

Stringhe

Stefano Ferrari

Università degli Studi di Milano
stefano.ferrari@unimi.it

Programmazione

anno accademico 2016–2017

Stringhe

In C, le stringhe sono rappresentate come

- ▶ vettori di caratteri (`char s[N+1];`)
- ▶ terminati dal carattere *null* (`'\0'`), detto *terminatore*, il quale ha *codifica binaria interamente nulla*

Il terminatore è implicito nelle stringhe costanti:

"abc" è un vettore di 4 caratteri: `'a'`, `'b'`, `'c'`, `'\0'`

Se la stringa `s` ha valore "abc:"

- | | |
|---|--|
| ▶ <code>s[0]</code> vale <code>'a'</code> | ▶ <code>s[2]</code> vale <code>'c'</code> |
| ▶ <code>s[1]</code> vale <code>'b'</code> | ▶ <code>s[3]</code> vale <code>'\0'</code> |

Terminatore

Una stringa si dichiara come un vettore di caratteri

```
#define LUNGHEZZA 6  
char s[LUNGHEZZA+1];
```

Una stringa termina col primo `'\0'` che contiene, anche se dichiarata di lunghezza superiore: un vettore di `LUNGHEZZA+1` caratteri può rappresentare stringhe da 0 a `LUNGHEZZA` caratteri

'p'	'r'	'o'	'\0'	'v'	'a'	'\0'
0	1	2	3	4	5	6

 vale `''pro''`

In C non c'è controllo che una stringa contenga il carattere `'\0'`:
ciò può causare errori (il vettore non è usabile come stringa!)

Stringhe vuote o quasi

La stringa vuota (`""`)

- ▶ contiene un solo carattere `'\0'`
- ▶ nella posizione di indice 0

Un carattere (`'a'`) e una stringa contenente un carattere (`"a"`)
fisicamente sono cose ben diverse

- ▶ la stringa contiene due caratteri (il secondo è `'\0'`)
- ▶ il carattere è convertibile in un numero, la stringa no

Funzioni (1)

La libreria `string.h` fornisce molte funzioni per gestire stringhe

Per poterle usare bisogna includere la libreria: `#include`

`<string.h>`

La funzione

`strlen(s)`

fornisce la lunghezza di una stringa (numero di caratteri tranne `'\0'`)

Non si possono copiare stringhe con l'operatore di assegnamento (`=`)

La funzione

`strcpy(dest,orig)`

copia la stringa *orig* nella stringa *dest*

- il C non controlla che la stringa *dest* possa contenere *orig*: se *dest* è più corta, la copia eccede i limiti e sporca altri dati

La funzione `strncpy(dest,orig,n)` copia al max. i primi *n* caratteri

Funzioni (2)

La funzione

`strcat(dest,add)`

aggiunge la stringa *add* in fondo alla stringa *dest*

- il C non controlla che la stringa *dest* possa allungarsi per l'intera aggiunta di *add*: se *dest* è troppo corta, l'aggiunta eccede i limiti e sporca altri dati

La funzione `strncat(dest,add,n)` aggiunge al max. *n* caratteri

Funzioni (3)

Non si possono confrontare stringhe con l'operatore ==

La funzione

`strcmp(s1,s2)`

confronta le stringhe *s1* e *s2* in modo lessicografico (dizionario)

1. **scorre** in parallelo **le due stringhe** per $i \geq 0$ caratteri **fino a** trovare
 - ▶ **due caratteri diversi** ($s1[i] \neq s2[i]$)
 - ▶ oppure **il termine** di una delle due stringhe
2. **restituisce**
 - ▶ **valore nullo** se **entrambe** le stringhe sono terminate
 - ▶ **valore negativo** se *s1* termina o $s1[i] < s2[i]$
 - ▶ **valore positivo** se *s2* termina o $s1[i] > s2[i]$