ESERCIZI:

INDUZIONE STRUTTURALE

CASO BASE ⇒ Lo dimostro Mostrando che Vale Sull'Elemento che Nella definizione è All'Insieme

PASSO ⇒ devo Applicare Prima L'IPOTESI INDUTTIVA

↳ Suppongo Quello che devo dimostrare a degli Elementi più Complessi Rispetto al CASO BASE

devo Riuscire a Ricondurmi alla fora delle FORMULA che devo DIMOSTRARE (USARE VARIABILI PER RAGRUPPARE)

SCOPING STATICO Ambiente di RIFERIMENTO Stabilito a Tempo di Compilazione (Quello che CONTIENE definizione della Chiamata) Quindi è indipendente da dove si fa la CHIAMATA, è Sempre UGUALE.

Gestisco Tramite la Catena Statica

① Fare Schema Annidamento e Calcolare Sd() per TUTTI

② Catena delle Chiamate

③ LinK Dinamico è Sempre All' RDA Precedente

④ Calcolare $K_{chiamato} = Sd(Chiamante) - Sd(chiamato) + 1 \quad \forall chiamato$

⑤ Per Vaciabili NON locali Calcolare:

$N_{var} = Sd(Procedura\ che\ lo\ Usa) - Sd(Più\ Vicina\ che\ la\ definisce)$

→ #Volte che devo Risalire la CATENA STATICA da P per Trovare il Blocco Contiene la var di RIFERIMENTO

⑥

Ricordarsi che se UNA Procedura Termina NON Va Ricopiato RdA

SCOPING DINAMICO L'AMBIENTE di RIFERIMENTO è Quello Valido al Momento della CHIAMATA della Procedura (POTENZIALMENTE DIVERSO OGNI VOLTA)

Gestisco Tramite CRT (Tabella Centrale dei Rifecimenti)

① faccio un ARRAY con tutte le Variabili che Vengono USATE

② Ogni Volta che Viene Sovrascritta una definizione di una var. Aggiungo in Testa Alla Coda

③ Posso Sempre farmi lo Schema di ANNIDAMENTO per Capire che Procedura definisce Cosa.

Nel caso in cui ci fosse una Procedura passata come PARAM che ha dei RIFERIMENTI non Locali allora devo Anche analizzare le REGOLE DI BINDING:

① SHALLOW ⇒ "SCOPING DINAMICO" per le PROCEDURE, fa riferim. all' Ultimo AMBIENTE Creato (QUELLO CHE FA LA CHIAMATA)

② DEEP ⇒ "SCOPING STATICO" per le PROCEDURE, si UTILIZZA l' ambiente

RICORSIONE

① Dare DEFINIZIONE
- Ricorsiva SSE Richiama SE STESSA

② Dire Quando è RICORSIONE DI CODA
- Solo se viene RITORNATO il Valore della chiamata Ricorsiva Senza Ulteriori Rielaborazioni o Computazioni
- Utile perchè AUMENTA EFFICIENZA dello SPAZIO USATO
- Costruisce il RISULTATO ad ogni Chiamata e alla fine (giunto Alla BASE) lo RESTITUISCE.

③ Dire Che Cosa fa (fare TEST)

④ Proporre RC

⑤ Simularne le Chiamate per DIMOSTRARE il Risultato

Ricorsione in Coda ha SEMPRE più Parametri Rispetto Alla Ricorsione Classica

Mi Serve allocazione DINAMICA della MEMORIA, Visto che NON so a priori Quanta Memoria mi Serve e Quante Volte Verrà Chiamata la PROCEDURA RICORSIVA. Ma Anche l'IMPLEMENTAZIONE delle procedu per Permetterla

COMPLETAMENTO CODICE

1. Deve Essere ESEGUIBILE Quindi tutto Quello che si USA deve Essere DICHIARATO
2.
3.

Semantica Comandi / Istruzioni

Dinamica:

- Valuta l'Espressione
- Elabora la dichiarazione
- Trasformazione della Memoria

Statica

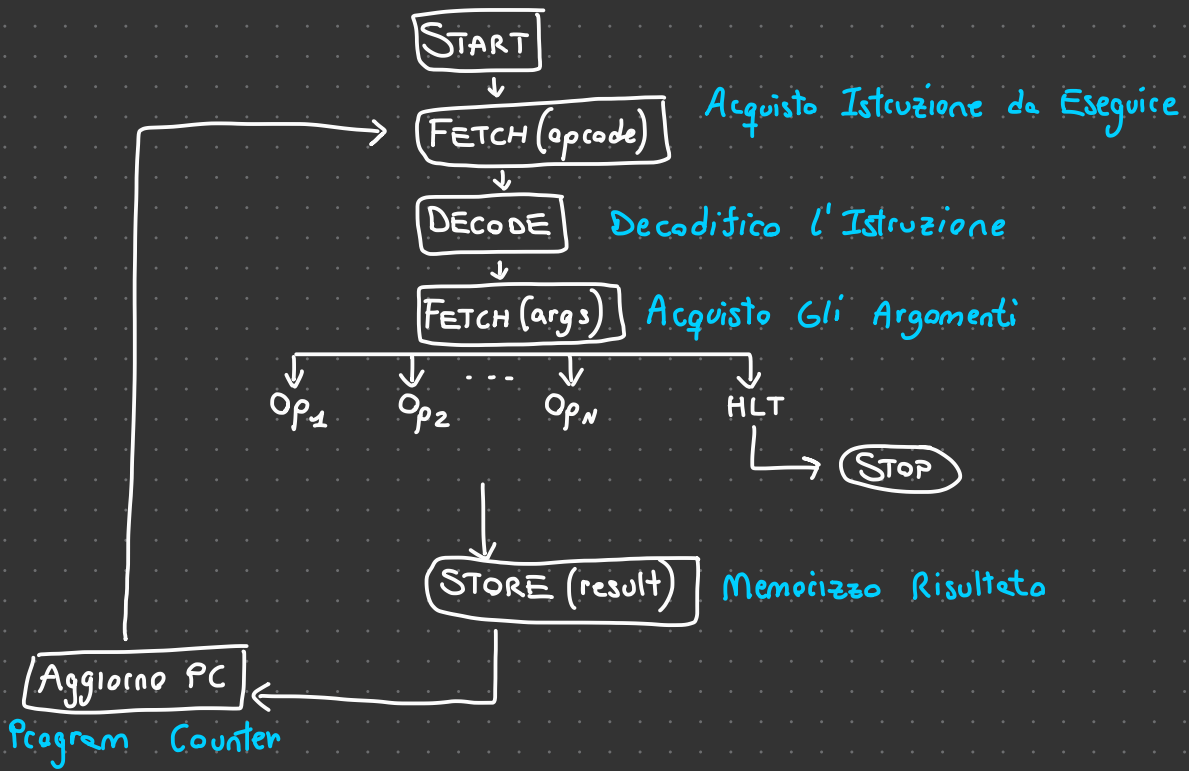
- Associa un tipo All' Espressione
- Assacia un Ambiente Statico per dichiarazioni
- Comando Sia ben formato

TEORIA:

INTERPRETE per un Linguaggio L Scritto in Lo è un Programma

$I^{LoL} : Prog^{L} \times D \to D \quad \forall.input \in D.\ I^{LoL}(P^{L}, input) = output$

$\forall.P^{L} \in Prog^{L}$

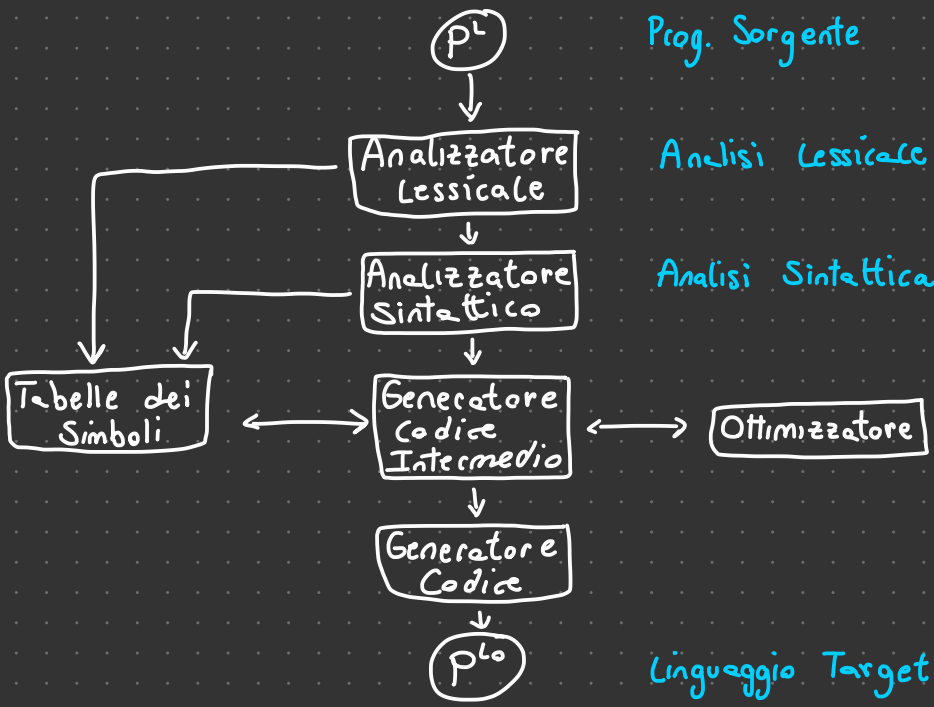


COMPILATORE da L in L_o è un Programma "TRADUTTORE" che:

Linguaggio Sorgente ↙ ↘ Linguaggio Target

$$C^{L L_o} : Prog^{L} \longrightarrow Prog^{L_o} \quad t.c.$$

$$\forall input \in D \quad Prog^{L}(input) = Prog^{L_o}(input)$$



Categorie Sintattiche

Espressioni denotano Valori, Vengono Valutate per Restituire un Valore, Sono Equivalenti se Restituiscono val=

Dichiarazioni denotano Richieste di modifica degli Ambienti, Vengono Elaborate per Ottenere la Trasformazione, Sono Equivalenti Se Producono Stesso Ambiente.

Occorrenze di:

- **Definizione** → Si Attribuisce Significato All'Id
- **Uso** → Si Accede al Suo Significato
- **Libera** → Non è Nello scope di una definizione

Aggiornamento Ambiente $\Rightarrow \rho[\rho'] = \rho''$

Composizione:

- Sequenziale → $\rho \vdash d1; d2 \rightsquigarrow \rho[\rho_1[\rho_2]]$
- Privata → $\rho \vdash d1 \text{ in } d2 \rightsquigarrow \rho[\rho_2]$ (ρ_1 Nascosto all'Esterno)

Comandi denotano Richieste di Modifiche alla memoria, Vengono Eseguiti, Sono equivalenti Se producono la Stessa memoria. (che Cattura Trasformazioni Irreversibili)

BINDING:

Se Siamo Nel CONTESTO Con SCOPING DINAMICO e Viene Passato una fz. Come PARAMETRO con RIFERIMENTI NON Locali.

- **DEEP BINDING** ⇒ Ambiente di RIFERIMENTO è Quello della creazione del legame Tra Attuale e formale

- **SHALLOW BINDING** ⇒ Ambiente Quando Viene Effettuata la Call è Quello Valido

Scoping Statica è SOTTOINTESO come DEEP BINDING

Differenza Tra Shallow e Deep è Come Tra Dinamico e Statico

Solo per i RIFERIMENTI NON Locali All' Interno di una Procedura