VARIABILI - SCOPE STATIC

venerdì 2 dicembre 2022 22:25

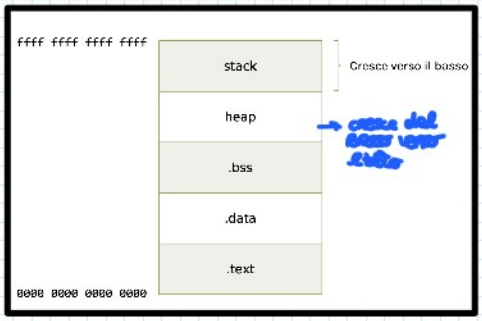
LE VARIABILI DEVONO AVERE UN TIPO, che poi corrisponde al suffisso IN ASSEMBLY, IN modo tale che il processore possa generare il corrispondente codice ASSEMBLY.

SCOPE - AMBITO

- Ogni variabile dichiarata nel programma ha un certo *ambito* (scope), che determina la *visibilità* della variabile a determinate porzioni del programma:

- variabili **globali**: occupano memoria all'interno delle sezioni `.data` e `.bss` (se inizializzate a zero)
- variabili **locali** (o *automatiche*): occupano memoria all'interno dello stack
- varibili **statiche**: come le variabili globali, ma possono essere accedute solo all'interno della funzione/modulo C

dichiarate localmente all'interno della funzione



Nel linguaggio C l'istruzione **static** mi permette di creare variabili persistenti in una funzione.

static tipo nome

Dove tipo e nome sono rispettivamente il formato e il nome della variabile.

A cosa serve?

Le variabili interne delle funzioni sono variabili locali. Una volta terminata l'esecuzione della funzione, il loro valore si perde.

L'istruzione static mantiene in memoria il valore di una variabile interna della funzione dopo ogni chiamata.

QUANDO LA FUNZIONE TERMINA, LA VARIABILE STATICA SOPRAVVIVE, PERCHÈ È STATA ELEVATA A GLOBALE, MA PUÒ ESSERE ACCEDUTA SOLO DENTRO LA FUNZIONE:

```c
#include <stdio.h>

int fun() {
    static int n=0;
    printf("n = %d \n",n);
    n++;
    return n;
}
int main() {
    fun();
    fun();
    fun();
    return 0;
}
```

OUTPUT:

```
n = 0
n = 1
n = 2
```