Le stringhe

Il linguaggio C++ non dispone di un tipo di dati specifico per rappresentare una stringa. Per identificare quest'ultima, quindi, sarà necessario definire un array monodimensionale di caratteri. La dichiarazione di una variabile di tipo stringa avverrà pertanto in base alla seguente sintassi:

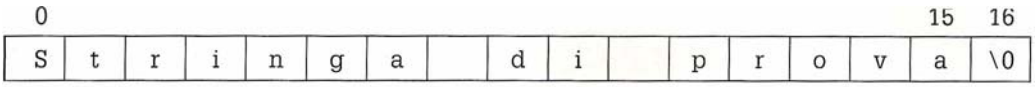
```
char <Nome Stringa> [ ];
```

Ad esempio, la seguente dichiarazione:

```
char Stringa[ ] = "Stringa di prova";
```

ha l'effetto di creare una variabile (array di caratteri) di nome *Stringa* di lunghezza pari alla lunghezza della costante stringa *"Stringa di prova"*, e cioè 16 caratteri più il carattere speciale "\0", che è come un punto alla fine di una frase, ossia non viene contato come una lettera, però occupa uno spazio. In totale avremo una stringa di 17 caratteri. Ricapitolando, per dichiarare una stringa di *N* elementi occorrerà creare un array di *N + 1* caratteri. La precedente dichiarazione corrisponde quindi alla seguente:

```
char Stringa[17]
```



Il carattere speciale "\0" prende il nome di **terminatore** di una stringa e corrisponde alla configurazione di 8 zeri in un byte.
Il linguaggio C++ prevede più metodi per inizializzare una stringa; vediamone due:

```
char Stringa 1 [6];
char Stringa2[10];
char Stringa3[5] = "casa";
```

```
//Inizializzazione della stringa Stringa 1 a mo' di vettore
Stringal[O] = 'p';
Stringal[l] = 'i';
Stringal[2] = 'a';
StiTngal[3] = 'n';
Stringal[4] = 'o';
Stringal[5] = '\0';
// Inizializzazione della stringa Stringa2 da input dell'utente
 cout « "Inserire una stringa di dieci caratteri: " « endl;
cin.getline(Stringa2, 10);
```

Le stringhe vanno sempre racchiuse tra doppi apici""i caratteri vanno tra apici singoli ‘ ':
le due seguenti sono errate

```
char Carattere = "c";
char Stringa[8] = 'Ciao';
```

È possibile accedere ai singoli caratteri che compongono una stringa per modificarli. Se ad esempio si volesse modificare la stringa *Stringa*, il cui contenuto supponiamo sia ora "Ciao" in una stringa costituita da tutti caratteri "x" si potrebbe impostare un ciclo *for* del tipo:

```
for (int J = O; J < 7; J++)
    Stringa[ J] = 'x';
```

Il C++ non dispone di operazioni predefinite sulle stringhe. Ad esempio, non consente di utilizzare l'operatore = per assegnare una stringa a un'altra. Tutte le elaborazioni dovranno pertanto essere realizzate seguendo le regole analizzate per i vettori. A tal proposito, se si volesse copiare la stringa *Str1* che contiene "Ciao a tutti" all'interno della stringa *Str2* occorrerebbe prevedere un apposito ciclo *while* finchè non incontra \0, ma ciò si può evitare ricorrendo alle funzioni messe a disposizione dalla classe stringa.

Riprendiamo l'inizializzazione di una stringa tramite la lettura da tastiera. Osserviamo il seguente esempio

```
#include <iostream.h>

int main( ) {
    char S[50];         //dichiarazione di una variabile stringa di 50 caratteri
    cout « "Inserire la stringa ";
    cin » S;    // viene letta la stringa proveniente dalla tastiera
    cout « "\nLa stringa inserita e' la seguente: "« S « endl;
    }
```

Se alla stringa S viene assegnata da tastiera la frase "Ciao mondo" la successiva istruzione *cout* visualizzerà soltanto "Ciao". Questo perché il flusso di input *cin* riconosce la fine della stringa in input quando incontra un carattere qualsiasi (tra spazio, tabulazione o ritorno a capo). Per ovviare a questo inconveniente le due istruzioni evidenziate possono essere sostituite con le funzioni *printf()* e *getsf().* La funzione printf(<Stringa>) visualizza sul video la <Stringa> fornita come parametro, mentre la funzione gets(<VariabileSringa>) esegue l'input di una stringa dalla tastiera. *Gets()* è sostanzialmente una funzione che legge una stringa e si ferma al primo a capo, ad esempio quando un utente preme *Invio.*

Per utilizzare queste funzioni è necessario includere all'interno del programma l'header ***stdio.h.*** Effettuando la sostituzione, il nostro programma diviene il seguente

```
#include <iostream.h>
#include <stdio.h>
int main( )
{
    char S[50];         Ildichiarazione di una variabile stringa di 50 caratteri
    printf("Inserisci la stringa: ");       / / viene visualizzato il messaggio sul video
    gets(S);        / / viene letta la stringa proveniente dalla tastiera
    cout «"\nLa stringa inserita e' la seguente: "« S « endl;
    }
```

È doveroso ricordare che la funzione *gets( )* può essere utilizzata anche in associazione con *cout* e non necessariamente con la funzione *printf.*

La gestione delle stringhe è agevolata da alcune funzioni contenute nella libreria string.h. Pertanto, quando occorre utilizzare tali funzioni è necessario includere nel programma la direttiva *#include <string.h>*. Le funzioni più utilizzate sono:

Strcpy(<Str1> ,<Str2>);	Copia la stringa <Str2> all'interno di <Str1>.
Strcpy(<Strl>, <Str2>,<N>);	Copia all'interno della stringa <Str1> i primi N caratteri della stringa <Str2>
Strcat(<Str1>, <Str2>);	Concatena la stringa <Str2> alla fine di <Str1>.
Strlen(<Str>)	Conta semplicemente il numero dei caratteri nella stringa specificata. Restituisce, pertanto, la lunghezza della stringa <Str>.
Strcmp(<Str1>, <Str2>);	Confronta, carattere per carattere, le due stringhe fornite come parametri e restituisce il valore intero zero se le due stringhe sono uguali, un valore intero minore di zero se <Str1> è minore di <Str2> e maggiore di zero se <Str1> è maggiore di <Str2>
strcmpi(<Str1>, <Str2>);	Confronta, carattere per carattere, le due stringhe fornite come parametri e restituisce il valore intero zero se le due stringhe sono uguali, un valore intero minore di zero se <St1> è minore di <Str2> e maggiore di zero se <Str1> è maggiore di <Str2>. A differenza della precedente funzione, *strcmpi* non è *case sensitive.* Ricordiamo che questa funzione, come le due successive, non è standard ANSI e, come tale, potrebbe causare errori su alcuni compilatori.
strupr(<Stringa>);	Converte <Stringa> in maiuscolo. Restituisce anche una stringa, che sarà tutta in maiuscole. Se <Stringa> è un array, sarà anche esso trasformato tutto in maiuscolo.
strlwr(<Stringa>);	Converte <Stringa> in minuscolo. Restituisce anche una stringa, che sarà tutta in minuscole. Se <Stringa> è un array, sarà anche esso trasformato tutto in minuscolo
atoi(<String>);	converte una stringa in un intero , atof converte la stringa in double
itoa (<N> , <String> , <base>);	converte un numero N in una stringa nella base indicata tra 2,10,16