Stringhe in C

Alessandra Giordani
agiordani@disi.unitn.it
Lunedì 23 aprile 2012
http://disi.unitn.it/~agiordani/

Stringhe

- Sono particolari array:
  - Ogni elemento del vettore è di tipo char
  - La stringa è terminata dal carattere speciale `'\0'`
- Quindi è una sequenza di caratteri delimitata dal terminatore di stringa e inclusa tra doppi apici (").
- Ad esempio:
  - `“ciao” é ‘c’ ‘i’ ‘a’ ‘o’ ‘\0’`
  - `“ciao mondo"`
  - `“questa è una stringa”`

Sequenze di Escape

- Alcuni caratteri non sono rappresentabili con un singolo carattere (es. carattere di fine riga)
- Delle sequenze speciali (escape sequences) ci consentono di inserire questi caratteri.
- Ad esempio, il carattere di fina riga puo essere rappresentato con `'\n'` (non `"\n"`!)
- Quindi, la stringa "questa e una stringa\n" termina con un carattere di fine riga.
- In ogni caso una stringa termina sempre con il carattere di fine stringa `'\0'` (non `"\0n"`!)

Dichiarazione e inizializzazione

```
char str1[6];
static char str2[]="testo";
```

- può essere inizializzato con una stringa costante
- dimensione array > lunghezza stringa + 1
  - per memorizzare il carattere nullo di terminazione (str2 ha lunghezza 6 byte)
- Cosa succede se si dichiariamo la lunghezza della stringa n, e si inizializza con una stringa costante di lunghezza >= n?
- `static char str3[3]="tre";`
- `static char str4[3]="quattro";`

Input di stringhe

- Queste considerazioni sono particolarmente delicate nel caso in cui vogliamo usare scanf per leggere stringhe
- La sequenza %s all'interno di un template consuma qualunque sequenza di caratteri che non contenga spazi
- Per leggere una stringa, dobbiamo prima avere allocato un buffer (cioe un array di caratteri) abbastanza capiente. Ad esempio:

```
char buf [256]; // crea un buffer
scanf("%s", buf);
```

- Se la stringa immessa dall'utente è piu lunga del buffer (nell'esempio la lunghezza massima è 255, tenendo conto del null character '\0') abbiamo un errore di **segmentation fault** (violazione di accesso in memoria)

printf() di stringhe

- printf può anche essere usata per stampare il contenuto di una variabile un array di caratteri o parte di esso
- All'interno della stringa template dobbiamo usare %s.
- Come altro parametro dobbiamo passare il nome della variabile, che usato da solo è il puntatore al primo elemento della stringa
- printf() stamperà quel elemento fino al carattere `'\0'`
  - `char str[] = "stringa";`
  - `printf("%s",str); //stampa` **`stringa`**
  - `printf("%s",str+2); // stampa` **`ringa`**
  - `printf("%s",str[2]); // non corretto!`
  - `printf("%c",str[2]); // stampa` **`r`**

scanf() di stringhe

- Allo stesso modo per leggere una stringa non carattere per carattere ma per intero
  - template “%s”
  - indirizzo del primo elemento della stringa che è proprio il nome della stringa! (no &)

Inversione di una stringa

- Leggo una parola *a* di lunghezza sconosciuta (dim max 20)
- Inserisco in *b* i caratteri di *a* in ordine inverso
- Stampo stringhe

```c
#include <stdio.h>
#define MAX 20
int main()
{
    char a[MAX], b[MAX];
    int i, dim;
    /*leggo parola*/
    printf("Inserire una parola (dim.max %d): ",MAX-1);
        scanf("%s", a);
    /*stampo parola - e calcolo lunghezza*/
    printf("Ho letto: '");
    for(i=0;a[i]!='\0';i++)
        printf("%c", a[i]);
    dim=i;
    printf("' lunga %d caratteri\n",dim);
    /*inverto caratteri*/
    for(i=0;i<dim;i++)
        b[i]=a[dim-1-i];
    b[dim]='\0';
    /*stampo parola invertita*/
    printf("Ecco la parola invertita '%s'\n",b);
    return 0;
}
```

Inversione di una frase?!

- Se proviamo ad immettere una stringa con spazi, quando si incontra uno spazio la stringa di input si considera finita…

scanf() di stringhe con spazi

- Nel template possiamo specificare…
  - quanti caratteri leggere:
    - “%20s” leggere massimo 20 caratteri (no spazi)
  - quali carattere accettare
    - “%[A-Za-z]” accettare solo caratteri alfabetici maiuscolo o minuscolo e lo spazio
    - “%[0-9-]” accettare solo caratteri numerici e il -

Buffer di lettura - accorgimenti

- Ogni volta che si legge una stringa devo aver preallocata un buffer sufficiente
- Se leggo una stringa di massimo 20 char devo aver allocato un buffer di 21 char (e impedire che ne vengano letti di più!)
  - `char string[21];`
  - `scanf("%20s", string);`
    - oppure
  - `scanf("%20[ a-b0-9]", string);`

EsErCiZiO PaRi DiSpArI

- Scrivere un programma che legge una frase e rende maiuscoli tutti i caratteri pari e minuscoli quelli dispari
- Per sapere se un numero è pari o dispari controlliamo il resto della divisione per 2
  - 3%2 = 1
  - 4%2 = 0
- Oppure sfruttiamo la divisione intera
  - 3/2*2 ≠ 3
  - 4/2*2 ≠ 4

EsErCiZiO PaRi DiSpArI

- Possiamo procedere in più modi
  - Rendere tutto minuscolo e poi solo cambiare in maiuscolo i pari (o contrario)
  - Rendere tutti i dispari minuscoli e tutti i pari maiuscoli
    - 2 cicli separati (da i=0 e da i=1 con i=i+2)
    - 1 ciclo che considera coppie i e i+1 a due a due

Soluzione 1

```c
#include <stdio.h>
#define MAX 20
int main()
{
    char a[MAX];
    int i, dim;
    printf("Inserire una parola: ");
    scanf("%s", a);
    for(i=0;a[i]!='\0';i++)
        a[i]=toupper(a[i]);
    dim=i;
    for(i=1;i<dim;i=i+2)
        a[i]=tolower(a[i]);
    printf("Ecco la parola cambiata '%s'\n",a);
    return 0;
}
```

Soluzione 2

```c
#include <stdio.h>
#define MAX 20
int main()
{
    char a[MAX];
    int i, dim;
    printf("Inserire una parola: ");
    scanf("%s", a);
    for(i=0;a[i]!='\0';i++) ; //attenzione a questo ;
    dim=i;
    for(i=0;i<dim;i=i+2)
        a[i]=toupper(a[i]);
    for(i=1;i<dim;i=i+2)
        a[i]=tolower(a[i]);
    printf("Ecco la parola cambiata '%s'\n",a);
    return 0;
}
```

Soluzione 3

```c
#include <stdio.h>
#define MAX 20
int main()
{
    char a[MAX];
    int i, dim;
    printf("Inserire una parola: ");
    scanf("%s", a);
    for(i=0;a[i]!='\0';i++)
      if (i%2==0)
        a[i]=toupper(a[i]);
      else
        a[i]=tolower(a[i]);
    printf("Ecco la parola cambiata '%s'\n",a);
    return 0;
}
```

Parole palindrome

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

main()
{
      char parola[32], i=0, n;

      printf("Inserisci una parola (lunga al max 31 caratteri): ");
      scanf("%31s", parola);
      n = strlen(parola);
      while((i <= n/2) && (parola[i] == parola[n-1-i]))
       i++;
      if(i > n/2)
            printf("La parola %s e' palindroma.\n", parola);
      else
       printf("La parola %s non e' palindroma.\n", parola);
      exit(0);
}
```

Es. ANNA, ONORARONO, …

Frasi palindrome (a meno di spazi)

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

main()
{
      char parola[52], app[52], i = 0, j = 0, n;
      printf("Inserisci una frase (lunga al max 51 caratteri): ");
      scanf("%51[A-Za-z ']", parola);
      while(parola[i]!='\0')
      {
          if ((parola[i]!=' ')&&(parola[i]!='\''))
          {
             app[j]=parola[i];
             j++;
          }
          i++;
      }
      app[j]='\0';
      n = j;
      i = 0;
      while((i <= n/2) && (app[i] == app[n-1-i]))
       i++;
      if(i > n/2)
           printf("La frase %s e' palindroma.\n", parola);
      else
           printf("La frase %s non e' palindroma.\n", parola);
      exit(0);
}
```

Es. i topi non avevano nipoti, ai lati d italia…

Conversione da stringa ad intero

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

main()
{
     char anno_nascita[5], anno_corrente[5];
     int anni;

      printf("Inserire l'anno di nascita: ");
     scanf("%s", anno_nascita);
     printf("Inserire l'anno corrente: ");
     scanf("%s", anno_corrente);

     /* atoi() converte una stringa in un intero */
     anni = atoi(anno_corrente) - atoi(anno_nascita);
     printf("Eta': %d\n", anni);
     exit(0);
}
```

Stringa MAIUSCOLA

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

main()
{
          char s[100], t[100];
          int i;

          printf("Inserisci una stringa: ");
          scanf("%[A-Za-z ]", s);
          for (i=0; i<strlen(s); i++)
          {
               if ((s[i] >= 'a') && (s[i] <= 'z'))
                  t[i] = s[i] - 32;
               else
                  t[i] = s[i];
          }
          t[i]='\0';
          printf("Stringa maiuscola: %s\n", t);
          exit(0);
}
```