s, int size, FILE *stream);
```

參數：

- s：字串
- size：指定最大讀取字串的長度（size-1）
- stream：文件指標，如果讀鍵盤輸入的字串，固定寫為stdin

功能說明：

從stream指定的文件內讀入字符，保存到s所指定的內存空間，直到出現換行字符、讀到文件結尾或是讀了size -1個字符為止，最後會自動加上字符'\0'作為字符串結束。

白話文說明：

就是從特定文件中讀取指定長度的字串，然後存到字元陣列中。

返回值：

- 成功：讀到的字串
- 失敗：NULL

白話使用方法：

`fgets(字串, 能讀入的最大字串長度, 從哪裡讀入);`

使用範例：

第一種：輸入大小小於原始指標的區域大小

輸入大小小於原始指標的區域大小，會在輸入完成自動在後面加上「\n」、「\0」

```
char arr[100];

// 輸入 hello world
fgets(arr, 100, stdin); //標準輸入
// 輸出 hello world + 換行
printf("str = \"%s\"\n", str);
```

第二種：輸入大小大於等於原始指標的區域大小

輸入大小小於原始指標的區域大小，會在輸入完成自動加上\0而不會再加上\n

```
char arr[10];

// 輸入hello world
fgets(arr, 10, stdin); //標準輸入

// 輸出： hello wor
printf("str = \"%s\\n\"", str);
```

fgets vs gets

fgets 在讀取字串會把輸入的「Enter」也作為字串一部分，通過gets或scanf輸入字串時不會包含：

- fgets結尾 -> '\0' '\n'
- gets結尾 -> '\0'

fgets因為有設定最大能讀取的字串長度，所以不會有overflow的問題，比gets,scanf安全：)

(三) puts()函數

函數定義：

```
int puts(const char *s);
```

參數：

- s：字串

功能說明：

將s字串輸出至螢幕上，輸出完成後自動輸出一個'\n'

白話文說明：

跟printf('%s',str)一樣都是印出資料，但多了一個

返回值：

- 成功：非負數
- 失敗：-1

使用方法：

puts(字串)

使用範例：

```
printf("hello world");  
// 輸出 hello world
```



```
puts("hello world");  
// 輸出 hello world
```

(四) fputs() 函數

函數定義：

```
int fputs(const char * str, FILE * stream);
```

參數：

- str：字串
- stream：文件指標，如果輸出到螢幕上，固定寫為stdout

功能說明：

將str字串寫入stream指定的文件，直到'\0'，'\0'但不寫入文件。

白話文說明：

放字串到文件中，puts的文件操作版，但fputs不會自動輸出一個

返回值：

- 成功：0
- 失敗：-1

白話使用方法：

fputs(字串, 寫入文件);

使用範例：

```
// 參數：字符指標 文件流
char arr[10] = "he\0llo";

fputs(arr, stdout);
// 輸出：he
```

（五）strlen函數

函數定義：

```
size_t strlen(const char *s);
```

參數：

- s：字串

功能說明：

取出字串的有效長度，忽略\0。

白話文說明：

str = “12345”

strlen(str) = 5

返回值：

- 字串s的長度

使用範例：

（一）字串

```
char arr[] = "hello";
int len = strlen(arr);
printf("字符串有效長度：%d", len);
printf("字符串長度：%d", sizeof(arr));
// 輸出
字符串有效長度:5
字符串長度:6
```

(二) 字元陣列 (非字串)

```
char arr[] = {"a", "n", "d", "y"};
int len = strlen(arr);
printf("字符串有效長度: %d", len);
printf("字符串長度: %d", sizeof(arr));
// 輸出
字符串有效長度: ?? 因為找不到{\0}
字符串長度: 4
```

strlen vs sizeof

- strlen：處理字符串（不包含\0），但是不能處理字元陣列
- sizeof：所有字元的長度（包含\0）

(六) strcpy() 函數

函數定義：

```
char *strcpy(char *dest, const char *src);
```

參數：

- dest：目的字串
- src：原來的字串

功能說明：

把src的字串複製到dest所指向的空間中，'\0'也會拷貝過去

白話文說明：

拷貝A字串到B字串去

返回值：

- 拷貝成功：dest字符串的首地址
- 拷貝失敗：NULL

使用範例：

```
char arr1[] = "hello word ";
char arr2[100];

// 參數：目標字符串 原始字符串
strcpy(arr2, arr1);

printf("%s\n", arr2);
// 輸出 hello word
```

注意：

如果dest所指的空間不夠大，可能會造成overflow的錯誤。

(七) strncpy() 函數

函數定義：

```
char *strncpy(char *dest, const char *src, size_t n);
```

參數：

- dest：目的字串
- src：原來的字串
- n：拷貝的個數

功能說明：

把src的字串前n個字串複製到dest所指向的空間中，由指定的長度是否包含\0決定複製是否結束。

白話文說明：

拷貝A字串的前n個字串到B字串去

返回值：

- 拷貝成功：dest字串的首地址
- 拷貝失敗：NULL

使用範例：

```
char arr1[] = "hello word ";
char arr2[100];
// 參數：目標字符串 原始字符串 字符長度
strncpy(arr2, arr1, 5);
printf("%s\n", arr2); // 輸出hello亂碼

// 注意：有限複製不會將\0複製到目標字符串中，須自己手動加才會停止

// 改善方法：

strncpy(arr2, arr1, 5);
arr2[5] = '\0';
printf("%s\n", arr2); // 輸出hello
```

下集預告

因為字串函數太多了QQ，一篇會講不完，所以剩下的會放到下一篇文章，strncat, strcmp, strncmp ...等，希望這些對你有幫助～

我是Andy，謝謝你看完這篇文章，如果文章有幫助到你的話，希望不吝於幫我拍手 🙌🙌🙌

[C](#)[字串函數](#)[程式自學](#)[Ccpp](#)

2 claps



WRITTEN BY

Andy Cheng[Follow](#)**Andy的趣味程式練功坊**

基礎程式 / 專案 / 技術分享 / 前端 / 後端 / App

[Follow](#)[Write the first response](#)

More From Medium

Related reads

Predicting Pokemon Battle Winner using Machine Learning

Saurabh Charde in AI Enigma



215



Related reads

Classify Passenger Jets Using PyTorch

Linping Yu



149



Related reads

Python Deep Learning: Part 1

Jon C-137

Sep 25, 2018 · 5 min read ★



238



Discover Medium

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. [Watch](#)

Make Medium yours

Follow all the topics you care about, and we'll deliver the best stories for you to your homepage and inbox. [Explore](#)

Become a member

Get unlimited access to the best stories on Medium — and support writers while you're at it. Just \$5/month. [Upgrade](#)

Medium

[About](#)

[Help](#)

[Legal](#)