LE STRINGHE IN C++

Le stringhe

Si usano per rappresentare parole, frasi, codici alfanumerici e più in generale qualunque tipo di sequenza numerica non aritmetizzabile.

In C e C++ non esiste un vero e proprio tipo stringa. Le stringhe sono sequenze di dati di tipo char che terminano con \0.

\0 è un carattere di controllo che indica al compilatore la fine della stringa.

Le stringhe

Un vettore di caratteri non è necessariamente una stringa perché non sempre termina con \0.

Le stringhe in C++

Il C++ mette a disposizione la classe **string,** dotata di una serie di metodi che rendono le stringhe di facile gestione, contenuti nella seguente libreria:

#include <string>

Le stringhe in C++

Una stringa si può definire come:

string a; ← dichiara una stringa di dimensione qualunque

string a=“ciao”; ← dichiara una stringa e la inizializza a ciao. Lo spazio in memoria viene preparato opportunamente per accogliere la stringa.

Le stringhe in C++

Una stringa si può definire come:

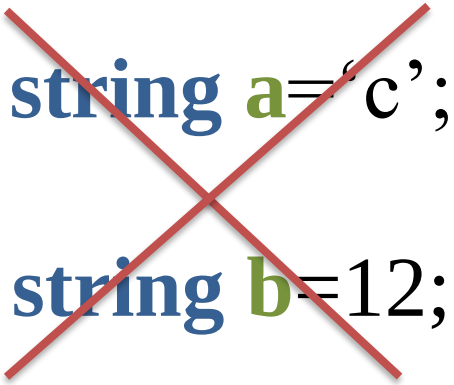
string a=“ciao”;

string b=a; ← dichiara una stringa e le assegna un contenuto pari a quello della stringa **a.**

L’operatore di assegnazione consente di effettuare la copia di stringhe.

Le stringhe in C++

Una stringa non si può inizializzare con caratteri o interi:



L'errore viene segnalato dal debugger del compilatore.

Le stringhe in C++

L’acquisizione e la visualizzazione di una stringa avvengono attraverso le funzioni standard di IO (cin e cout)

Gestione delle stringhe

Per la gestione delle stringhe il linguaggio C++ mette a disposizione una libreria apposita:

string

Alcune utili funzioni (metodi) della libreria sono:

substr()

empty()

length()

Lunghezza di una stringa

s.length()

Restituisce un intero, pari alla lunghezza della stringa alla quale si applica, compresi spazi bianchi.

```
string a="Leggo un libro giallo";
int x;
x=a.length();
cout << x;
```

21

Test di stringa vuota

s.empty()

Restituisce un booleano [true (1) o false (0)] che indica se la stringa è vuota o meno.

```
string a="Leggo un libro";
bool x;
x=a.empty();
cout << x;
```

0

```
string a="";
bool x;
x=a.empty();
cout << x;
```

1

Test di stringa vuota

Attenzione: Una stringa composta da spazi bianchi non è vuota.

```
string a="";
bool x;
x=a.empty();
cout << x;
```

```
1
```

```
string a=" ";
bool x;
int n;
x=a.empty();
n=a.lenght();
cout << x;
cout << n;
```

```
0
1
```

Concatenazione di stringhe

stringa1 + stringa2

L’operatore + agisce come somma quando riceve due dati numerici e agisce come concatenazione quando riceve due stringhe.

```
string a = "ciao", b="a tutti";
string saluto = a+b;
cout << saluto;
```

ciaoa tutti

Attenzione ad inserire gli spazi dove necessario

Confronto tra stringhe

Il confronto tra stringhe può essere effettuato tramite gli operatori relazionali

== != Restituiscono true (1) o false (0).

Due stringhe sono uguali se hanno la stessa lunghezza e sono composte dagli stessi caratteri (maiuscolo e minuscolo sono diversi) che si succedono nel medesimo ordine.

```
string a = "ciao", b="Ciao", c="ciao " ;
cout << (a==b) << endl;
cout << (a!=c);
```

```
0
1
```

Confronto tra stringhe

Il confronto tra stringhe può essere effettuato tramite gli operatori relazionali

< > >= <=

Restituiscono true (1) o false (0).

L'ordinamento è da intendersi in senso lessicografico.

Le lettere minuscole sono maggiori delle maiuscole (vedi codice ASCII)

Confronto tra stringhe

Il confronto tra stringhe può essere effettuato tramite gli operatori relazionali

< > >= <=

```
string a = "ciao", b="Ciao", c="ciao " ;
cout << (a > b) << endl;
cout << (c < b);
```

```
1
0
```

Elementi di una stringa

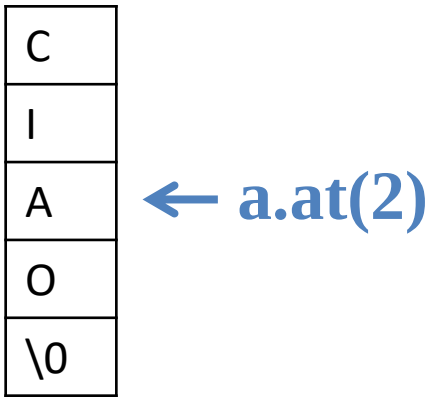
s.at(int n**)**

Restituisce un char: il carattere della stringa in posizione n.

Attenzione: In C++ il primo elemento di una stringa si trova in posizione 0.

```
string a="ciao";      c=a.at(n);
char c;               cout << c;
int n=2;
```

a



Estrazione di una sottostringa

s.substr(int start, int num**)**

Restituisce una stringa: la parte delle stringa iniziale che inizia dal carattere start e composta da num caratteri

```
string a="ciao a tutti";
string b;
b=a.substr(0,4);
cout << b;
```

ciao

Estrazione di una sottostringa

s.substr(int start, int num**)**

Restituisce **una stringa**: la parte delle stringa iniziale che inizia dal carattere start e composta da num caratteri

```
string a="ciao a tutti";
string b;
b=a.substr(0,1);
cout << b;
```

```
string a="ciao a tutti";
char b;
b=a.substr(0,1);
cout << b;
```

Estrazione di una sottostringa

s.substr(int start, int num**)**

Se start + num supera la dimensione della stringa o num è negativo, la funzione restituisce la stringa da start alla fine.

```
string a="ciao a tutti";
string b=a.substr(5,10);
cout << b << endl <<b.length();
```

a tutti
7

```
string a="ciao a tutti";
string b=a.substr(0,20);
cout << b << endl;
cout << b.length();
```

ciao a tutti
12

Estrazione di una sottostringa

s.substr(int start, int num**)**

Se num è negativo, la funzione restituisce l'intera stringa.

```
string a="ciao a tutti";
string b=a.substr(0,-3);
cout << b << endl;
cout << b.length();
```

ciao a tutti
12

Estrazione di una sottostringa

s.substr(int start, int num**)**

Se num vale 0 la funzione restituisce stringa vuota.

```
string a="ciao a tutti";
string b=a.substr(0,0);
cout << b;
cout << b.length();
cout << b.empty();
```

```
0
1
```

Sostituzione di una sottostringa

s.replace(int start, int num, string a**)**

Restituisce una stringa: la parte delle stringa iniziale che inizia dal carattere start e composta da num caratteri viene sostituita dalla stringa a

```
string a="ciao a tutti";
string b="mondo";
b=a.replace(5,7,b);
cout << b;
```

ciao mondo

Sostituzione di una sottostringa

s.replace(int start, int num, string a**)**

Restituisce una stringa: la parte delle stringa iniziale che inizia dal carattere start e composta da num caratteri viene sostituita dalla **stringa** a.

```
string a="ciao a tutti";
string b="C";
b=a.replace(0,1,b);
cout << b;
```

```
string a="ciao a tutti";
char b='C';
b=a.replace(0,1,b);
cout << b;
```

Sostituzione di una sottostringa

s.replace(int start, int num, string a**)**

La stringa a può essere di dimensione diversa rispetto alla sottostringa che deve sostituire

```
string a="ciao a tutti";
string b="";
b=a.replace(4,8,b);
cout << b;
```

ciao

Sostituzione di una sottostringa

s.replace(int start, int num, string a**)**

E’ una funzione molto dinamica che ricalcola la posizione dei caratteri che compongono la stringa e la sua lunghezza.

```
string a="ciao a tutti";
string b="Buongiorno";
b=a.replace(0,5,b);
cout << b << endl << b.length();
```

Buongiorno a tutti
18

Sostituzione di una sottostringa

s.replace(int start, int num, string a**)**

Se num=0 si comporta come un inserimento.

```
string a="ciao tutti";
b="a ";
a=a.replace(4,0,b);
cout << a;
```

ciao a tutti

Eliminazione di una sottostringa

s.erase(int start, int num**)**

Restituisce una stringa: la parte delle stringa iniziale che inizia dal carattere start e composta da num caratteri viene eliminata

```
string a="ciao a tutti";
int n = a.lenght()
b=a.erase(4,n);
cout << b;
```

ciao