

## DETERMINARE L'AMBIENTE DI RIFERIMENTO:

### IN PROCEDURE CON AMB. NON LOCALE

- Regole di SCOPING
- Metodi Passaggi di Parametri
- Regole di BINDING → procede Come Parametri (Solo Pochi linguaggi)

### SCOPING:

/// Statico ⇒ Ambiente di Riferimento Nel contesto dove vado a definire la PROCEDURA (al Momento di definizione)

/// AMBIENTE RIFERIMENTO SEMPRE LO STESSO

/// SI RISALE LA CATENA STATICA DI DEFINIZIONE

/// Dinamico ⇒ Ambiente di Riferimento è Quello Valido al Momento della Chiamata della PROCEDURA

/// AMBIENTE DI RIFERIMENTO SEMPRE ≠ POTENZIALM.

/// SI RISALE LA CATENA DINAMICA DI CHIAMATA

### ESEMPI REALI:

/// Lisp Utilizza Scoping DINAMICO

/// Perl Se non Specifico Nulla CONSIDERA Var Globali, come se Avessi un UNICA Variabile che Sovrascrivo.

# ALLOCAZIONE DELLA MEMORIA:

Determinare Regole di:

- /// Allocazione fisica di Oggetti (VARIABILI, PROCEDURE,...)
- /// Come Risolvere (ALGORITMI) per riferimenti NON Locali

Dobbiamo Sapere da dove Arriviamo per poi TORNARCI una volta che Abbiamo finito di ESEGUIRE la procedura

→ ALLOCAZIONE STATICA ⇒ Tempo di "Compilazione", si determina Solo dal Codice la Quantità di memoria Necessaria

ALLOCAZIONE DINAMICA ⇒ Tempo di "Esecuzione", Quantità di Memoria Viene determinata durante l'ESECUZIONE Effettiva del Programma (PER TUTTI GLI OGGETTI: ID, PROCEDURE,...)

Allocazione STATICA non permette la RICORSIONE perchè NON so a PRIORI quante Volte ESEGUIRÒ

Allocazione Dinamica Sfrutta:

/// STACK → Gestito dal Linguaggio

/// HEAP → Gestito più dal Programmatore

## ALLOCAZIONE STATICA:

Ad Esempio COBOL/FORTRAN (appl. Bancario/Assicurative)

Memoria Rimane Allocata Per l'INTERA ESECUZIONE, Quindi non può Essere OTTIMIZZATA

Dobbiamo Conoscere a PRIORI Quanta Memoria Serve, non accetta RICORSIONE

No Blocchi ANNIDATI  $\Rightarrow$  Scoping Statico

Ad Esempio Var. Globali che HANNO un TEMPO DI VITA pari a tutta l'ESECUZIONE del PROGRAMMA

## ALLOCAZIONE DINAMICA:

A Tempo di Esecuzione l'INDIRIZZO FISICO e l'Area di Memoria da Utilizzare

Cosa mi Serve per OTTIMIZZARE Questo processo?

Per Quanto Riguarda le PROCEDURE Viene usata la PILA/STACK (per via del funzionamento LIFO)

Info Memorizzata = Record di Attivazione (che deve Essere Salvato Nello STACK)  $\Rightarrow$  Allocazione Richiede:

❧ Serie di Operazioni a CHIAMATA  $\Rightarrow$  fase di Apertura

❧ Serie di Operazioni di RITORNO  $\Rightarrow$  fase di Chiusura



Non Conosco a Priori l'Indirizzo ma  
SOLO la DIMENSIONE

DIMENSIONE Nota a Priori (TEMPO DI COMPILAZIONE)

## SEMANTICA DELLA SEQ. DI CHIAMATA:

- 1 Sicuramente Gestire il METODO DI PASSAGGIO DI PARAMETRI
  - Per Valore
  - Per Riferimento} Così So QUANTO devo Allocare
- 2 Si Alloca lo Spazio Sullo STACK per AMBIENTE LOCALE
- 3 Memorizzare INFO Necessarie per Continuare l'EXEC del Chiamante (AL RITORNO)
  - Indirizzo del
    - Risultato
    - Ritorno
- 4 Trasferisce il Controllo al Codice della procedura
- 5 Allocare lo Spazio per il Risultato e il Calcolo
- 6 Gestire le Regole di Scoping

## SEMANTICA DELLA SEQ. DI RITORNO:

- 1 Gestire il METODO di RESTITUZIONE del RISULTATO
- 2 Dealloca lo STACK
- 3 Recupero le INFO per lo Stato del Chiamante
- 4 Si Ritorna il Controllo

## IMPLEMENTAZIONE REGOLE SCOPING:

### STATICO:

- Devo Costruire la Catena Statica
- Display (STRUTTURA SPECIFICA)

### DINAMICO:

- Catena Dinamica (NON EFFICIENTE DOVER RICERCARE IN MODO NAIVE)
- CRT (Tabella Centrale del Riferim.)  
una Struttura dati per un'implementazione Efficiente

