Introduzione

Che cos'è il linguaggio C?

Il linguaggio C è nato per rispondere alle esigenze dei primi programmatori che erano costretti ad utilizzare i numeri binari per programmare, così vennero sviluppati i linguaggi assembly che permettevano di assegnare nomi simboli a specifiche istruzioni, uno speciale programma detto assembler si occuperà poi a tradurlo in linguaggio macchina. I linguaggi assembly sono comunque considerati linguaggi di basso livello in quanto esiste una corrispondenza uno a uno tra ciascuna istruzione del linguaggio sempre e una specifica istruzione macchina. Il programma non risulterà portabile, in quanto ogni processore avendo diversi set di istruzioni dovrà essere riscritto per ogni differente processore.

Nacquero successivamente i linguaggi di alto livello (tra cui il primo FORTRAN). I programmatori con i linguaggi di alto livello non dovevano più preoccuparsi dell'architettura di un particolare computer in quanto le istruzioni del programma venivano tradotte nelle varie istruzioni macchina.

Per funzionare un linguaggio di alto livello necessita di uno speciale programma che traduca le istruzioni del programma a istruzioni macchina ovvero il compilatore.

Tutti i processi di editing, la compilazione, esecuzione e debugging si può svolgere in un unico ambiente integrato detto Integrated Development Environment (IDE).

Esiste un altro modo per eseguire i programmi sviluppati in alto livello: possono essere interpretati. Un interprete analizza ed esegue contemporaneamente le istruzioni del programma (questo per agevolare la correzione dei programmi); i linguaggi interpretati però sono tipicamente più lenti dei linguaggi compilati (BASIC e Java sono due linguaggi che vengono interpretati).

Compilazione di un programma su Linux

Una volta scelto l'editor di testo (vi, emacs, visual studio code, Atom ecc…) Bisognerà salvare il nostro programma con l'estensione **'.c'**e si potrà procedere a compilarlo; se utilizzate il compilatore C GNU potrete utilizzare il comando gcc (esempio **$gcc programma.c**); una volta compilato potremmo procedere all'esecuzione grazie al comando **a.out.**

Il primo programma in C

Nella libreria del linguaggio C vi sono delle funzioni che sono già definite (come printf(), scanf(), ecc..).

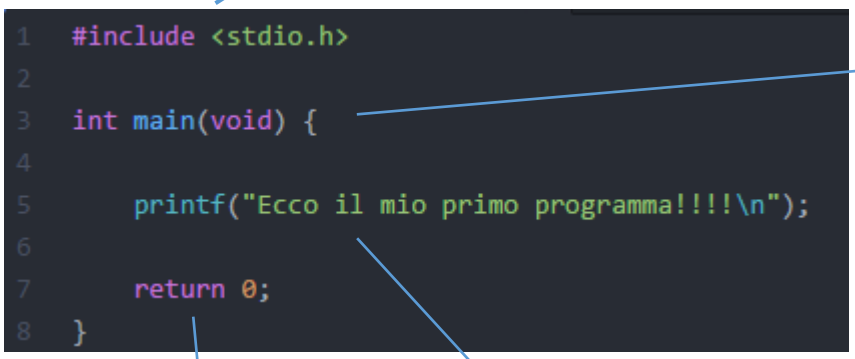
Come saprai già **scanf** è una funzione di **input**

Mentre **printf** è una funzione di **output**

#include ci permette di processare tutte le funzioni prima di compilare il programma

<stdio.h> sta per standard **input** and **output** e permette al programma l'input e l'output, per questo va in testa ad ogni programma

Con **#include** possiamo includere anche altre librerie come **#math** se vogliamo la radice quadrata



Definisce l'inizio della parte che verrà compilata.

Int sta a significare che verrà restituito un valore intero (0 nel caso il programma venga chiuso con **return 0;**

Main è invece il nome principale delle funzione

(void) significa che la funzione non necessita di alcun parametro in ingresso, perciò non è detto che debba essere sempre usato

Return 0; conclude l'esecuzione di main producendo il valore 0, ovvero senza errori.

È possibile utilizzare altri numeri per indicare degli stati di errore

printf è una funzione della libreria C che permette di visualizzare su schermo il suo argomento.

\n è riconosciuto come un singolo carattere che permette di andare a capo

Informazioni importanti

Nel linguaggio C esiste la distinzione fra caratteri maiuscoli e caratteri minuscoli, inoltre non ha importanza il punto da cui si inizia a digitare un'istruzione ciò può essere sfruttato per rendere il programma più leggibile usando ad esempio spazi e tabulazione.

La funzione printf e le variabili

Grazie alla routine printf è possibile visualizzare i valori delle variabili e i risultati dei calcoli, nell'esempio

```c
#include <stdio.h>

int main(void){

    int var1=50;
    int var2=25;
    int sum;

    sum = var1 + var2;

    printf("La somma di 50 + 25 e': %i\n",sum);

    return 0;
}
```

La prima istruzione ***int var1=50;*** dichiara che il valore dell'intero var1 è pari a 50 mentre la seconda istruzione dichiara che il valore di var2 è pari a 25; nell'istruzione int sum viene solo dichiarato che si tratta di una variabile intera, nella quarta istruzione prenderà il valore della somma di var1 e var2.

Per concludere printf mostrerà a schermo la somma delle prime due istruzioni contenute nella variabile sum.

```c
    printf("La somma di 50 + 25 e': %i\n",sum);
```

%i dichiara come verrà stampata (il tipo di dato) in ordine di posizione la variabile dopo la virgola, in questo caso verrà stampata la variabile sum di tipo intero (%i)

il carattere % è un carattere speciale riconosciuto dalla funzione printf che permette di specificare il tipo di valore da visualizzare.