Fondamenti di Informatica
Ingegneria Clinica
Lezione speciale

Raffaele Nicolussi

FUB - Fondazione Ugo Bordoni

Via del Policlinico, 147 – 00161 Roma

Libreria gestione caratteri (<ctype.h>)

- **int isdigit (intc):restituisce un valore ≠ 0 se c è una cifra (0-9), altrimenti torna 0**
- **int isalpha (intc): restituisce un valore ≠ 0 se c è una lettera dell'alfabeto, altrimenti torna 0**
- **int isalnum (intc): restituisce un valore ≠ 0 se c è una cifra (0-9) o una lettera dell'alfabeto (alfanumerici), altrimenti 0**
- **int isxdigit (intc): restituisce un valore ≠ 0 se c è una cifra esadecimale, altrimenti torna 0**
- **int islower(intc): restituisce un valore ≠ 0 se c è una lettera minuscola, altrimenti torna 0**
- **int isupper (intc): restituisce un valore ≠ 0 se c è una lettera maiuscola, altrimenti torna 0**
- **int tolower (intc): se c è una lettera restituisce il suo equivalente minuscolo, altrimenti il carattere inalterato**

- **int toupper (intc): se c è una lettera restituisce il suo equivalente maiuscolo, altrimenti il carattere inalterato**

- **int isspace (intc): restituisce un valore ≠ 0 se c è uno spazio ' ', un newline '\n', un salto pagine '\f' (form feed), un ritorno carrello '\r' (carriage return), una tabulazione orizzontale '\t' o verticale '\v', altrimenti torna 0**

- **int iscntrl (intc): restituisce un valore ≠ 0 se c è un carattere di controllo (compreso tra 0 e 0x1F(31) o 0x7F (127) Del) , altrimenti torna 0**

- **int ispunct (intc): restituisce un valore ≠ 0 se c è un carattere stampabile diverso dai caratteri alfanumerici e dallo spazio (è un segno di punteggiatura), altrimenti 0**

- **int isprint (intc): restituisce restituisce un valore ≠ 0 se c è un carattere stampabile considerando anche lo spazio, altrimenti 0**

- **int isgraph (intc): restituisce un valore ≠ 0 se c è un carattere stampabile diverso dallo spazio (generalmente i caratteri tra 0x21 e 0x7E (33-126)**

Funzioni per la conversione di stringhe (libreria <stdlib.h>)

- Convertono stringhe che contengono interi o float in numeri interi o float
- Permettono di trattare stringhe (testi) composte da caratteri qualsiasi e di estrarre da essi le parti numeriche

double atof (const char *nPtr)

- argomento: puntatore a costante di caratteri
- converte la stringa puntata da nPtr in un valore double e ne restituisce il risultato.
- la stringa deve contenere un numero in virgola mobile valido, altrimenti il valore che torna è indefinito
- il numero può essere concluso da qualsiasi carattere che non sia valido come numero in virgola mobile

int atoi (const char *nPtr)

- argomento: puntatore a costante di caratteri
- converte la stringa puntata da nPtr in un valore int e ne restituisce il risultato.
- la stringa deve contenere un numero intero valido, altrimenti il valore che torna è indefinito
- il numero può essere concluso da qualsiasi carattere che non sia valido come numero intero

```
#include <stdio.h>
#include <stdlib.h>
int main (void) {
    char *s1="9823boo!!";
    printf("L'intero inserito e' = %d\n",
    atoi(s1));
    return 0;
}
/* Soluzione
L'intero inserito e' = 9823
*/
```

atoi_p.c

long atol (const char *nPtr)

- argomento: puntatore a costante di caratteri
- converte la stringa puntata da nPtr in un valore long int e ne restituisce il risultato.
- la stringa deve contenere un numero long valido, altrimenti il valore che torna è indefinito
- il numero può essere concluso da qualsiasi carattere che non sia valido come numero intero

double strtod (const char *nPtr, char **endPtr)

- due argomenti: puntatore a costante di caratteri e indirizzo
- converte la stringa puntata da nPtr (eliminando eventuali spazi vuoti iniziali) in un valore double e ne restituisce il risultato.
- endPtr viene assegnato all'eventuale parte rimanente della stringa originale
- se la stringa non punta ad un numero, viene tornato 0

long strtol (const char *nPtr, char **endPtr, int base)

- tre argomenti: puntatore a costante di caratteri, un indirizzo, un intero
- converte la stringa puntata da nPtr (eliminando eventuali spazi vuoti iniziali) in un valore long int e ne restituisce il risultato.
- endPtr viene assegnato all'eventuale parte rimanente della stringa originale
- se la base è uguale a 0, la conversione è fatta nella base determinata dalle regole di conversione che governano la specifica delle costanti
- se la base è diversa da 0, deve essere compresa tra 2 e 36 e specifica la base di conversione.
- se la stringa non punta ad un numero, viene tornato 0

unsigned long strtoul (const char *nPtr, char **endPtr, int base)

- tre argomenti: puntatore a costante di caratteri, un indirizzo, un intero
- converte la stringa puntata da nPtr (eliminando eventuali spazi vuoti iniziali) in un valore unsigned long int e ne restituisce il risultato.
- endPtr viene assegnato all'eventuale parte rimanente della stringa originale
- se la base è uguale a 0, la conversione è fatta nella base determinata dalle regole di conversione che governano la specifica delle costanti
- se la base è diversa da 0, deve essere compresa tra 2 e 36 e specifica la base di conversione.
- se la stringa non punta ad un numero, viene tornato 0

Funzioni per la manipolazione delle stringhe

Si deve includere **<string.h>**

- sono funzioni che permettono di manipolare, confrontare, concatenare, copiare, etc. stringhe
- ne vedremo solo alcune

char *strcpy (char *s1, const char *s2)

- copia il contenuto dell'array puntato da s2 nell'array puntato da s1
- restituisce s1
- la dimensione di s1 deve essere tale da contenere s2 e \0

```
void strcpy (char s1[ ], char s2[ ])
{   int i;
    for (i=0; s2[i]; ++i)
            s1[i] = s2[i];
    s1[++i] = 0;            }
```

```
char str[30];
strcpy (str, "salve”);
puts(str);

char str1[50], str2[] = “salve!!”;
strcpy (str1, str2);
puts(str1);
```

char *strncpy (char *s1,const char *s2,size_t n)

- **copia fino a n caratteri dell'array puntato da s2 nell'array puntato da s1**
- **restituisce s1**
- **la dimensione di s1 deve essere tale da contenere gli n caratteri di s2 e il carattere di terminazione**
- **se s2 contiene meno di n caratteri, ad s1 verranno aggiunti dei caratteri nulli, fino al raggiungimento di n**
- **se s2 contiene più di n caratteri, la stringa s1 risultante non conterrà il carattere di terminazione**

```
char str1[80], str2[80];
gets(str2);
strncpy(str1, str2, 79);
```

char *strcat (char *s1, const char *s2)

- concatena una copia di s2 a s1 e chiude s1 con il carattere nullo
- il codice di terminazione che chiudeva s1 viene sostituito dal primo carattere di s2
- s2 non viene modificata dall'operazione
- si ricordi che il C non controlla le dimensioni dei vettori, quindi s1 deve avere una dimensione tale da contenere s2

```c
#include <stdio.h>
#include <string.h>
int main (void)
{
    char s1[80], s2[80];
    /* acquisizione stringhe */
    gets(s1);
    gets(s2);
    /* concatenazione */
    strcat (s2, s1);
    printf(s2);
    return 0;

}
```

bollo
franco
francobollo

strcat.c

char *strncat (char *s1,const char *s2, size_t n)

- concatena non più di n caratteri presi da s2 a s1 e chiude s1 con il carattere nullo
- il codice di terminazione che chiudeva s1 viene sostituito dal primo carattere di s2
- s2 non viene modificata dall'operazione
- si ricordi che il C non controlla le dimensioni dei vettori, quindi s1 deve avere una dimensione tale da contenere la parte di s2 che si vuole concatenare

```
#include <stdio.h>
#include <string.h>
int main (void)
{   char s1[80], s2[80], *p;
    int i=0;
    gets(s1);
    /* acquisizione stringhe */
    gets(s2);
    p = s2;
    /*conteggio caratteri */
    while (*p != 'd')   {
                    ++i;
                    ++p;   }
    strncat (s1, s2, i);
    printf("stringa risultante: %s\n", s1);
    return 0;  }
```

strncat.c

```
/*        guarda
          come dondolo
         stringa risultante: guardacome     */
```

Confronto tra stringhe

- **Definite nella libreria <string.h>**
- **Dato che i caratteri sono rappresentate da codici numerici che danno loro un ordinamento, è possibile decidere se una stringa è maggiore, minore o uguale ad un'altra.**
- **Miles Davis è maggiore di Miles David**
- **Sting è minore di String**
- **Casa è uguale a Casa**
- **Casa è minore di casa**

char *strcmp (const char *s1, const char *s2)

- **Confronta carattere per carattere le stringhe s1 e s2**
- **Le stringhe non sono modificate**
- **Restituisce un intero che si basa sul risultato del confronto**

Valore tornato	Significato
< 0	s1 è minore di
0	s1 è uguale a s2
> 0	s1 è maggiore di s2

char *strncmp (const char *s1, const char *s2, size_t n)

- **Confronta un massimo di caratteri di s1 con la stringa s2**
- **Le stringhe non sono modificate**
- **Restituisce un intero che si basa sul risultato del confronto**

Valore tornato	Significato
< 0	s1 è minore di s2
0	s1 è uguale a s2
> 0	s1 è maggiore di s2