Programmazione e Algoritmica - corso C - 2022-01-25

Esercizio 1. [p.ti 3]
Scrivere un frammento di programma in L che mostri la differenza tra scoping statico e scoping dinamico. Costruirne poi il sistema di transizione e dimostrare la transizione che evidenzia la differenza.

Esercizio 2. [p.ti 3]
Progettare un algoritmo efficiente e ricorsivo in coda che determini l'indice dell'elemento minimo in un array ciclicamente ordinato. Calcolarne la complessità.

A array si dice CICLICAMENTE ORDINATO $\Leftrightarrow$
$\exists\, i : 1 \le i \le n \quad A[i] < A[i+1] < \ldots < A[n] < A[1] < \ldots < A[i-1]$

Esercizio 3. [p.ti 1]
Fornire una definizione di ricorsione in coda.

Esercizio 4. [p.ti 1]
Enunciare il Master Theorem.

SINTASSI DI L (BNF)

C ::= nil | **Id** = **E** | **C**;**C** | if (**E**) {**C**} *[else {**C**}]* | while (**E**) {**C**} | **D**;**C** | return **E** |
do {**C**} while (**E**) | for (**D**; **E**; **C**) {**C**} | switch (**E**) {**cList**}
E ::= v | **Id** | **uop E** | **E bop E** | (**E**) | **id**(**ae**)
D ::= nil | let **Id** *[:**T**]* = **E** | var **Id** *[:**T**]* = **E** | **D**;**D** | func **Id**(**form**) -> **T** {**C**; return **E**} | **form** = **ae** |
rec **D** | ρ
cList ::= case **Val**: **C**; break **cList** | case **Val**: **C**; break | default: **C**; break
Val ::= $\mathbb{N} \cup \mathbb{Z} \cup \mathbb{R} \cup \{\text{true, false}\} \cup \{s \mid s \in \text{ASCII}^*\}$
T ::= Int | Double | Bool | String
form := nil | let **Id**:**T**, **form** | var **Id**:**T**, **form**
ae ::= nil | **E**, **ae** | **Loc**, **ae**
uop ::= + | - | !
bop ::= + | - | * | \ | % | == | != | > | >= | < | <= | && | || | ·
Id ::= insieme degli identificatori validi
Loc ::= insieme delle locazioni

METAVARIABILI			
C	C, C', C0, C1, …	**T**	T, T', T1, T2, …
E	E, E', E0, E1, …	**Val**	v, v', v0, v1,…
D	D, D', D0, D1, …	**Int**	n, n', n0, n1, …
form	form, form', form0, form1,…	**Double**	d, d', d0, d1, …
ae	ae, ae', ae0, ae1, …	**Bool**	b, b', b0, b1, …
Id	Id, Id', Id1, Id2, …	**String**	s, s', s0, s1, …

SEMANTICA STATICA

Ambiente statico: $\Delta : \text{Id} \cup \text{Val} \to \text{T} \cup \text{TLoc}$

$$\Delta[\Delta'](x) = \begin{cases} \Delta'(x) & \text{if } \Delta'(x) \text{ defined} \\ \Delta(x) & \text{otherwise} \end{cases}$$

ESPRESSIONI formato: $\Delta \vdash_E E : T$
Assiomi:
$(A1)\ \varnothing \vdash_E n : \text{Int}$ $\quad (A2)\ \varnothing \vdash_E d : \text{Double}$ $\quad (A3)\ \varnothing \vdash_E b : \text{Bool}$ $\quad (A4)\ \varnothing \vdash_E s : \text{String}$
Regole di Inferenza:

$$(R1)\ \frac{\Delta(\text{Id}) = \text{T} \vee \Delta(\text{Id}) = \text{TLoc}}{\Delta \vdash_E \text{Id:T}} \qquad (R2)\ \frac{\Delta \vdash_E \text{E:T1, uop:T1} \to \text{T}}{\Delta \vdash_E \text{uop E:T}} \qquad (R3)\ \frac{\Delta \vdash_E \text{E:T}}{\Delta \vdash_E \text{(E):T}}$$

$$(R4)\ \frac{\Delta \vdash_E \text{E1:T1, E2:T2, bop:T1xT2} \to \text{T}}{\Delta \vdash_E \text{E1 bop E2:T}} \qquad (R5)\ \frac{\Delta \vdash_{ae} \text{ae:aet}, \Delta(\text{Id}) = \text{aet} \to \text{T}}{\Delta \vdash_E \text{Id(ae):T}}$$

COMANDI formato: $\Delta \vdash_C C$
Assiomi: $(A2)\ \varnothing \vdash_C \text{nil}$
Regole di Inferenza:

$$(R6)\ \frac{\Delta(\text{Id}) = \text{TLoc}, \Delta \vdash_E \text{E:T}}{\Delta \vdash_C \text{Id=E}} \qquad (R7)\ \frac{\Delta \vdash_C \text{C1}, \Delta \vdash_C \text{C2}}{\Delta \vdash_C \text{C1;C2}} \qquad (R8)\ \frac{\Delta \vdash_E \text{E:Bool}, \Delta \vdash_C C}{\Delta \vdash_C \text{while (E) do\{C\}}}$$

$$(R9)\ \frac{\Delta \vdash_E \text{E:Bool}, \Delta \vdash_C \text{C1}, \Delta \vdash_C \text{C2}}{\Delta \vdash_C \text{if (E)\{C1\}else\{C2\}}} \qquad (R10)\ \frac{\vdash_D \text{D:}\Delta', \Delta \vdash_D D, \Delta[\Delta'] \vdash_C \text{C}}{\Delta \vdash_C \text{D;C}}$$

$$(R11)\ \frac{\Delta \vdash_E \text{E:T}}{\Delta \vdash_C \text{return E}}$$

DICHIARAZIONI formato: $\vdash_D D : \Delta$ (costruzione) e $\Delta \vdash_D D$ (validazione)

Assiomi costruzione:

$(A3)\ \vdash_D$ nil:$\varnothing$ $(A4) \vdash_D$ const Id:T=E : [Id:T] $(A5) \vdash_D$ var Id:T=E : [Id:TLoc]

$(A6)\ \vdash_D$ func Id(form) $\rightarrow$ T{var res:T = E;C;return E} : [(Id,$\mathscr{T}$(form) $\rightarrow$ T)] dove

$$\mathscr{T} = \begin{cases} \mathscr{T}(\text{nil}) = \text{nil} \\ \mathscr{T}(\text{const Id:T, form}) = \text{T, form} \\ \mathscr{T}(\text{var Id:T, form}) = \text{T, form} \end{cases}$$

Regole di Inferenza costruzione:

$$(R12)\ \frac{\vdash_D \text{D1:}\Delta 1,\ \vdash_D \text{D2:}\Delta 2}{\vdash_D \text{D1;D2:}\Delta 1[\Delta 2]} \qquad (R13)\ \frac{\vdash_D \text{D:}\Delta}{\vdash_D \text{rec D:}\Delta}$$

Assiomi validazione: $(A7)\ \Delta \vdash_D$ nil

Regole di Inferenza validazione:

$$(R14)\ \frac{\Delta \vdash_E \text{E:T}}{\Delta \vdash_D \text{const Id:T = E}} \qquad (R15)\ \frac{\Delta \vdash_E \text{E:T}}{\Delta \vdash_D \text{var Id:T = E}}$$

$$(R16)\ \frac{\vdash_D \text{D1:}\Delta 1, \Delta \vdash_D \text{D1},\ \Delta[\Delta 1] \vdash_D \text{D2}}{\Delta \vdash_D \text{D1;D2}}$$

$$(R17)\ \frac{\vdash_D \text{D:}\Delta', \Delta[\Delta'_{|I_0}] \vdash_D \text{D}}{\Delta \vdash_D \text{rec D}}, I_0 = FI(\text{D}) \cap BI(\text{D})$$

$$(R18)\ \frac{\text{form:}\Delta 0,\ \Delta[\Delta 0] \vdash_C \text{var res:T=E; C; return res}}{\Delta \vdash_D \text{func Id(form)} \rightarrow \text{T\{var res:T=E; C; return res\}}}$$

FORMALI formato: form : Δ

Assiomi costruzione: $(A8)$ nil:$\varnothing$

Regole di Inferenza:

$$(R19)\ \frac{\text{form:}\Delta, \text{Id} \notin \Delta}{\text{const Id:T : }\Delta[(\text{Id,T})]} \qquad (R20)\ \frac{\text{form:}\Delta, \text{Id} \notin \Delta}{\text{var Id:T : }\Delta[(\text{Id,TLoc})]}$$

ATTUALI formato: $\vdash_D D : \Delta$ (costruzione) e $\Delta \vdash_D D$ (validazione)

Assiomi costruzione: $(A9)\ \Delta \vdash_{ae}$ nil

Regole di Inferenza:

$$(R21)\ \frac{\Delta \vdash_E \text{E:T}, \Delta \vdash_{ae} \text{ae:aet}}{\Delta \vdash_{ae} \text{E, ae : T, aet}}$$

SEMANTICA DINAMICA (ERRATA CORRIGE)

La regola FD1 diventa:

$$\langle \text{func Id(form) -> T \{C\}}, \rho, \sigma\rangle \rightarrow_D \langle [Id, \lambda\text{form . }\{\rho'; C\}], \sigma\rangle, \quad \rho' = \begin{cases} \rho_{|FI(C)-\text{form}} & \text{scoping statico} \\ \varnothing & \text{scoping dinamico} \end{cases}$$

Regola mancante:

$$\langle \text{return E}, \rho, \sigma\rangle \rightarrow_C \langle \text{E}, \rho, \sigma\rangle$$

Semantica Dinamica

esecuzione **C**: $\langle C,\rho,\sigma\rangle \longrightarrow_c \langle C',\rho',\sigma'\rangle$, $\mathbf{Exec}(C,\rho,\sigma)=\sigma' \iff \langle C,\rho,\sigma\rangle \longrightarrow_c^* \sigma'$

valutazione **E**: $\langle E,\rho,\sigma\rangle \longrightarrow_e \langle E',\rho,\sigma\rangle$, $\mathbf{Eval}(E,\rho,\sigma)=v \in Val \iff \langle E,\rho,\sigma\rangle \longrightarrow_e^* v$

elaborazione **D**: $\langle D,\rho,\sigma\rangle \longrightarrow_d \langle D',\rho',\sigma'\rangle$, $\mathbf{Elab}(D,\rho,\sigma)=\langle\rho',\sigma'\rangle \iff \langle D,\rho,\sigma\rangle \longrightarrow_d^* \langle\rho',\sigma'\rangle$

ambiente (dinamico) ρ : Id $\longrightarrow$ Loc $\cup$ Val   memoria σ : Loc $\longrightarrow$ Val

$\longrightarrow_c$, $\longrightarrow_e$, $\longrightarrow_d$ sono le funzioni di interpretazione semantica di C, E e D

stato i

Istruzioni da eseguire
$\rho = \{(Id_0, L_0), \ldots, (Id_i, v_i), \ldots\}$
$\sigma = \{(L_0, v_0), \ldots, (L_i, v_i), \ldots\}$

$\longrightarrow_c$ $\longrightarrow_e$ $\longrightarrow_d$

stato i+1

Istruzioni da eseguire
$\rho' = \{(Id_0, L_0), \ldots, (Id_i, v_i), \ldots\}$
$\sigma' = \{(L_0, v_0'), \ldots, (L_i, v_i'), \ldots\}$

stato finale

nil
$\rho = \{(Id_0, L_0), \ldots, (Id_i, v_i), \ldots\}$
$\sigma = \{(L_0, v_0), \ldots, (L_i, v_i), \ldots\}$

Sistema di transioni

Semantica Dinamica Espressioni

$$(Id1)\ \frac{\rho(Id)=v \ \vee\ (\rho(Id)=L \in Loc \wedge \sigma(L)=v)}{\langle Id,\rho,\sigma\rangle \longrightarrow_e v}$$

$$(uop1)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e \langle E',\rho,\sigma\rangle}{\langle uop\,E,\rho,\sigma\rangle \longrightarrow_e \langle uop\,E',\rho,\sigma\rangle} \qquad (uop2)\ \langle uop\ v,\rho,\sigma\rangle \longrightarrow_e v' = \text{uop}\ v$$

$$(bop1)\ \frac{\langle E_1,\rho,\sigma\rangle \longrightarrow_e \langle E_1',\rho,\sigma\rangle}{\langle E_1\,bop\,E_2,\rho,\sigma\rangle \longrightarrow_e \langle E_1'\,bop\,E_2,\rho,\sigma\rangle} \qquad (bop2)\ \frac{\langle E_2,\rho,\sigma\rangle \longrightarrow_e \langle E_2',\rho,\sigma\rangle}{\langle v_1\,bop\,E_2,\rho,\sigma\rangle \longrightarrow_e \langle v_1\,bop\,E_2',\rho,\sigma\rangle}$$

$$(bop3)\ \langle v_1 bop\ v_2,\rho,\sigma\rangle \longrightarrow_e v = v_1\ \text{bop}\ v_2$$

!!! *bop* è sintassi
bop è semantica

Semantica Dinamica Comandi

$$(id2)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* v}{\langle Id = E,\rho,\sigma\rangle \longrightarrow_c \langle Id = v,\rho,\sigma\rangle} \qquad (id3)\ \langle Id = v,\rho,\sigma\rangle \longrightarrow_c \sigma[\rho(Id) = v]$$

$$(seq1)\ \frac{\langle C_1,\rho,\sigma\rangle \longrightarrow_c \langle C_1',\rho,\sigma'\rangle}{\langle C_1;C_2,\rho,\sigma\rangle \longrightarrow_c \langle C_1';C_2,\rho,\sigma'\rangle} \qquad (seq2)\ \frac{\langle C_1,\rho,\sigma\rangle \longrightarrow_c \sigma'}{\langle C_1;C_2,\rho,\sigma\rangle \longrightarrow_c \langle C_2,\rho,\sigma'\rangle}$$

$$(if1)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* true}{\langle \mathtt{if}\,(E)\{C_1\}\,\mathtt{else}\,\{C_2\},\rho,\sigma\rangle \longrightarrow_c \langle C_1,\rho,\sigma\rangle} \qquad (if2)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* false}{\langle \mathtt{if}\,(E)\{C_1\}\,\mathtt{else}\,\{C_2\},\rho,\sigma\rangle \longrightarrow_c \langle C_2,\rho,\sigma\rangle}$$

$$(rep1)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* true}{\langle \mathtt{while}\,(E)\{C\}\,\rho,\sigma\rangle \longrightarrow_c \langle C;\mathtt{while}\,(E)\{C\},\rho,\sigma\rangle} \qquad (rep2)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* false}{\langle \mathtt{while}\,(E)\{C\},\rho,\sigma\rangle \longrightarrow_c \sigma}$$

$$(b1)\ \frac{\langle D,\rho,\sigma\rangle \longrightarrow_d^* \langle \rho',\sigma'\rangle}{\langle D;C,\rho,\sigma\rangle \longrightarrow_c \langle C,\rho[\rho'],\sigma[\sigma']\rangle}$$

Semantica Dinamica Dichiarazioni

$$(let1)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* v}{\langle let\ Id : T = E,\rho,\sigma\rangle \longrightarrow_d \langle [(Id,v)],\sigma\rangle}$$

$$(var1)\ \frac{\langle E,\rho,\sigma\rangle \longrightarrow_e^* v}{\langle var\ Id : T = E,\rho,\sigma\rangle \longrightarrow_d \langle [(Id,new\,L)],[(L,v)]\rangle}$$

$$(dd1)\ \frac{\langle D_1,\rho,\sigma\rangle \longrightarrow_d \langle D_1',\rho',\sigma'\rangle}{\langle D_1;D_2,\rho,\sigma\rangle \longrightarrow_d \langle D_1';D_2,\rho',\sigma'\rangle} \qquad (dd2)\ \frac{\langle D_2,\rho[\rho_1],\sigma\rangle \longrightarrow_d \langle D_2',\rho[\rho_1]',\sigma'\rangle}{\langle \rho_1;D_2,\rho[\rho_1],\sigma\rangle \longrightarrow_d \langle \rho_1;D_2',\rho[\rho_1]',\sigma'\rangle}$$

$$(dd3)\ \langle \rho_1;\rho 2,\rho,\sigma\rangle \longrightarrow_d \langle \rho_1[\rho_2],\sigma\rangle$$

le regole (dd2) e (dd3) contengono configurazioni non ammissibili rispetto alla definizione di sistema di transizione

$(dd2)\ \langle \rho_1; D_2, \rho, \sigma\rangle,\ \langle \rho_1; D_2', \rho, \sigma'\rangle \quad (dd3)\ \langle \rho_1; \rho 2, \rho, \sigma\rangle$

la parte codice delle configurazioni di stato deve essere generabile dalla grammatiche che definisce D, e questo non vale per le configurazioni sopra

aggiungo gli ambienti alla sintassi

D ::= nil | let **Id**[:**T**] = **E** | var **Id**[:**T**] = **E** | **D**;**D** | ρ
T ::= Int | Double | Bool | String

!!! solo il compilatore può generare gli ambienti della sintassi, non l'utente

Il sistema di transizione delle dichiarazioni è

$(\{\langle D, \rho, \sigma\rangle \cup \langle \rho', sigma'\rangle\}, \longrightarrow_d, \{\langle \rho', sigma'\rangle\},$ ⟨dichiarazione da elaborare, ambiente iniziale, memoria iniziale⟩)

Semantica Dinamica Funzioni

$$(FD1)\frac{\langle \text{func Id(form)} \to T\{C; \text{return } E\}, \rho, \sigma\rangle}{\langle (Id, \lambda\, \text{form} \,.\, \{\rho'; C; \text{return } E\}), \sigma\rangle} \qquad \begin{cases} \rho' = \rho_{|FV(C)-BV(form)} & \textit{scoping statico} \\ \rho' = nil & \textit{scoping dinamico} \end{cases}$$

$$(FD2)\frac{\rho(Id) = \lambda\, \text{form} \,.\, C}{\langle Id(ae), \rho, \sigma\rangle \to_e \langle \{form = ae; C\}, \rho, \sigma\rangle}$$

$$(FD3)\frac{\langle E, \rho, \sigma\rangle \to_e \langle E', \rho, \sigma\rangle}{\langle E, ae, \rho, \sigma\rangle \to_{ae} \langle E', ae, \rho, \sigma\rangle} \qquad (FD4)\frac{\langle ae, \rho, \sigma\rangle \to_{ae} \langle ae', \rho, \sigma\rangle}{\langle k, ae, \rho, \sigma\rangle \to_{ae} \langle k, ae', \rho, \sigma\rangle}$$

$$(FD5)\frac{\langle ae, \rho, \sigma\rangle \to_{ae} \langle ae', \rho, \sigma\rangle}{\langle \text{form} = ae, \rho, \sigma\rangle \to_d \langle \text{form} = ae', \rho, \sigma\rangle} \qquad (FD6)\frac{ak \vdash \text{form} : \rho_0}{\langle \text{form} = ak, \rho, \sigma\rangle \to_d \langle \rho_0, \sigma\rangle}$$

$$nil \vdash nil : \emptyset \qquad \frac{ak \vdash \text{form} : \rho}{k, ak \vdash \text{let Id} : \tau, \text{form} : \rho[(\text{Id}, k)]} \qquad \frac{ak \vdash \text{form} : \rho}{k, ak \vdash \text{var Id} : \tau, \text{form} : \rho[(\text{Id}, l_{(new)})]}$$

$$(RD1)\frac{\langle D, \rho - I_0, \sigma\rangle \to_d \langle D', \rho', \sigma'\rangle}{\langle rec\ D, \rho, \sigma\rangle \to_d \langle rec\ D', \rho', \sigma'\rangle}, I_0 = FI(D) \cap BI(D)$$

$$(RD2)\ \langle rec\ \rho_0, \rho, \sigma\rangle \to \langle \{(f, \lambda form \,.\, (rec\ \rho_0) - form; C) \mid \rho_0(f) = \lambda form \,.\, C\}, \sigma\rangle$$

Scoping e Identificatori Liberi

$FI_e : E \rightarrow$ {occorrenze Id liberi}

FI_e(v) = ∅

FI_e(Id) = {Id}

FI_e(uop E) = FI_e(E)

FI_e(E1 bop E2) = FI_e(E1) ∪ FI_e(E2)

$FI_c : C \rightarrow$ {occorrenze Id liberi}

FI_c(nil) = ∅

FI_c(Id = E) = {Id} ∪ FI_e(E)

FI_c(C1;C2) = FI_c(C1) ∪ FI_c(C2)

FI_c(if (E) {C1} else {C2}) =
FI_e(E) ∪ FI_c(C1) ∪ FI_c(C2)

FI_c(while (E) {C}) = FI_e(E) ∪ FI_c(C)

FI_c(D;C) = FI_d(D) ∪ (FI_c(C) - BI_d(D))

$FI_d : D \rightarrow$ {occorrenze Id liberi}

FI_d(nil) = ∅

FI_d(let Id:T = E) = FI_e(E)

FI_d(var Id:T = E) = FI_e(E)

FI_d(D1;D2) = FI_d(D1) ∪ (FI_d(D2)-BI_d(D1))

$BI_c = \overline{FI_c}$

$BI_e = \overline{FI_e}$

$BI_d = \overline{FI_d}$

FI_c(return E) = $FI_e(E)$

FI_e(Id(ae)) = {Id} ∪ $FI_{ae}(ae)$

FI_d(func Id(form))->T{C} =
FI_c(C) - BI_{form}(form)

FI_{form}(form) = ∅

FI_{ae}(E,ae) = $FI_e(E)$ ∪ $FI_{ae}(ae)$

Anatomia Funzioni