



## STRUTTURE DATI NEL SUPPORTO A RUN TIME

1

### 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

2

# 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

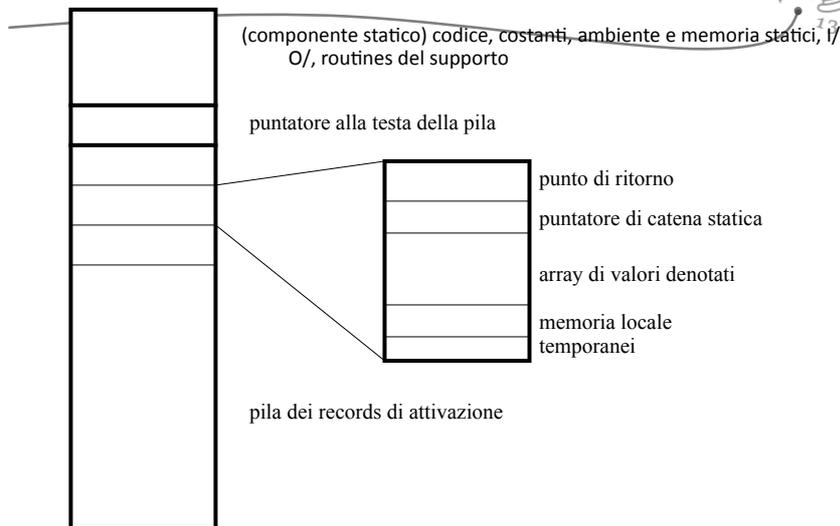
- il codice per l'intero programma
  - ✓ incluso quello per la generazione (a tempo di esecuzione) dei record di attivazione
- costanti
- l'area dati locali statica (ambiente + memoria)

## 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

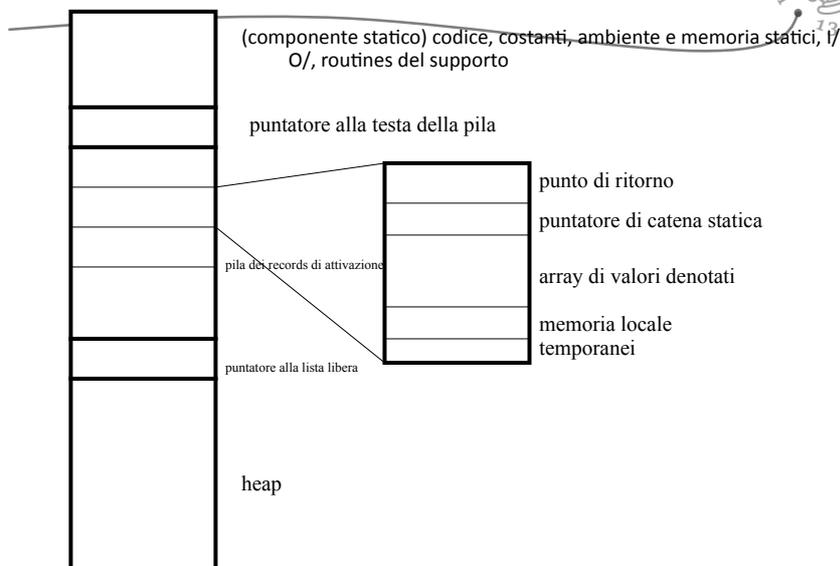
## PASCAL



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

6

## PASCAL: struttura della memoria



## 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



- ✦ 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
- ✦ 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

# 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

10

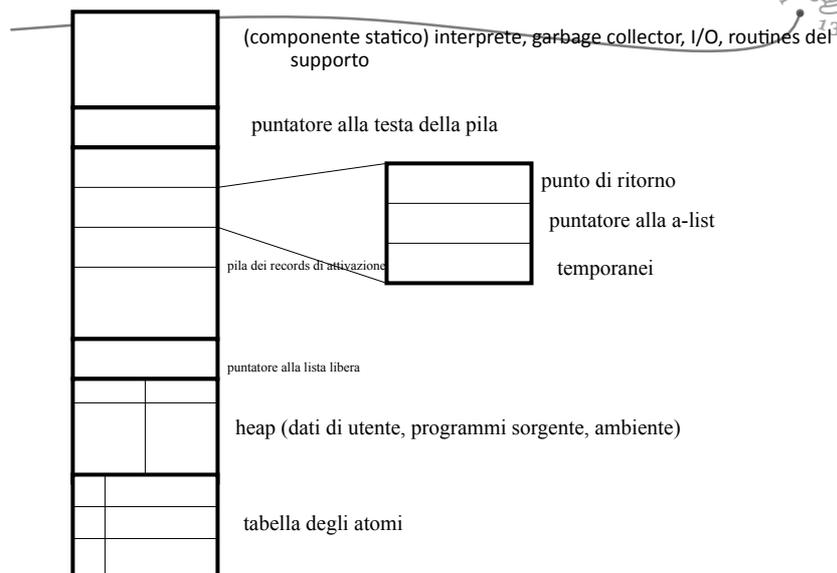
# LISP



- ✦ 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

11

## LISP: struttura della memoria



12