Introduzione al C
Lez. 5

Stringhe

Stringhe

Una stringa è una sequenza di caratteri. Ad esempio una parola, una frase, un testo...

In C non è previsto un tipo per le stringhe.
Una stringa è vista come un array di caratteri che, per convenzione, termina con il simbolo speciale `'\0'`.

Si usa

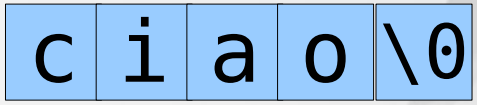
```
char s[N+1];
```

per memorizzare una stringa di N caratteri.

Stringhe

Le costanti stringa vengono rapresentate tra virgolette.

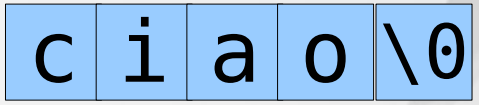
Esempio
“`ciao`” è un array di caratteri di dimensione 5.



Stringhe

Le costanti stringa vengono rapresentate tra virgolette.

Esempio
“ciao” è un array di caratteri di dimensione 5.



Una costante stringa viene trattata come un puntatore al primo carattere della stringa.

Esempio

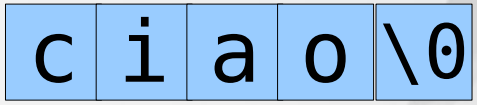
```
char *s = “ciao”; // anche char s[] = “ciao”; s qui è costante
print(“%s %s\n”, s, s+1);// stampa ciascun carattere fino a '\0'
print(“%c %c\n”, s[0], s[1]);
```

Cosa stampano?

Stringhe

Le costanti stringa vengono rapresentate tra virgolette.

Esempio
“ciao” è un array di caratteri di dimensione 5.



Una costante stringa viene trattata come un puntatore al primo carattere della stringa.

Esempio

```
char *s = “ciao”; // anche char s[] = “ciao”; s qui è costante
printf(“%s %s\n”, s, s+1);// stampa ciascun carattere fino a '\0'
printf(“%c %c\n”, s[0], s[1]);
```

Cosa stampano?

```
ciao iao
c i
```

Nota: la `stdlib` contiene molte funzioni per gestire stringhe.

Stringhe

Esempio

```
void my_printf(char *s) {

   int i;
   for(i=0; s[i] != '\0'; i++) /* s[i]!= '\0' <--> s[i] */
      printf(“%c”, s[i]);


}

int main() {

   char s[101]; // stringhe fino a 100 caratteri
   scanf(“%s”, s);
   my_printf(s);
   return 0;

}
```

Stringhe

Esempio

```
void my_printf(char *s) {

   int i = 0;
   while(s[i])
      printf(“%c”, s[i++]);

}

int main() {

   char s[101]; // stringhe fino a 100 caratteri
   scanf(“%s”, s);
   my_printf(s);
   return 0;

}
```

Stringhe

Ovviamente potete allocare dinamicamente memoria per contenere una stringa.

```
char * s = malloc((N+1)*sizeof(char));
if(!s) exit(1);
```

Esercizio 1

Scrivere la funzione

```
int my_strlen(char *s)
```

che restituisce il numero di caratteri della stringa `s`.

Scrivere un programma che provi questa funzione leggendo una stringa da tastiera. Si può assumere che la stringa in input contenga non più di 1000 caratteri.

Strumenti utili:
Si può leggere una stringa da tastiera con

```
char str[1001];
scanf(“%s”, str);
```

Esercizio 1

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

int my_strlen(char *s){

 int len;
 for (len=0; s[len]; len++);
 return len;

}

int main(){

 char str[1001];
 scanf("%s", str);
 printf("%d\n",my_strlen(str));
 return 0;

}
```

Esercizio 2

Scrivere la funzione

```
char * my_strcat(char *s1, char *s2)
```

che restituisce un puntatore alla nuova stringa ottenuta concatenando le stringhe puntate da `s1` e `s2`.

Scrivere un programma che legga due stringhe da tastiera e stampi la concatenazione delle due.

Si può assumere che le stringhe in input contengano non più di 1000 caratteri.

Esercizio 2

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

char * my_strcat(char *s1, char *s2){

 int len = strlen(s1) + strlen(s2);
 char *cat = malloc(len+1);
 int i, j;
 j=0;

 /* copia prima stringa */
 for (i=0; s1[i]; i++)
   cat[j++]=s1[i];

 /* copia seconda stringa */
 for (i=0; s2[i]; i++)
   cat[j++]=s2[i];

  cat[len]='\0';
  return cat;
}
```

Esercizio 2

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

char * my_strcat(char *s1, char *s2){

 int len = strlen(s1) + strlen(s2);
 char *cat = malloc(len+1);
 int i, j;
 j=0;

 /* copia prima stringa */
 for (i=0; s1[i]; i++)
   cat[j++]=s1[i];

 /* copia seconda stringa */
 for (i=0; s2[i]; i++)
   cat[j++]=s2[i];

  cat[len]='\0';
  return cat;
}
```

```
int main(){

 char s1[1001];
 char s2[1001];
 scanf("%s", s1);
 scanf("%s", s2);
 char*cat= my_strcat(s1,s2);
 printf("%s", cat);

}
```

Esercizio 3

Scrivere la funzione:

```
int anagramma(char *s1, char *s2);
```

che restituisca 1 se le stringhe puntate da s1 e s2 sono una l'anagramma dell'altro e 0 altrimenti.

Es.:

anagramma(“pizza”,”pazzi”) == 1

Assumere che le stringhe siano composte da solo lettere minuscole a-z, aventi codifica intera 97-122.

Hint: Costruire un vettore di frequenze F di 26 posizioni tale che il numero di occorrenze del carattere c sia memorizzato in F[c-97]. Due stringhe sono anagrammi *se e solo se i due rispettivi vettori di frequenza contengono valori uguali*

Esercizio 4

Scrivere un programma che, ricevuto in input un intero n, genera un array A casuale di 4*n unsigned char (quindi interi in [0,255]).

Quindi calcola il massimo dei valori contenuti in A nell'ipotesi di interpretare il suo contenuto come array di 4*n char, 2*n short, e n int, e stampa i tre risultati così ottenuti.