# LE FUNZIONI

Una stringa in informatica è una sequenza di caratteri con un ordine prestabilito.

I segmenti di codice che realizzano operazioni semplici o complesse, ma che già di per sé consentono di risolvere un problema piccolo o grande, possono essere visti come programmi “autonomi” (ad esempio il calcolo della media o di una somma), da utilizzare all’interno di altri programmi.

In generale un programma C++ è formato da una o più funzioni di cui la principale assume il nome **main**: si tratta della funzione che viene eseguita per prima all’avvio del programma. La prima riga della funzione ne definisce la firma (signature o header): essa comprende il nome, i parametri (con i relativi tipi) e il tipo del valore restituito(ad esempio: tipo\_restituio nome\_funzione(<lista\_parametri\_formali**)**. Le altre righe rappresentano il corpo della funzione che, nel linguaggio di programmazione C++, costituiscono un blocco delimitato dai simboli «{» e «}».

Va inoltre sottolineato che le funzioni lavorano in modalità parametrica, in quanto devono soddisfare situazioni sempre differenti.

* **VANTAGGIO NELL’UTULIZZO DELLE FUNZIONI**
* Riusabilità: è possibile riutilizzare lo stesso codice (come “mattone”) per la soluzione di problemi diversi .
* Astrazione: esprimere operazioni complesse in modo sintetico.
* Risparmio: scrivere una sola volta un codice utilizzato più volte.
* OSSERVAZIONE:

L’istruzione return conclude l’esecuzione del codice del corpo di una funzione e costituisce il meccanismo per la restituzione del risultato della computazione eseguita dalla funzione stessa: il valore della variabile, o dell’espressione, che segue la parola chiave return è il valore della valutazione della funzione al momento dell’invocazione. L’invocazione di una funzione precedentemente definita avviene specificandone il nome e i parametri su cui deve essere eseguito il calcolo che essa implementa.

**Nell’uso delle funzioni si distinguono i parametri formali dai parametri**

**attuali**: i primi sono quelli definiti nell’intestazione della funzione,

mentre i secondi sono quelli specificati al momento della chiamata della

funzione e sono anche chiamati argomenti della funzione.

Al momento dell’invocazione della funzione il valore dei parametri attuali

viene «copiato» nelle variabili che costituiscono i parametri formali

della funzione stessa ed è in base a questo valore che viene eseguita la

computazione. Nel caso in cui una funzione preveda più di un parametro

formale, nella sua invocazione devono essere specificati altrettanti

parametri attuali di tipo conforme a quelli formali. L’associazione tra

parametri formali e attuali avviene per posizione: il primo con il primo,

il secondo con il secondo e così via.

* OSSERVAZIONE:

Una funzione può non avere parametri: in questo caso

la sua firma conterrà al posto dell’elenco dei parametri la parola chiave

void e la sua invocazione avverrà specificandone il nome della funzione

seguito dai simboli “(“ e ”)”.

La restituzione del risultato da parte di una funzione mediante l’istruzione

return deve soddisfare le seguenti regole:

* Una funzione restituisce al più un unico valore in accordo con il tipo

di risultato dichiarato nella sua firma.

* Una funzione può non restituire valori; in questo caso il tipo di risultato

da dichiarare nella definizione della funzione è il tipo void.

* OSSERVAZIONE:

Una funzione di tipo void, non restituendo alcun risultato,

viene invocata direttamente, senza dover dipendere da istruzioni

di assegnamento o di altro tipo.

* **L’INVOCAZIONE DI UNA FUNZIONE PREVEDE I SEGUENTI PASSI:**
* Invocazione della funzione e memorizzazione del punto corrente di esecuzione

nella funzione invocante;

* trasferimento dell’esecuzione alla prima istruzione della funzione invocata

con relativa «copia» del valore dei parametri attuali nelle variabili

che costituiscono i parametri formali;

* esecuzione del codice della funzione invocata;
* ripresa dell’esecuzione dal punto di sospensione nel codice della funzione;
* invocante memorizzato al momento dell’invocazione con eventuale;
* assegnazione del valore calcolato restituito dalla funzione invocata.
* OSSERVAZIONE:

L’ambiente interno di una funzione è locale alla funzione stessa: le variabili

definite nel corpo della funzione – così come le variabili che

costituiscono i parametri formali – hanno una visibilità e un tempo di

vita limitato al tempo di esecuzione della funzione stessa.

Solo le variabili definite a livello globale (esternamente al corpo di una

qualsiasi funzione, compresa la funzione main) sono visibili da parte

di tutte in tutte le funzioni che compongono un programma. Valgono

in proposito le stesse regole che si applicano a un blocco: se in una

funzione viene definita una variabile che ha lo stesso nome di una variabile

definita a livello globale, la variabile locale maschererà quella

globale nel corso dell’esecuzione della funzione.

* **Passaggio dei parametri per valore e per riferimento**

Nel linguaggio di programmazione C++ sono previste due distinte modalità

per il passaggio dei parametri alle funzioni.

**Passaggio dei parametri per valore**: al momento dell’invocazione

della funzione viene effettuata una copia del valore dei parametri attuali

nelle variabili dei corrispondenti parametri formali.

* OSSERVAZIONE:

Nel passaggio dei parametri per valore qualsiasi modifica

del valore dei parametri effettuata nel codice della funzione invocata

non si rifletterà in alcun modo nel valore degli argomenti della funzione

invocante.

**Passaggio dei parametri per riferimento**: al momento dell’invocazione

della funzione vengono associate le variabili che costituiscono i

parametri attuali alle variabili che costituiscono i parametri formali,

ovvero i parametri formali fanno riferimento alle stesse variabili che

costituiscono i parametri attuali. Il passaggio di parametri per riferimento

si effettua anteponendo nell’intestazione della funzione al

nome dei parametri formali il simbolo «&».

* OSSERVAZIONE:

In pratica, nel passaggio dei parametri per riferimento,

la funzione invocata e la funzione invocante utilizzano le stesse variabili e

le eventuali modifiche apportate al valore dei parametri formali nel codice

della funzione invocata si riflettono sul valore delle variabili corrispondenti

passate come parametri attuali nel codice della funzione invocante.

Il prototipo di una funzione è costituito dalla sola firma della funzione

(tipicamente un’unica riga di codice composta dal tipo e dal nome

della funzione, seguiti dalla lista dei tipi dei singoli parametri formali)

seguita dal simbolo «;».

* OSSERVAZIONE:

Nella lista dei parametri del prototipo di una funzione

è possibile – anche se non obbligatorio – inserire il nome dei parametri;

in questo caso il nome può differire da quello dei parametri formali

effettivi presenti nell’intestazione della definizione della funzione e utilizzati

nel codice.

La definizione preventiva dei prototipi delle varie funzioni utilizzate in un

programma permette al compilatore di conoscere le caratteristiche formali

delle singole funzioni prima della loro invocazione, liberando il programmatore

dalla necessità di seguire un ordine gerarchico nella loro definizione.

* **OVERLOADING DEI NOMI DELLE FUNZIONI:**

Nell’ambito dello stesso programma il linguaggio C++ permette di fornire

più definizioni di funzioni aventi lo stesso nome, se queste prevedono

parametri formali e/o tipi restituiti diversi.

Il compilatore distingue tra loro tali funzioni in base alla loro firma. Questa

tecnica di definizione multipla di una funzione prende il nome di overloading

(letteralmente «sovra caricamento») ed è una forma di polimorfismo.

# LE STRINGHE

Ogni *array* di caratteri che contiene una stringa termina con un valore 0 (cioè la sequenza di *escape* «\0»): un vettore di *N* caratteri può dunque ospitare stringhe lunghe al più *N* – 1 caratteri, perché una cella è riservata al carattere terminatore «\0».

### Per dichiarare una stringa lunga al massimo N (dove N è un qualsiasi numero) è possibile utilizzare una dichiarazione del tipo:

Char Stringa[N+1];

* OSSERVAZIONE:

Nel caso in cui i caratteri da immettere nel vettori siano minori di N, il carattere di terminazione verrà messo immediatamente dopo l’ultimo carattere, lasciando le celle restanti vuote. Si ponga attenzione al fatto che in fase di input la stringa inserita non contenga caratteri quali spazio bianco, tabulazione o ritorno a capo, perché questi vengono interpretati come fine stringa, causando la perdita degli eventuali caratteri che seguono.

## FUNZIONE STRCMP

La funzione strcmp consente di confrontare due stringhe passate come parametro,

ovvero:

Strcmp (stringa1, stringa2)

Date le due stringhe, stringa1 e stringa2

* Se stringa 1=stringa 2, il risultato è 0
* Se stringa1<stringa2, il risultato è <0
* Se stringa1>stringa2, il risultato è >0

## FUNZIONE STRCPY

Vi sono due modi per copiare l’elemento di un vettore in un altro vettore, essi sono:

* Copiare elemento per elemento dalla stringa sorgente alla stringa di destinazione fino a quando si incontra il carattere di fine stringa
* Utilizzare la funzione strcpy(destinazione, sorgente)

Il formato per utilizzare quest’ultima è il seguente:

Dove ‘destinazione’ è la stringa in cui verrà copiata la stringa ‘sorgente’

* OSSERVAZIONE:

La stringa ‘destinazione’ deve essere grande tale da ospitare tutti gli elementi della stringa ‘sorgente’.

## FUNZIONE STRLEN

La funzione strlen(stringa) restituisce la lunghezza della stringa passata come parametro, escluso il carattere di terminazione.

## FUNZIONE STRCAT

La funzione strcat(stringa1, stringa2) consente di concatenare due stringhe passate come parametro. Il risultato della concatenazione è memorizzato nel primo parametro (stringa1 conterrà il parametro finale).

## VETTORE DI STRINGHE

La sintassi per la dichiarazione di una di un vettore di N elementi di tipo stringa di D è:

Char NomeVettore [N] [D];

BIBLIOGRAFIA

* Documenti di supporto fornitici dal professore, fra i quali possiamo citare in particolare:
  + Prof. G. Ascia - Gestione delle stringhe in C (file pdf.);
  + D. Calvanese - Fondamentali di informatica - Corso di Laurea in Ingegneria Elettronica - A.A 2001/2002, (Sezione su strutture, file pdf.);
  + Cosimo Laneve - Dichiarazioni, funzioni e passaggio dei parametri – (file pdf.);
  + Prof. C. Maccheroni 2008/2009 - C/ Dev-C++ procedurale –(file pdf.)
  + F. Formichi, G. Meini - Corso di Informatica Algoritmi e linguaggio C++ (file pdf.);
* P. Camagni, R. Nikolassy - Corso di informatica linguaggio C e C++ Nuova edizione OPENSCHOOL (libro in adozione scolastico).
* [www.wikipedia.org](http://www.wikipedia.org);
* www.wikipedia.it;
* www.cplusplus.com;
* [www.html.it](http://www.html.it);
* [www.skuola.net](http://www.skuola.net).
* https://it.answers.yahoo.com/