

IMPLEMENTAZIONE SCOPING:

■ Statico \Rightarrow Catena Statica

Raccolgo INFO a tempo di Compilazione (dal codice), riesco
poi anche ad EVITARE la RICERCA.

STRUTTURA ANNIDAMENTO

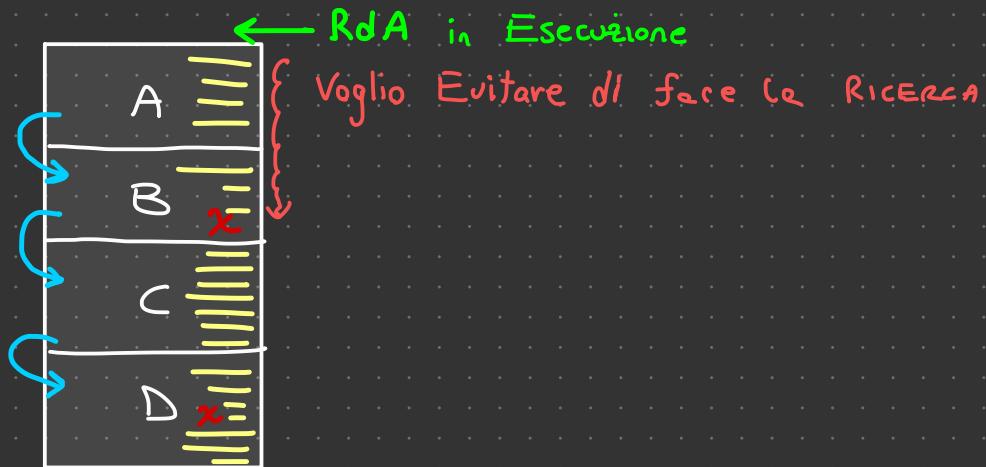
DELLE DEFINIZIONI

Sapendo Quanto devo Risalire

■ Scoping Dinamico \Rightarrow Per RIDURRE TEMPO AUMENTO MEMORIA
con Strutture dati Esterne (DA TENERE AGGIORNATA)

CRT \Rightarrow Tabella Centrale dei Riferimenti, garantisce/Mantiene
aggiornata l'associazione TRA ID \leftrightarrow AMBIENTE.

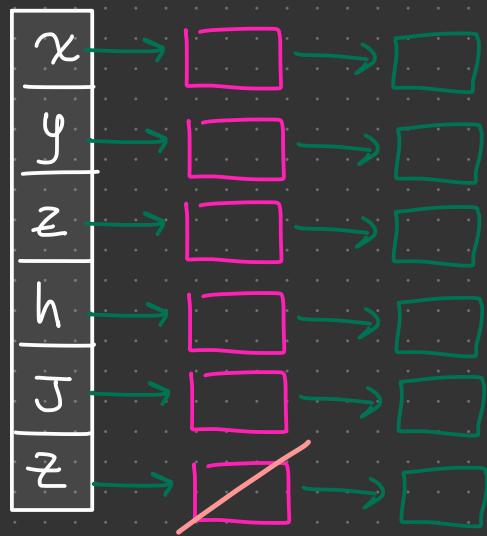
Costantemente devo Avere Accessibile l'ultimo Riferimento



Creo una Struttura Dati che Sfrutta lo SHALLOW ACCESS

Prendo un ARRAY DI ELEMENTI INDICIZZATI sul Nome delle Variabili.

Ogni ENTRY punta ad una lista Cinkata di Elementi, che ognuno Rappresenta/describe un AMBIENTE in cui la Variabile è definita



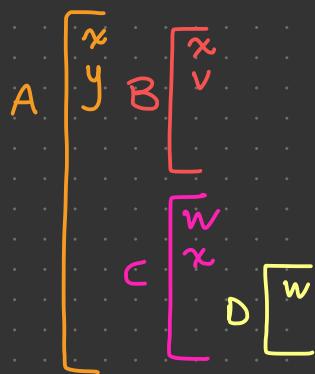
Sono TUTTI ambienti Attivi per cui l'Ambiente in TESTA è l'ultimo ARRIVATO (per EVITARE RICERCHE TROPPO LUNGHE) che è Quello di RIFERIMENTO per lo SCOPING DINAMICO

Così NON devo fare SCANSIONI \Rightarrow Tempo di Accesso Costante Garantito (ALLE INFO CHE MI PERMETTONO DI RECUPERARE LA VARIABILE)

Elimina la procedura ed Imposta Come ULTIMO ambiente il NEXT della PROCEDURA.

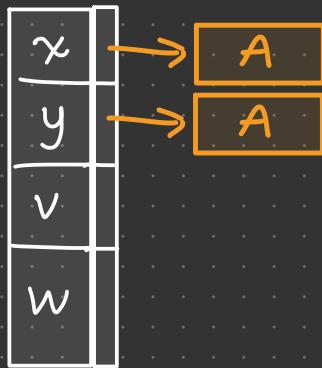
ESEMPIO:

$A \rightarrow B \rightarrow C \rightarrow D$

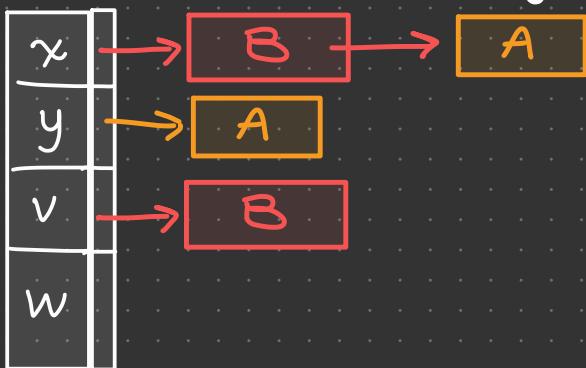


CRT:

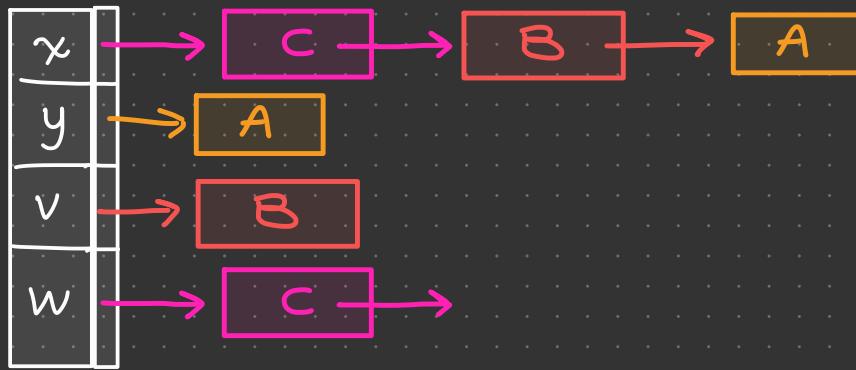
Quando Eseguo A, defini Sce x e y



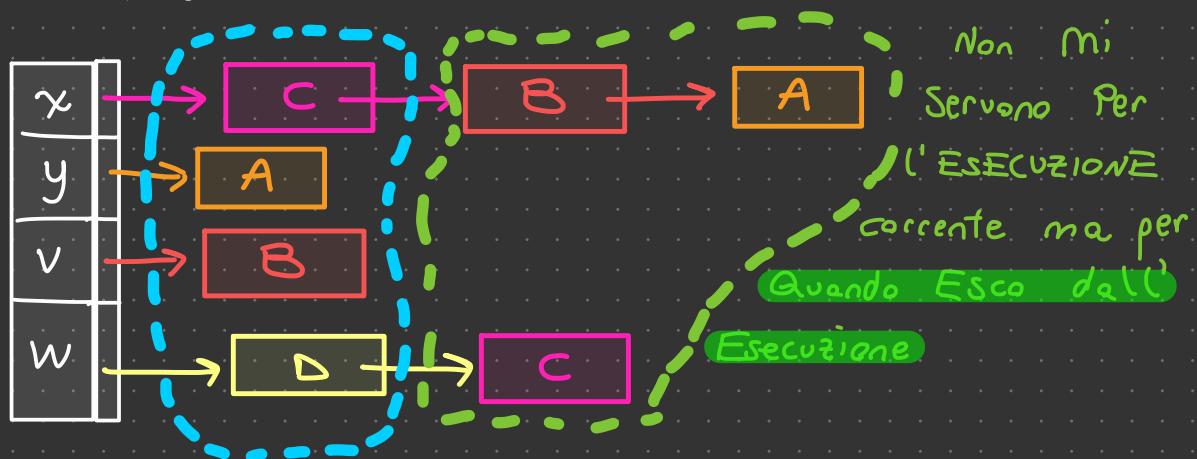
Passo Ad Eseguire B:



Poi Passo a C:



Ed in fine a D:



Utili per l'ESECUZIONE CORRENTI

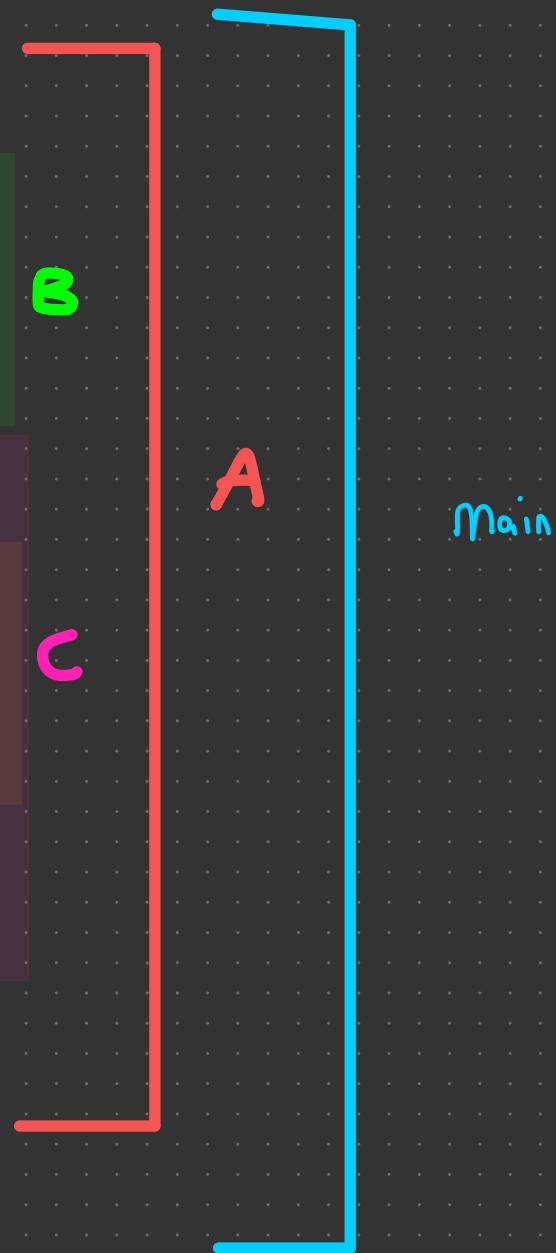
Mantenerla ha dei COSTI, AGGIORNAMENTO ed ELIMINAZIONE
ma CONVIENE per via dell' ACCESSO DIRETTO (dipende Sempre poi
dal programma)

Se Referenzio Molte Volte Sarà Sempre Vantaggioso

ESEMPIO:

```
{int b;  
Void A () {  
    int x=0, y=0, z=5;  
    void B () {  
        int x=3, w=3;  
        x = y+z;  
    }  
}
```

```
Void C () {  
    int y=1, v=1;  
    Void D () {  
        int z=2, v=2;  
        B ();  
        v=x+y;  
    }  
    D ();  
    x = z+y;  
}  
C ();  
}  
A ();  
}
```



b
x
y
z
v
w

main ≡

Chiamata di A

b
x
y
z
v
w

main ≡

A, 0

A, 0

A, 5

Chiamata C(?)

b
x
y
z
v
w

main ≡

A, 0

C, 1 → A, 0

A, 5

C, 1

Chiamata di D()

b
x
y
z
v
w

main ≡

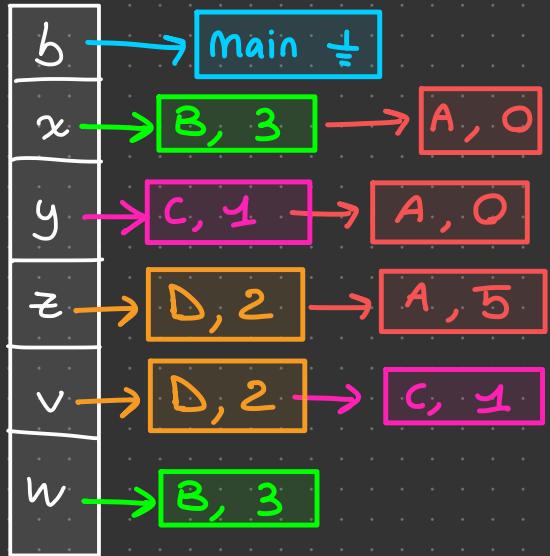
A, 0

C, 1 → A, 0

D, 2 → A, 5

D, 2 → C, 1

Chiama B() da D:

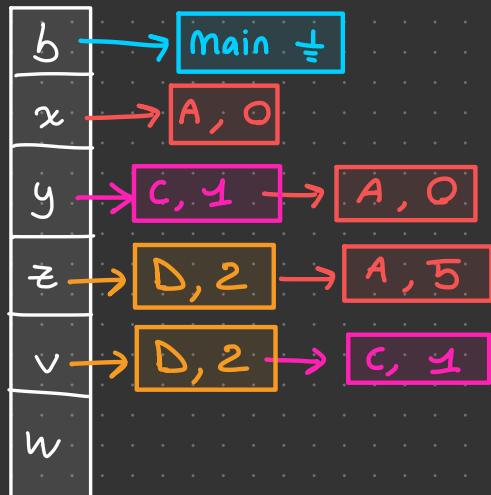
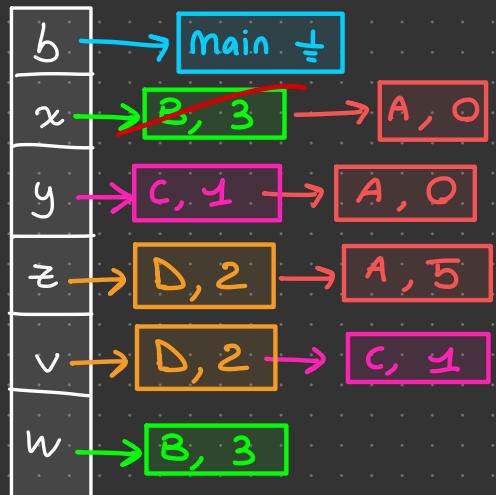


Prendo da C → 1

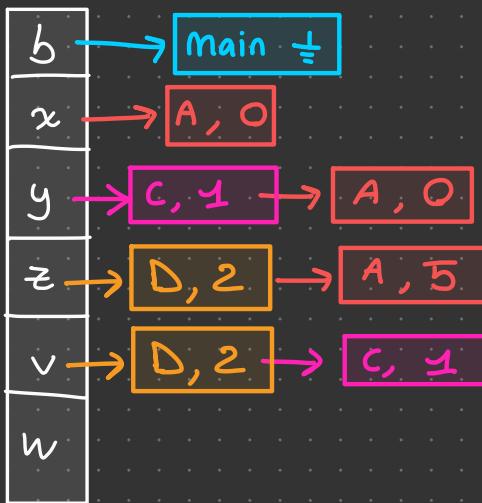
Dopo l'EXEC di $x = y + z$ ↗ Prendo da D → 2

che mi porta a Riscrivere 3 (non c'è nessuna modifica)

Ora Termina B quindi:

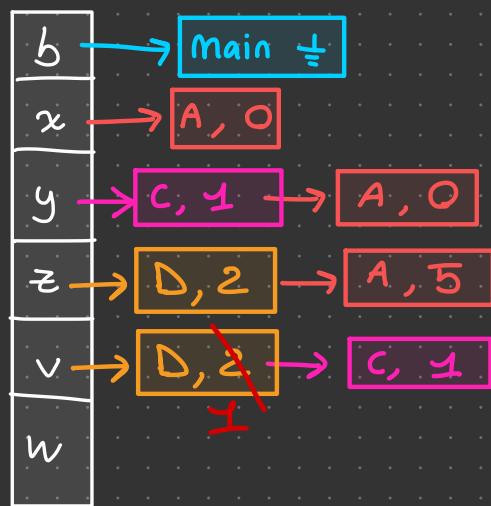


Torno in D:

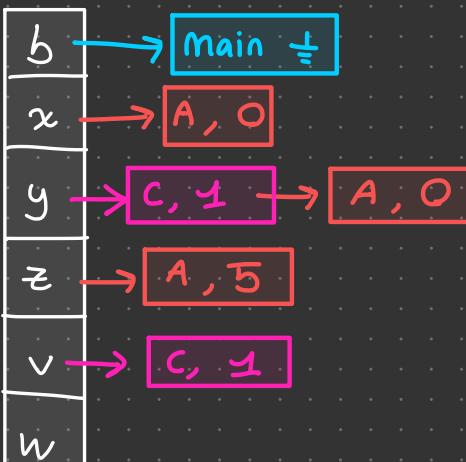


$$v = y + x \quad \begin{array}{l} \nearrow C \rightarrow 1 \\ \searrow A \rightarrow 0 \end{array}$$

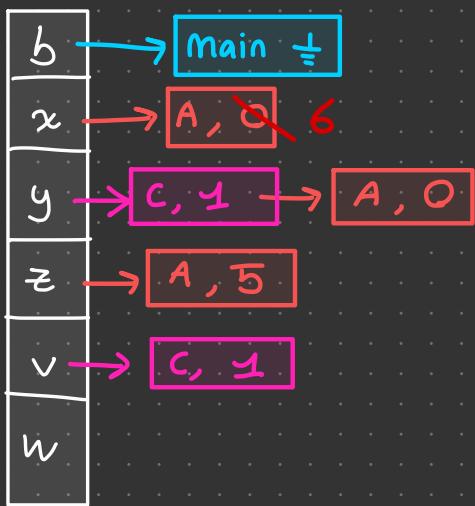
E Quindi:



Termino D e Siamo in C



$$x = z + y \rightarrow C \rightarrow 1 \quad \begin{array}{l} \nearrow A \rightarrow 5 \\ \searrow \end{array}$$



Il Resto delle chiamate distrugge le liste fino a Smotterle
Completamente (CRT Vuota)

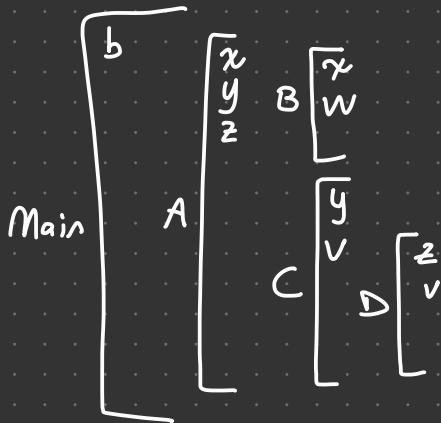
PERCORSO SULLO STACK (scoping Statico):

$Sd(\text{main}) = 0$

$Sd(A) = 1$

$Sd(B) = Sd(C) = 2$

$Sd(D) = 3$



ESECUZIONE MAIN:

main	
b	≠

Chiamata di A

A	
x	0
y	0
z	5

Chiamata di C(7)

C	
a	≠
y	1
v	1

Link Statico: $K = Sd(\text{main}) - Sd(A) = 1 - 1 = 0$

$\text{Ls}(A) = \text{Main}$

Param. formale

$K = Sd(A) - Sd(C) + 1 - 1 = 0$

$\text{Ls}(C) = A$ (me lo Aspetto, C è definito dentro A)

Chiamo D()

D	
z	2
✓	2

$$K_D = Sd(C) - Sd(D) + 1 = 0$$

|

$$Sd(D) = C$$

Chiamo B()

B	
x	3
w	3

$$K_B = Sd(D) - Sd(B) + 1$$

|

$$= 2$$

|

$$LS(B) = LS(LS(D))$$

|

$$LS(B) = A$$

Eseguo in B: $x = y + z$ (x però è costruito, calcolo n per le altre 2 variabili)

$$\begin{aligned} Ny: \quad Sd(B) - Sd(A) + 1 &\Rightarrow 1 \\ Nz: \quad Sd(B) - Sd(A) + 1 &\Rightarrow 1 \end{aligned} \left. \begin{array}{l} \text{Per Trovare } y \text{ e} \\ z \text{ Risalgo di 1} \\ \text{e mi Trovo in A} \end{array} \right\}$$

$$x = 0 + 5 \Rightarrow \boxed{x = 5}$$

vado a Modificare

B	
x	5
w	3

Termino B e si Elimina il suo RDA e la situazione è Questa:

main	
b	1

A	
x	0
y	0
z	5

C	
a	7
y	1
v	1

D	
z	2
v	2
v	1

Devo Compiere l'ESECUZIONE di D ed Eseguo:

$$V = x + y$$

↳ Locale

$$Ny = Sd(D) - Sd(C) + 1 \\ = 1$$

$$Nx = Sd(D) - Sd(A) + 1 \\ = 2$$

$$V = 0 + 1 = 1$$

Sovrascrivo

Termina D ed Elimino RDA

main	
b	4
A	
x	6
y	0
z	5

C	
a	7
y	1
v	1

ESEGUEO C:

$$\chi = z + y$$

Riferito ad A

locale

$\rightarrow y \rightarrow 1$

$$N_z = S_d(C) - S_d(A) + 1 \\ = 1$$

$$\chi = 5 + 1 \text{ e vado a SOVRASCRIVERE}$$

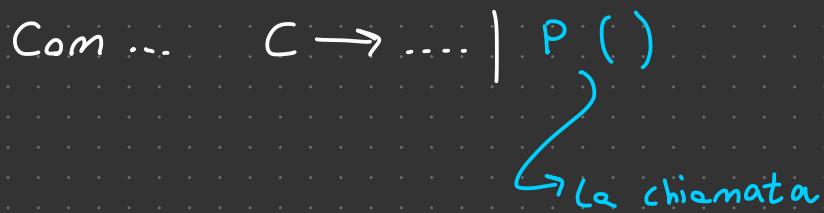
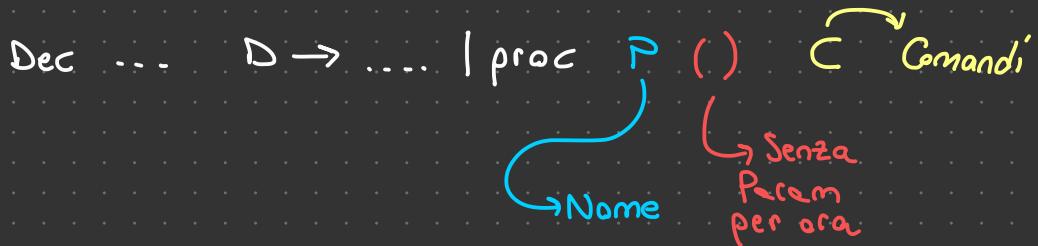
E via via Si Chiudono gli RDA fino al Main per Poi Svuotarsi Completamente

RECAP BREVE:

Nei linguaggi, per usare le procedure Abbiamo Bisogno

② Dichiarazione \Rightarrow Interfaccia \Rightarrow Nome, Parametri, Codice/Corpo

② Comando per la Chiamata \Rightarrow Eseguire Codice



Con Aggiunta dei parametri Avrò:

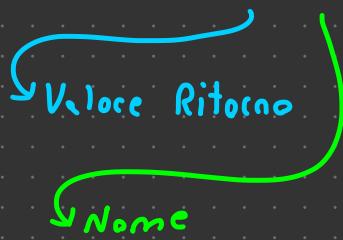


PARAMETRI DELLA PROCEDURA:

Permettono di far Corrispondere ad un UNICO CODICE più ESECZIONI.

C'è una **COMUNICAZIONE** tra chiamente e Chiamato

DICHIARAZIONE: int f(int n) {return $n+1$;}



Parametro formale che Quando uso Procedura avrà un Significato

Uso/CHIAMATA: $x = f(3)$; oppure $x = f(3+x)$

Parametro Attuale

r-Value

Direzione Comunicazione:

In Mode \Rightarrow dal Chiamente Verso Chiamato (Procedura),
Il Param. formale è Ambiente Locale e **riceve**
il Valore dell' Attuale

Out-Mode \Rightarrow dal Chiamato al Chiamente, Passaggio di
Parametro per RISULTATO (non USATI).

Out-Mode più comune è il **RETURN**.

In - OUT \Rightarrow Bidirezionale, ho Entrambi i TIP!, posso Separare i 2 Canali oppure no

MODELLO CONCETTUALE DI PASSAGGIO:

M Sposto Fisicamente il Valore su Nuove locazione
che è il Passaggio per NOME

M Per Riferimento, Trasferisco il link di ACCESSO al
Valore