Le stringhe

Stringhe di caratteri (esempi di utilizzo dei vettori)

Nel linguaggio C++ una **stringa** è semplicemente un **vettore di caratteri**

- **Vettori di caratteri**
  - La stringa **"hello" è in realtà un vettore di** caratteri
  - ogni carattere occupa **un byte**
  - I vettori di caratteri possono essere inizializzati usando stringhe letterali

    char Saluto[ ] = "Hello";

H	e	l	l	o	\0

  - le stringhe terminano con il **carattere NULL '\0'** (0 della tabella ASCII)
  - Il vettore **Saluto ha in realtà 6 elementi**

    char Nome[ ] = {'M', 'a', 's', 's', 'i', 'm', 'o' };

  - Accesso ai singoli caratteri:

    Nome[1] è il carattere **'a'**

M	a	s	s	i	m	o	\0

  - Il nome del vettore è l'indirizzo del primo elemento del vettore stesso

Dichiarazione di una stringa in C++:

char nome_variabile[dimensione];

Segue le stesse regole di un vettore di tipo numerico, dove il tipo è **char.**

E' possibile **inizializzare** una stringa al momento della dichiarazione:

char Nazione[16] = "Italia";

I	t	a	l	i	a	\0									

Sono riempiti solo i primi 7 elementi del vettore (6 + NULL) , e gli altri non sono usati, al momento. Se il numero di caratteri supera la dimensione della variabile viene segnalato ERRORE.

Le stringhe possono contenere anche **spazi bianchi**:

char Nazione[16]="REPUBBLICA CECA";

R	E	P	U	B	B	L	I	C	A		C	E	C	A	\0

Operazioni di base sulle stringhe

- **Stampa di una stringa: cout**
  E' possibile stampare un elemento per volta, all'interno di un ciclo **for** proprio come si fa per un vettore di tipo numerico, ma **attenzione alla condizione di uscita dal ciclo** (non è detto che sia nota a priori la lunghezza di una stringa):

```
// stampa di una stringa
for  (int i=0; Nazione[i]!='\0' ; i++)
     cout << Nazione[i];
```

// stampa fino a quando non viene trovato il carattere NULL di fine stringa

Che è equivalente a scrivere:

```
// stampa di una stringa
for  (int i=0; Nazione[i] ; i++)
     cout << Nazione[i];
```

// stampa fino a quando non viene trovato il valore binario 0 che equivale al valore FALSO

Operazioni di base sulle stringhe

- **Stampa di una stringa: cout**

A differenza dei vettori numerici, la stampa di una stringa si ottiene anche semplicemente scrivendo il suo **nome nel cout senza l'uso dell'indice di scorrimento degli elementi:**

```
// stampa di una stringa
cout << Nazione;
```


Operazioni di base sulle stringhe

- **Lettura di una stringa da tastiera: cin**

A differenza dei vettori numerici, la lettura di una stringa si ottiene semplicemente scrivendo il suo **nome nel cin senza l'uso dell'indice di scorrimento degli elementi:**

```
char Nazione[16];

// lettura di una stringa
cout << "Inserisci il nome della nazione ";
cin >> Nazione;
```

Operazioni di base sulle stringhe

- **Lettura di una stringa da tastiera: cin**

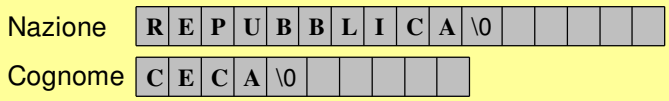
  Si deve tener presente che **cin** acquisisce caratteri da tastiera con la seguente **limitazione**:

  – si possono leggere **soltanto parole** e **non intere frasi** in quanto l'operatore di estrazione usa come delimitatore qualsiasi occorrenza di un carattere invisibile (**spazio**, **tabulazione**, **nuova linea**).

  Questo significa che se un nome contiene uno ***spazio bianco*** viene considerato non una stringa, ma due stringhe.

```
char Nazione[16];
char Cognome[10];
// lettura di una stringa
cout << "Inserisci la nazione ";
cin >> Nazione;
cout << "Inserisci il cognome ";
cin >> Cognome;
```

Se da tastiera viene digitato "REPUBBLICA CECA", il cin legge solo "REPUBBLICA" e la parola "CECA" viene letta con il cin successivo.



Operazioni di base sulle stringhe

- **Lettura di una stringa da tastiera: gets**

  Uno dei modi per superare il problema di poter leggere da tastiera stringhe contenenti il carattere *spazio* è utilizzare la funzione **gets** della **libreria stdio.h**:

  gets (*stringa*); // la stringa può contenere spazi bianchi

```
Esempio:
    #include <iostream>
    #include <stdio.h>
    using namespace std;

    int main( )
    { char Cognome[20];
      cout<<"Inserisci il cognome : ";
      gets (Cognome);
     ......
     ......
```

La funzione **gets** trasferisce l'intero buffer di input nella stringa Cognome e riconosce come unico terminatore il carattere **new line** (INVIO) che é sempre l'ultimo carattere del buffer e lo sostituisce con il carattere NULL.

Operazioni di base sulle stringhe

- **Lettura di una stringa da tastiera: cin.getline**

  Un altro modo per poter leggere da tastiera stringhe contenenti il carattere *spazio* è utilizzare la funzione **cin.getline** della **libreria stdio.h**:

cin.getline (*stringa, max*);

// vengono letti i caratteri inseriti qualsiasi essi siano fino a **new line** (INVIO) o per un massimo di **max–1** caratteri. La stringa viene completata con '\0'.

Esempio:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main( )
{ char Indirizzo[80];
  cout<<"Inserisci l'indirizzo: ";
  cin.getline (Indirizzo, 80);
 ......
 ......
```


Operazioni di base sulle stringhe

- **Lettura di una stringa da tastiera**

Riassumendo

char S[30];	
cin >> S;	// si legge fino al primo ***spazio*** (cin interrompe l'acquisizione di un avariabile quando viene incontrato uno *spazio bianco*)
gets(S);	// si legge fino al **new line** (INVIO).
cin.getline(S, max);	// si legge fino al **new line**, per un massimo di **max – 1** caratteri. La stringa viene completata con '\0'.

Attenzione: può accadere che nel buffer di input restino dei caratteri non letti, perché eccedenti la dimensione della stringa che viene acquisita da tastiera (es. la stringa è di 30 car. e ne digitiamo 35). Questi caratteri residui (nel nostro es. 5) andrebbero a finire nel successivo input e il programma non si fermerebbe in attesa di digitazione da tastiera trovando già i caratteri nel buffer.

Le funzioni di libreria per le stringhe

Una stringa è un vettore e per questo motivo non è possibile ad esempio copiarne direttamente il contenuto in un'altra stringa o confrontare direttamente due stringhe.

Esistono delle funzioni predefinite contenute nella **libreria string.h**.

Qui di seguito ve ne sono elencate alcune delle più usate:

strlen()	Calcola la lunghezza di una stringa
strcpy()	Copia una stringa in un'altra stringa
strncpy()	Copia i primi n caratteri di una stringa in un'altra stringa
strcat()	Concatena due stringhe
strncat()	Concatena i primi n caratteri di una stringa ad un'altra stringa
strcmp()	Confronta due stringhe
strncmp()	Confronta i primi n caratteri di due stringhe

Le funzioni di libreria per le stringhe

– **strlen(***Str1***)** restituisce il numero di caratteri che compongono una stringa (escluso il carattere NULL)

```
int n;
char Nazione[16]="Italia";
n= strlen (Nazione);              // n conterrà il valore 6
```

Le funzioni di libreria per le stringhe

– **strcpy(***Str1, Str2***)** copia la stringa *Str2* nella stringa *Str1*.

L'eventuale contenuto di *Str1* precedente all'assegnamento viene perso.

Attenzione: la dimensione di *Str1* deve essere sufficientemente ampia da accogliere tutti i caratteri di *Str2* compreso il terminatore '\0.
Se il numero di caratteri della variabile sorgente supera la dimensione della variabile destinazione **NON viene segnalato ERRORE** a run time.

```
char Studente[30];
char Cognome[15]="Rossi";
strcpy (Studente, Cognome);   // Studente conterrà la
                                 stringa "Rossi"
```


Le funzioni di libreria per le stringhe

– **strcat(***Str1, Str2***)** concatena la stringa *Str2* in fondo alla stringa *Str1* (appende in fondo) e valgono le stesse avvertenze di **strcpy.**

```
int n;
char Studente[30];
char Cognome[15]="Rossi", Nome[10]="Mario";
strcpy (Studente, Cognome);   // Studente conterrà la
                                 stringa "Rossi"
strcat (Studente, ' ');          // Concatena lo spazio
strcat (Studente, Nome);         // Concatena il nome
```

Studente

R	o	s	s	i		M	a	r	i	o	\0		.....	

Le funzioni di libreria per le stringhe

– **strcmp(***Str1, Str2***)** confronta la stringa *Str1* con la stringa *Str2.*

Restituisce:
- il valore 0 se le stringhe sono identiche
- un valore <0 (-1) se la stringa *Str1* è minore di *Str2*
- un valore >0 (1) se la stringa *Str1* è maggiore di *Str2*

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'.

Il confronto viene fatto partendo dal primo carattere di entrambe le stringhe e se sono uguali continua confrontando la successiva coppia di caratteri delle due stringhe fino a quando non viene trovata una coppia di caratteri diversi o non viene raggiunto il NULL di fine di una delle due stringhe.

Le funzioni di libreria per le stringhe

– **strcmp(***Str1, Str2***)** confronta la stringa *Str1* con la stringa *Str2.*

Esempio:

```
…..
char s1[10]="ieri";
char s2[10]="oggi";
int c;
c = strcmp(s1, s2);
if (c==0)
   cout<<"le due stringhe sono uguali";
else
    if (c<0)
        cout <<s1<< " è minore di "<<s2;
    else
        cout <<s1<< " è maggiore di "<<s2;
…..
```

Stringhe (la classe *string*)

Le librerie standard consentono di definire variabili di **tipo string**.

Con questo tipo di variabili il programmatore può disinteressarsi del problema della lunghezza delle stringhe.

Esempio

```
#include <string>
using namespace std;
int main()
{string s1 = “Hello”;
 string s2 = “world”;
 string s3 = s1 + “  ” + s2;
 cout << s3 << endl;
 }
```

Con le variabili di tipo *string* la concatenazione tra stringhe avviene con l’operatore **+** .

Nell’esempio sul monitor comparirà:

```
Hello world
```


Stringhe (la classe *string*)

Con le variabili di tipo *string* si possono utilizzare gli operatori di confronto come per le variabili numeriche (==, <, <=, >, >= e !=).

```
string a = “Paolo”;
string b = “Gianni”;
if (a = = b)
     cout <<a << “ = ”<<b;
else
     if (a < b)
          cout <<a << “ < ”<<b;
     else
          cout <<a << “ > ”<<b;
…..
```

sul monitor compare:

```
Paolo  >  Gianni
```

```
string a = “Paolo”;
string b = “Paolo”;
if (a = = b)
     cout <<a << “ = ”<<b;
else
     if (a < b)
          cout <<a << “ < ”<<b;
     else
          cout <<a << “ > ”<<b;
…..
```

sul monitor compare:

```
Paolo  =  Paolo
```

Stringhe (la classe *string*)

Dichiarazione di un vettore di stringhe

```
string  Nazioni[5];          // questo è un vettore costituito da 5 elementi,
                                ciascuno di tipo string
```

```
// lettura da tastiera degli elementi del vettore
   (ricordare che  cin non  consente  l'acquisizione di stringhe contenenti
   spazi bianchi)

for (int i =0; i < 5; i++)
   { cout <<"Inserisci la nazione di posizione "<<i<<": ";
     cin >> Nazioni [ i ];
   }
```

Vettore Nazioni

0	Italia
1	Francia
2	Spagna
3	Inghilterra
4	Germania

Stringhe (la classe *string*)

```
// Visualizza il contenuto degli elementi

for (int i =0; i < 5; i++)
   cout<< Nazioni [ i ]<<endl;
```

```
Italia
Francia
Spagna
Inghilterra
Germania
```