L’AMBIENTE LOCALE ...

In C, ogni funzione ha il suo *ambiente locale* che comprende i parametri e le variabili definite localmente alla funzione.

Esempio:

```
float fahrToCelsius(float f){
  float fattoreDiScala = 5.0 / 9;
  return fattoreDiScala * (f - 32);
}
```

… E L'AMBIENTE GLOBALE

Esiste però anche un *ambiente globale:* quello dove tutte le funzioni sono definite.

Nell'ambiente globale si possono anche definire variabili, dette *variabili globali*

La denominazione *"globale"* deriva dal fatto che l’environment di definizione di queste variabili *non coincide con quello di nessuna funzione* (neppure con quello del main).

REGOLE DI VISIBILITÀ

- **Variabili locali e parametri formali sono *visibili* ***SOLO all'interno della funzione*** di cui fanno parte.**
- **Viceversa, le variabili globali sono ***visibili ANCHE dentro le funzioni*** scritte fisicamente di seguito alla definizione della variabile globale stessa
- ...a meno che non siano mascherate da variabili locali omonime.

VARIABILI GLOBALI

- **Una *variabile globale* è dunque definita *fuori da qualunque funzione* ("a livello esterno")**
- **tempo di vita = *intero programma***
- **scope = *il file in cui è dichiarata* dal punto in cui è scritta in avanti**

ESEMPIO

Definizione (e inizializzazione) di una *variabile globale*

```
int trentadue = 32;

float  fahrToCelsius( float F ){
float temp = 5.0 / 9;
    return temp * ( F - trentadue );
}
```

ESEMPIO

Scope di visibilità: l'intero file in cui è scritta, da lì in avanti

```
int trentadue = 32;

float  fahrToCelsius( float F ){
float temp = 5.0 / 9;
    return temp * ( F - trentadue );
}
```

VARIABILI GLOBALI: DICHIARAZIONI E DEFINIZIONI

Anche per le variabile globali si distingue fra *dichiarazione* e *definizione*

- **al solito, la dichiarazione esprime proprietà associate al simbolo, *ma non genera un solo byte di codice***
- **la *definizione* invece implica *anche allocazione di memoria,* e <u>funge anche da dichiarazione</u>.**

ESEMPIO

File `prova3.c`

```
int trentadue = 32;
float fahrToCelsius(fl

main() {
  float c = fahrToCelsius(86);
}

float fahrToCelsius(float f) {
  return 5.0/9 * (f-trentadue);
}
```

<u>Definizione</u> (e inizializzazione) **della variabile globale**

***<u>Uso</u>* della variabile globale**

VARIABILI GLOBALI: DICHIARAZIONI E DEFINIZIONI

Problema:

Come distinguere la dichiarazione di una variabile globale dalla sua definizione?

- nelle funzioni è facile perché la dichiarazione ha un "`;`" al posto del corpo `{….}`
- ma qui non c'è l'analogo ….. e allora??

Risposta:

si usa l'apposita parola chiave `extern`

VARIABILI GLOBALI: DICHIARAZIONI E DEFINIZIONI

Quindi:

- `int trentadue = 10;`

  è una definizione (con inizializzazione)

- `extern int trentadue;`

  è una dichiarazione

ESEMPIO

File `prova4.c`

```
extern int trentadue;

float fahrToCelsius(float f) {
  return 5.0/9 * (f-trentadue);
}

main() {
  float c = fahrToCelsius(86);
}
int trentadue = 32;
```

Dichiarazione **variabile globale**

Uso della var.glob.

Definizione **della variabile globale**

VARIABILI GLOBALI: DICHIARAZIONI E DEFINIZIONI

- **Come sempre, una applicazione può contenere**
  - **più dichiarazioni per lo stesso simbolo**
  - **ma una e una sola *definizione* di tale simbolo.**
- **NB: la dichiarazione *non* può contenere *anche una inizializzazione***
  - **è solo un avviso, *la variabile non è lì!***

```
extern float pigreco = 3.1415;
```

NO!

VARIABILI GLOBALI come COMPONENTI SW

- **Anche una variabile globale è un *componente software***
  - **in particolare, un componente che fornisce *dati* (non comportamenti)**
- **Come tale, costituisce una *unità di traduzione:***
  - **può essere definita in un file a sé stante**
  - **e compilata per proprio conto**
  - **pronta per essere usata da chiunque**

VARIABILI GLOBALI : USO

- **Il cliente deve *incorporare la dichiarazione* della variabile globale che intende usare:**

```
extern int trentadue;
```

- ***Uno dei file sorgente nel progetto dovrà poi contenere la definizione* (ed eventualmente l'inizializzazione) della variabile globale**

```
int trentadue = 10;
```

DALL'ESEMPIO SU UN SOLO FILE...

File `prova4.c`

```
float fahrToCelsius(float f);
main() {
  float c = fahrToCelsius(86);
}
extern int trentadue;
float fahrToCelsius(float f) {
  return 5.0/9 * (f-trentadue);
}
int trentadue = 32;
```

... ALL'ESEMPIO SU TRE FILE

File `main.c`

```
float fahrToCelsius(float f);
main() { float c = fahrToCelsius(86); }
```

File `f2c.c`

```
extern int trentadue;
float fahrToCelsius(float f) {
  return 5.0/9 * (f-trentadue);
}
```

File `32.c`

```
int trentadue = 32;
```

ARCHITETTURA DELL'APPLICAZIONE

Strutturando l'esempio su più file, l'importante è capire *chi usa cosa*:

- **Il `main` usa la funzione `fahrToCelsius`**
- **La funzione `fahrToCelsius` usa la variabile globale `trentadue`**