CONFRONTO TRA I SUPPORTI A RUN TIME DI VARI LINGUAGGI

Entità presenti quando un programma va in esecuzione

- Programmi d'utente (compilati)
- Routines del supporto
  - interprete
  - I/O, librerie, routines per la gestione delle altre strutture, garbage collector
- Strutture dati per gestire le attivazioni (funzioni, procedure, classi)
  - ambiente
  - memoria
  - temporanei
  - punti di ritorno

FORTRAN

- Caratteristiche del linguaggio
  - permette compilazione separata dei sottoprogrammi
  - non permette ricorsione, struttura a blocchi, procedure annidate
  - **ambiente e memoria locali sono statici**
  - **non esiste ambiente non locale**
- Gestione completamente statica
  - il compilatore crea, per ogni sottoprogramma, una "unità compilata" che contiene
    - ✓ codice compilato
    - ✓ il punto di ritorno
    - ✓ l'area dati locali (ambiente + memoria)
    - ✓ temporanei
  - linker e loader risolvono i riferimenti globali e allocano in memoria
    - ✓ tutte le unità necessarie
    - ✓ le (poche e semplici) routines del supporto a tempo di esecuzione
      - – input-output, operazioni matematiche,...

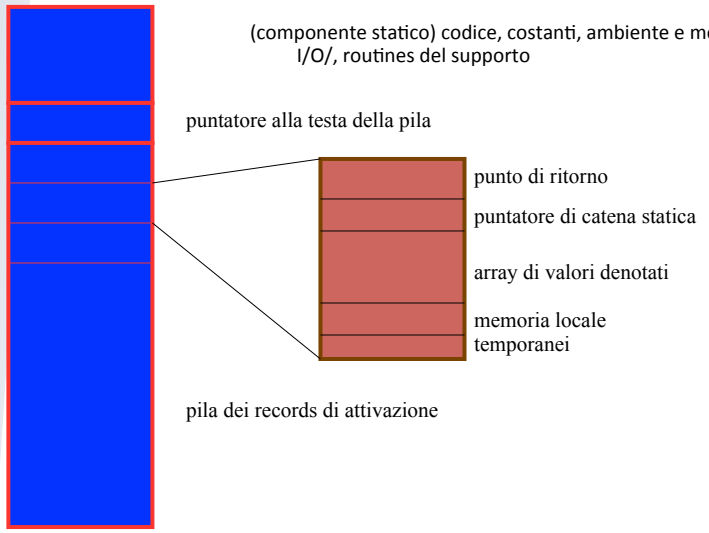
3

ALGOL

- Caratteristiche del linguaggio
  - il programma è **un unico blocco, con blocchi e procedure annidati**
    - ✓ non permette compilazione separata dei sottoprogrammi
  - permette la ricorsione
  - ambiente e memoria locali dinamici (anche statici, se dichiarati tali)
  - **scoping statico**
  - **non permette puntatori**
- Semplice gestione dinamica basata sulla pila dei records di attivazione
- Il compilatore genera
  - il codice per l'intero programma
    - ✓ incluso quello per la generazione (a tempo di esecuzione) dei record di attivazione
  - costanti
  - l'area dati locali statica (ambiente + memoria)
- un record di attivazione contiene
  - punto di ritorno (puntatore di catena dinamica)
  - puntatore di catena statica
  - ambiente e memoria locali (senza nomi, inclusi i parametri formali)
  - temporanei

4

ALGOL: struttura della memoria



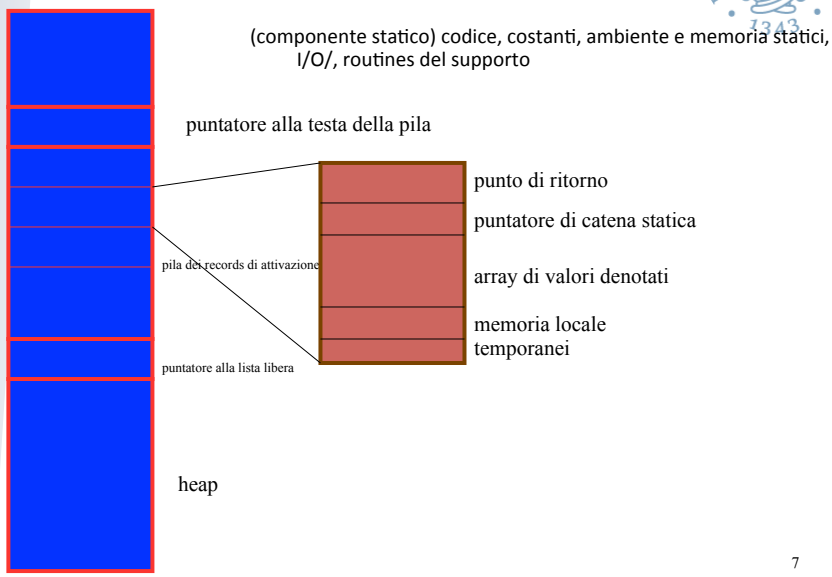
5

PASCAL

- Caratteristiche del linguaggio
  - il programma è un unico blocco, con blocchi e procedure annidati
    - ✓ non permette compilazione separata dei sottoprogrammi
  - permette la ricorsione
  - ambiente e memoria locali dinamici (anche statici se dichiarati tali)
  - scoping statico
  - **permette puntatori**
- Pila dei records di attivazione **+ heap**
- Il compilatore genera
  - il codice per l'intero programma
    - ✓ incluso quello per la generazione (a tempo di esecuzione) dei record di attivazione
  - costanti
  - l'area dati locali statica (ambiente + memoria)
- Un record di attivazione contiene
  - punto di ritorno (puntatore di catena dinamica)
  - puntatore di catena statica
  - ambiente e memoria locali (senza nomi, inclusi i parametri formali)
  - temporanei
- **Heap senza garbage collector**

6

PASCAL: struttura della memoria



7

C

- Caratteristiche del linguaggio
  - **il programma è composto da moduli compilati separatamente**
  - permette la ricorsione
  - ambiente e memoria locali dinamici (anche statici se dichiarati tali)
  - scoping statico
  - permette puntatori
- Pila dei records di attivazione
  - un record di attivazione contiene
    - ✓ punto di ritorno (puntatore di catena dinamica)
    - ✓ puntatore di catena statica
    - ✓ ambiente e memoria locali (senza nomi, inclusi i parametri formali)
    - ✓ temporanei
- Heap senza garbage collector
- Come PASCAL, con compilazione separata

8

Java (JVM)

- Caratteristiche del linguaggio
  - il programma consiste di un **insieme di classi compilate separatamente**
  - permette la ricorsione
  - ambiente e memoria locali dinamici (anche statici se dichiarati tali)
  - scoping statico (per i blocchi)
  - **oggetti e puntatori**
- **Stack machine con pila dei records di attivazione + heap per gli oggetti**
- Il compilatore genera, per ogni classe,
  - il codice compilato
    - ✓ incluso quello per la generazione (a tempo di esecuzione) degli oggetti
    - ✓ e quello per la generazione (a tempo di esecuzione) dei record di attivazione dei metodi
  - l'area dati locali statica (ambiente + memoria relativi alle dichiarazioni static)
- Un record di attivazione contiene
  - punto di ritorno (puntatore di catena dinamica)
  - ambiente e memoria locali (senza nomi, inclusi i parametri formali)
  - temporanei
- **Heap con garbage collector**

9

C# (.NET)

- Caratteristiche simili a Java …
- Stack per i record di attivazione
  - Possibilità di gestire **run-time stack con record di attivazione eteorogenei (C call, C# call, …)**
- Meccanismi più strutturati per la gestione del class loading (**assembly**)
- Meccanismi avanzati di supporto all'interoperabilità tra linguaggi differenti

10

ML

- Caratteristiche del linguaggio
  - il programma è un insieme di **definizioni di funzioni compilabili separatamente**
  - permette la ricorsione
  - ambiente locale dinamico
  - scoping statico
  - **valori di ordine superiore**
- Pila dei records di attivazione **+ heap (per i termini e le liste)**
- Il compilatore genera
  - il codice per ogni funzione
    - ✓ incluso quello per la generazione (a tempo di esecuzione) dei record di attivazione
- Un record di attivazione contiene
  - punto di ritorno (puntatore di catena dinamica)
  - puntatore di catena statica
  - ambiente locale (senza nomi, inclusi i parametri formali)
  - temporanei
  - **può essere necessario effettuare la retention di records di attivazione**
- **Heap con garbage collector**

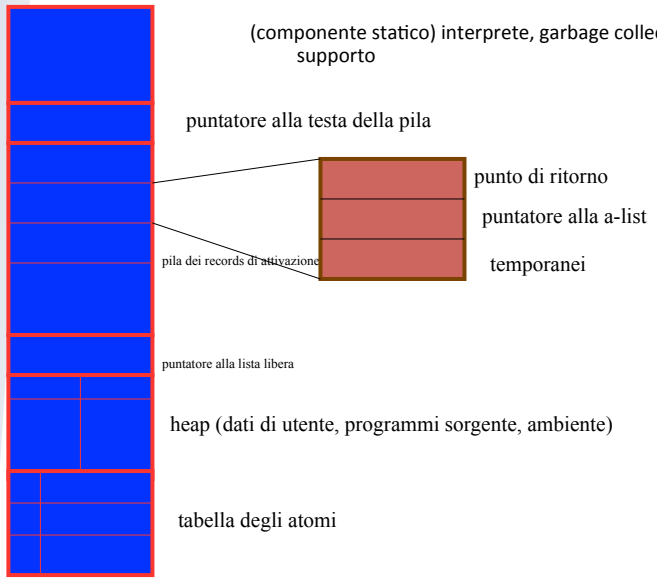
11

LISP

- Caratteristiche del linguaggio: simile ad ML, ma ...
  - **scoping dinamico**
- Le definizioni di funzioni (ed il loro codice compilato) sono associate al nome della funzione in **una tabella degli atomi**
  - una specie di ambiente globale
- Pila dei records di attivazione + **heap (per le s-espressioni)**
- Un record di attivazione dovrebbe contenere
  - punto di ritorno (puntatore di catena dinamica)
  - ambiente locale (con i nomi)
  - temporanei
- **Ma la pila di ambienti locali (a-list) è rappresentata come s-espressione e risiede nella heap**
  - il record di attivazione contiene un puntatore alla a-list
- Heap con garbage collector

12

LISP: struttura della memoria



13