

ISTITUTO SECONDARIO DI ISTRUZIONE SUPERIORE  
"Guido Tassinari" di POZZUOLI

**Informatica**

**Prof. F. F. F.**

**A.S. 2011/2012**

## **Linguaggio C - Stringhe**

27/04/2012



# Array di caratteri

- Una stringa è:
  - Un array di caratteri
  - Termina con il carattere '\0' (carattere terminatore)
- Essendo un array di caratteri la sua dichiarazione è del tipo

```
char miastringa [20];
```

- `miastringa` è il nome della variabile che contiene la stringa
- La lunghezza MASSIMA della stringa utilizzabile per i caratteri è 19 caratteri (un carattere deve essere riservato obbligatoriamente al carattere terminatore)
- La lunghezza effettiva di una stringa dipende dalla posizione del carattere terminatore

# Inizializzazione di una stringa

- Una variabile di tipo stringa è l'unico tipo di array che è possibile inizializzare esplicitamente in blocco
- Il valore della stringa deve essere racchiuso tra doppi apici e può essere assegnato nella dichiarazione
- I singoli apici indicano il singolo carattere

```
char miastringa [20] = "la mia stringa";
```

- Il carattere terminatore viene automaticamente inserito dal linguaggio C

```
:" l   a   m   i   a   s   t   r   i   n   g   a   \0
 0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16  17  18  19
```

# Accesso ai caratteri

Essendo una stringa un array di caratteri posso accedere ai suoi elementi anche singolarmente

```
miastringa[0]='l';
```

```
::
```

```
l a m i a s t r i n g a \0 4
```

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

- Devo accertarmi che io acceda ai soli caratteri precedenti al terminatore
- I caratteri inseriti dopo il terminatore non generano errore ma sono semplicemente ignorati

```
miastringa[15]='4';
```

```
l a m i a s t r i n g a \0 4
```

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

# Lunghezza di una stringa

- Per calcolare la lunghezza della stringa basta scandire l'array e cercare il carattere terminatore

```
int main() {
    int i;

    i=0;
    while (miastringa[i] != '\0') {
        i++;
    }
    printf("La lunghezza è %d", i);
}
```

- La dimensione della variabile `miastringa` è sempre di 20 caratteri ed è fissato a tempo di compilazione come tutti gli array (struttura dati a carattere STATICO)
- È il contenuto della variabile `miastringa` che può variare anche a tempo di esecuzione ma non può essere mai superiore a 19

# Copia di una stringa

- Come per tutti gli array, date due stringhe  $s1$  ed  $s2$  con DIM elementi la seguente assegnazione in C non produce la copia della stringa  $s2$  in  $s1$

`a = b;` (N.B. istruzione errata)

- È necessaria la copia della stringa  $s2$  in  $s1$

elemento per elemento

```
for (i = 0; i < DIM; i++)  
    a[i] = b[i];
```

# Libreria string.h

- In C è presente una libreria per la manipolazione delle stringhe
- Nella libreria sono definite alcune funzioni utili alla gestione delle stringhe
- Noi analizzeremo:
  - `strlen(char stringa[])`: restituisce la lunghezza di una stringa
  - `strcpy(char stringa1[], char stringa2[])`: copia `stringa1` in `stringa2`
  - `strcat(char stringa1[], char stringa2[])`: accoda a `stringa1` il contenuto di `stringa2`
  - `strcmp(char stringa1[], char stringa2[])`: confronta `stringa1` e `stringa2`
- Per poter utilizzare le funzioni la libreria deve essere opportunamente inclusa nella sezione dichiarativa globale

```
#include <string.h>
```

# strlen

- Restituisce la lunghezza della stringa

```
int main () {  
  
    char s[10]="pippo";  
    int l;  
  
    l = strlen(p);  
    printf("La lunghezza della stringa è: %d", l);  
  
    return 0;  
}
```

- A destra dell'operatore di assegnazione c'è l'invocazione ad una funzione
- Indica che il risultato dell' funzione viene inserito nella variabile a sinistra dell'operatore di assegnazione



# strcpy

- Copia il contenuto della stringa passata come secondo parametro nella stringa passata come primo parametro

```
int main () {  
  
    char sorgente[10]="pippo";  
    char destinazione[10];  
    int l;  
  
    strcpy(destinazione, sorgente);  
    printf("Il valore di destinazione è: %s",  
          destinazione);  
  
    return 0;  
}
```

- L'eventuale contenuto di `destinazione` precedente all'assegnamento viene perso

# strcat

- Accoda il contenuto della stringa passata come secondo parametro nella stringa passata come primo parametro

```
int main () {
```

```
    char sorgente[10]="mamma";
```

```
    char destinazione[10]="ciao";
```

```
    int l;
```

```
    strcpy(destinazione, sorgente);
```

```
    printf("Il valore di destinazione è: %s",  
          destinazione);
```

```
    return 0;
```

```
}
```

- Devo essere certo che la dimensione massima della stringa `destinazione` sia sufficiente ad ospitare la nuova stringa

# strcmp 1/2

- Confronta il contenuto di due stringhe
- Restituisce:
  - 0 se le stringhe sono identiche
  - <0 se la stringa passata come primo parametro è minore della stringa passata come secondo parametro
  - >0 se la stringa passata come primo parametro è maggiore della stringa passata come secondo parametro
- La relazione d'ordine tra stringhe è definita dalla relazione d'ordine della codifica ASCII dei caratteri che la compongono
  - '0' < '9' < 'A' < 'Z' < 'a' < 'z'

## strcmp 2/2

```
int main () {  
  
    char s1[10]="mamma";  
    char s2[10]="ciao";  
    int c;  
  
    c = strcmp(s1, s2);  
    if (c==0) {  
        printf("le due stringhe sono uguali");  
    } elseif (c<0) {  
        printf("s1 < s2");  
    } else {  
        printf("s1 > s2");  
    }  
  
    return 0;  
}
```