COMANDI:

SINTASSI:

C : Skip $\mid x := e \mid$ WHILE B DO $C \mid$ IF B DO C_1 ELSE C_2

$d; C$

Blocco (deve Essere delimitato) ⇒ procedure Esempio

BLOCCHI IN-LINE ⇒ Senza Nome (permette di Creare un **Ambiente locale**)

(σ)

MEMORIE ⇒ Associazioni Tra **Locazioni e Valori Memorizzab.** e Sono **Legami dinamici** (CAMBIA VALORE NELLA LOCAZIONE DURANTE ESECUZIONE)

Sono **MODIFICHE IRREVERSIBILI** (una Direzione) Quelle che Subisce la Memoria

(ρ)

AMBIENTE ⇒ Insieme di Associazioni Tra **Identificatori** ed **Oggetti Denotabili** (COSTANTI e LOCAZIONI)

Vengono Associati/**BINDING STATICI**, che NON Cambiano durante l'ESECUZIONE dei Comandi

Subiscono delle **Trasformazioni REVERSIBILI** (esistono dei Meccanismi Automatici per la Manipolazione dell'Ambiente)

BLOCCO:

SCOPO ⇒ Per CREARE Ambienti Locali

Esistono METODI di Allocazioni di BLOCCHI diversa, che ad esempio possono Come NO permettere la RICORSIONE

BLOCCHI SENZA-NOME/INLINE:

PROCEDURE: Blocchi con NOME che possiamo Richiamare

VISIBILITÀ:

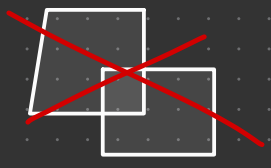
Visibilità di una dichiarazione va di Pari Passo con la dichiarazione di un Riferimento per un Identificatore

Occorrenze libere di un Identificatore Sono definiti dalla dichiarazione

Qual'è la dichiarazione che NE definisce il SIGNIFICATO della DICHIARAZIONE

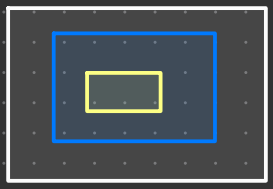
CONCETTO DI BLOCCO:

Devono Essere ANNIDATI INTERAMENTE



Non Può Essere devo Chiudere Prima Quello Precedente Per Aprirne uno Nuovo

Possono Essere Solamente ANNIDATI Tra di loro



Regola STANDARD di **Visibilità di una DICHIARAZIONE**

Locale è visibile (SCOPE) Nel BLOCCO in cui è INSERITA ed in TUTTI i Blocchi Annidati

Ma Nel Caso in cui Andassimo a dichiarare un NUOVO **identificatore con Stesso Nome** allora Verrebbe **SOVRASCRITTO TEMPORANEAMENTE** (ed una Volta chiuso il BLOCCO Tornerà alla dichiarazione precedente)

ESEMPIO:

```
(1)
{const  i: int = 3;
 :
 {var   i: real = 1.2;
  :
  i
 }
 i....
}
(2)
```

ha SOVRASCRITTO LA PRECEDENTE DICHIARAZIONE

Scope della Variabile i REALE (validità della i REALE)

Scope della i CONST ha un BUCO, (1) e (2) visto che è stata SOVRASCRITTA

Si vede Sempre e Solo Verso l'INTERNO MAI Verso l'ESTERNO

SCOPE:

Area di Testo del CODICE Nel Quale tutte le OCCORRENZE Libere di un Identif. Sono LEGATE alla Stessa dichiarazione

Concetto di SPAZIO e NON di TEMPO

SCOPING STATICO ⇒ Ambiente di riferimento degli identificatori è STABILITO a Tempo di Compilazione

BLOCCO che lo CONTIENE SINTATICAMENTE

NON tutti i linguaggi lo CONCEDONO

Classificazione degli ID:

- LOCALE ⇒ (Al Blocco) Quando è SCRITTA Nel BLOCCO
- NON LOCALE ⇒ Quando è VISIBILE al Blocco ma Scritta in un BLOCCO DIFFERENTE (a lvl più ESTERNO)

NON ACCORGERSI DELLA RIDEFINIZIONE È UN ERRORE Comune

- GLOBALI ⇒ Quando è VISIBILE a TUTTI I BLOCCHI

AMBIENTE LOCALE:

Insieme delle Associazioni locali del BLOCCO:

- dichiarazioni locali
- Parametri formali (PROCEDURE)

add(m,n) ⇒ m,n fanno Parte dell' AMBIENTE

AMBIENTE NON LOCALE:

Visibile ma NON definito localmente

AMBIENTE GLOBALE:

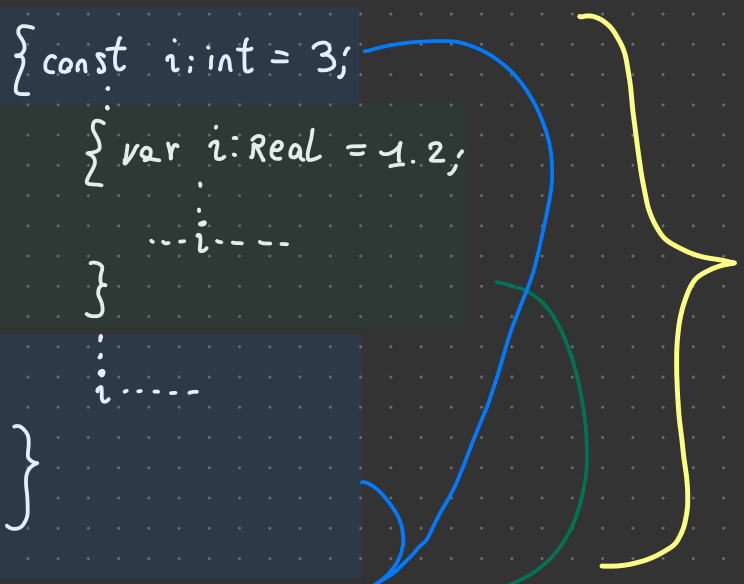
Ambiente Visibile all' Intero Programma

TEMPO DI VITA:

Corrisponde al Tempo in cui la locazione è ASSOCIATA all'ID.

Consiste nel TEMPO di Esecuzione Nel Quale tutte le occor. libere di un ID si Riferiscono alla Stessa Locazione

SCOPE:	TEMPO DI VITA:
Area di Codice	Tempo di Esecuzione
Dichiarazione	Locazione



Tempo di Vita di CONST di NON ha Interruzioni altrimenti NON sarei in Grado di TORNARE indietro e Viene Salvato da un ALTRA PARTE

→ Tempo di Vita = Scope in Questo Caso

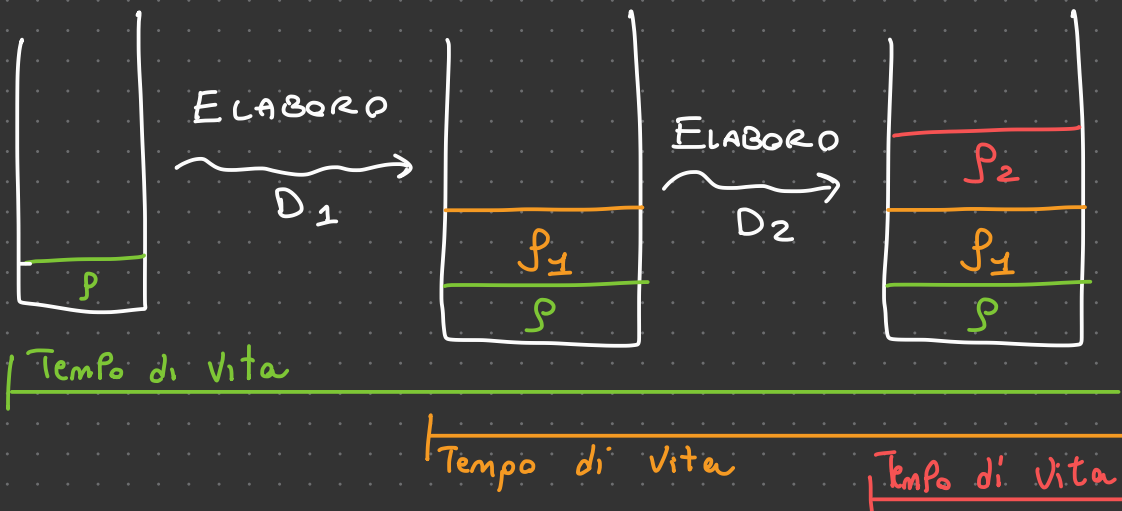
→ SCOPE di CONST i

SCOPE dipende Solo dalle REGOLE Mentre TEMPO di Vita dipende da Scelte Implementative RIGUARDO alla MEMORIA

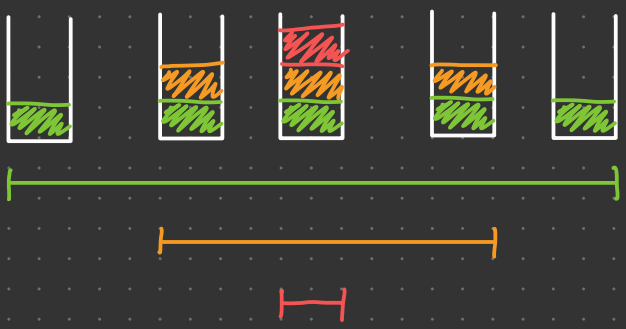
ESEMPIO:

```
{WHILE
  :
  { D1; //dichiarazioni
    C1; //Codice
    :
    { D2; //dichiarazioni
      C2; //Codice
    }
    C13;
  }
}
```

ρ_2 ρ_1 ρ



(*) Che poi Viene SUBITO deallocata e Quindi Si CONCLUDE Subito



Quando Sono USCITO dal WHILE

SEMANTICA D; C :

$$FI(d; c) = FI(d) \cup (FI(c) \setminus DI(d))$$

```
Var x:int = 3*y;   D
x := z + x;        C
```

} y è LIBERA ⇒ FI ma x è DEFINITA

$FI(c) = \{z, x\}$ ma Nel Blocco $FI(d; c) = \{z\}$

$$DI(d; c) = DI(d) \cup DI(c)$$

SEMANTICA STATICA dice Quando un Blocco è BEN formato ma NON ho Vincoli vista che devo Verificare che sia i COMANDI e dichiarazioni Siano BEN FORMATI

Ambiente Costruito da d

$$\frac{\Delta \vdash d \to \Delta_0 \qquad \Delta[\Delta_0] \vdash c}{\Delta \vdash d; c \to}$$

Rispettare i Vincoli Anche dell'Ambiente Aggiornato

Ricordarsi di Aggiornarlo

SEMANTICA DINAMICA:

$$\rho \vdash \langle c, \delta \rangle \to \langle c, \delta' \rangle$$

Potenzialmente Modificata

① ELABORO d:

$$\frac{\rho \vdash \langle d, \delta \rangle \to^* \langle \rho_0, \delta' \rangle}{\rho \vdash \langle d; c, \delta \rangle \to \langle \rho_0; c, \delta' \rangle}$$

ρ_0: AMBIENTE DINAMICO COSTRUITO DA d

② ESEGUO C

$$\frac{\rho[\rho_0] \vdash \langle c, \delta \rangle \to \langle c', \delta' \rangle}{\rho \vdash \langle \rho_0; c, \delta' \rangle \to \langle \rho_0, c', \delta' \rangle}$$

③ COMANDO HA TERMINATO → ANCHE BLOCCO HA TERMINATO

$$\frac{\rho[\rho_0] \vdash \langle c, \delta \rangle \to \delta'}{\rho \vdash \langle \rho_0; c, \delta \rangle \to \delta'}$$

(Nota che ρ_0 è locale al Blocco)

ESECUZIONE ed EQUIVALENZA:

$Exec: Com \times Mem \rightarrow Mem$

$Exec(c, \sigma) = \sigma'$ sse $\vdash \langle c, \sigma \rangle \rightarrow^* \sigma'$

Senza Ambiente EXT.

EQUIVALENZA: $\equiv \subseteq Com \times Com$

$c_1, c_2 \in Com$

$$c_1 \equiv c_2 \text{ sse } \forall \sigma \; Exec(c_1, \sigma) = Exec(c_2, \sigma)$$

Indipendentemente dalla Memoria ottengo lo Stesso Risultato, la funzione Calcolata Ψ di Fondamenti e per Questo so che Questa è la prima Equivalenza NON decidibile per via del CICLO WHILE

Un EXEC può NON Terminare (per CICLO WHILE) visto che il CICLO WHILE porta ad un ALtro WHILE e se la guardia fosse Sempre Vera allora potenzialmente divergo/ NON Termino MAI

ESEMPIO SEMANTICA BLOCCO:

$\underbrace{\text{var } x:\text{int} = 3}_{d} \; ; \; \underbrace{x := x+1}_{c}$

da DIM

SEMANTICA STATICA:

$$\frac{\vdash \text{var } x:\text{int} = 3 : \Delta_0 \quad (?) \qquad \Delta_0 \vdash x := x+1 \quad (?)}{\vdash \text{var } x:\text{int} = 3; \; x := x+1}$$

$$\frac{\vdash 3 : \text{int}}{\vdash \text{var } x:\text{int} = 3 : [x \mapsto \text{int}_{Loc}] \equiv \Delta_0}$$

$$\frac{\Delta_0 \vdash x+1 = \text{int} \quad (?)}{\Delta_0 \equiv [\text{int}_{Loc}] \vdash x := x+1}$$

$\Delta_0(x) = \text{int}_{Loc}$

$$\frac{\Delta_0 \vdash x : \text{int} \qquad \Delta_0 \vdash 1 : \text{int}}{\Delta_0 \vdash x+1 : \text{int}}$$

$\Delta_0(x) = \text{int}_{Loc}$

→ Viene SUBITO Usato l'AMBIENTE Creato

Semantica Dinamica:

$\rho \vdash \langle d; c, \phi \rangle \longrightarrow ?$

$\delta = \phi$

① $\dfrac{\rho \vdash \langle \text{Var } x:int = 3, \delta \rangle \longrightarrow^* ? \langle \rho_0, \delta' \rangle ?}{\rho \vdash \langle \text{Var } x:int = 3; c \rangle \longrightarrow \langle \rho_0; c, \delta' \rangle}$

Elaboro dichiarazione ma c'è già assioma (un Valore)

$$\frac{\rho \vdash \langle 3, \delta \rangle \longrightarrow}{\rho \vdash \langle \text{Var } x:int = 3; c \rangle \longrightarrow \langle \underbrace{[x \mapsto l_x]}_{\rho_0}, \underbrace{[l_x \mapsto 3]}_{\delta'} \rangle}$$

Tornando ad ①

$$\frac{}{\rho \vdash \langle \text{Var } x:int = 3; c \rangle \longrightarrow \langle \rho_0;}$$

② $\dfrac{\rho_0 \vdash \langle x := x+1, \delta' \rangle \longrightarrow ?}{}$