2 - Identificatori, Sequenza e Selezione

Un identificatore, dal punto di vista sintattico è una sequenza di caratteri alfabetici e/o numerici.
In C un identificatore deve iniziare con una lettera o con il simbolo underscore (`_`)
Ci sono due tipi di identificatori:

- **Standard:** nome che ha già un significato preciso (come printf o scanf)
- **Definito dall'utente**: una variabile o un operazione definita dall'utente (blocco di istruzioni)

Nel caso dei dati definiti dall'utente non è possibile usare le parole riservate (printf), gli identificatori sono case-sensitive (es. Media, media e MEDIA son 3 variabili diverse) e non possono contenere spazi (in generale)

Le istruzioni di assegnazione

Le istruzioni di assegnazione permettono di memorizzare un valore o un risultato di una computazione di una variabile

```
Variabile = Espressione

Esempio

a=6 // Valore costante
b=secondo valore // variabile
Media=(a+b)/2 //risultato di una espressione aritmetica
```

Le istruzioni di ingresso

Le istruzioni di ingresso consentono di acquisire informazioni dall'esterno, con il C si usa l'operatore `scanf` definita nella libreria `stdio.h`

Placeholder

I placeholder permettono alle operazioni di ingresso (e di visualizzazione) di indicare con quale tipo di dato si sta lavorando

In C i placeholders possono rappresentare anche la precisione di variabile

Casting

L'istruzione di casting consente di convertire un tipo di dato in un altro tipo di dato
Un float viene automaticamente convertito in un intero o viceversa

Commenti

Sono parti del programma che sono ignorate dal compilatore e non tradotte in linguaggio macchina, ma utili per:

- Rendere più facile la comprensione di un programma
- Descrivere lo scopo del programma
- Descrivere il significato degli identificatori e/o lo scopo di ciascun passo del programma

Costanti

In alcuni programmi si usano anche dati che non variano mai, tali vengono chiamate costanti, definite nella parte del programma che contiene le dichiarative

```
#define  NOME_COSTANTE valore
#define KMS_PER_MIGLIA 1.609
```

Strutture di controllo

La logica di un programma si basa sulle **strutture di controllo**, che regolano il flusso di esecuzione:

- **Sequenza**: È la più semplice, in cui le istruzioni vengono eseguite una dopo l'altra, nell'ordine in cui sono scritte.
- **Selezione**: Permette di scegliere tra due o più percorsi in base a una condizione. Ad esempio, un programma può stampare "Minorenne" se l'età dell'utente è inferiore a 18 anni. Questo concetto include il classico blocco `if-else`.

- **Iterazione**: Consente di ripetere un blocco di codice, magari per calcolare una somma o analizzare una lista di numeri, finché una condizione è vera.

Operatori logici

Gli operatori relazionali e logici permettono di costruire condizioni e verifiche:

- **Relazionali**: Paragonano due valori (`<`, `>`, `==`, etc.) e restituiscono un valore booleano (vero o falso). Ad esempio, `x > 0` verifica se `x` è maggiore di zero.
- **Logici**: Combinano più condizioni con operatori come `&&` (AND), `||` (OR) e `!` (NOT). Ad esempio, `voto >= 18 && voto <= 30` verifica che il voto sia compreso tra 18 e 30. Come nelle espressioni matematiche, esiste un ordine gerarchico tra gli operatori. Le parentesi possono essere usate per forzare un certo ordine di valutazione e rendere il codice più leggibile.

Selezioni innestate

Le selezioni innestate (o `if-else` multipli) permettono di gestire situazioni con più alternative, ma quando il numero di alternative cresce è importante prestare attenzione all'indentazione del codice per evitare errori e mantenere il programma leggibile.