Come descrivere un Linguaggio di Programmazione

Sommario: 4-5 marzo, 2021

- Sintassi, Semantica, Pragmatica e Implementazione
- Grammatiche e Sintassi
- Grammatiche Context Free: Derivazione, Alberi, Ambiguità
- Sintassi: Proprietà contestuali
- Compilatore: Le fasi di analisi sintattica, semantica e di generazione del codice
- Esercizi Parte1: Front-End
- Semantica: Formalismi
- Semantica operazionale strutturata: Stato, Transizione, Computazione.
- Esercizi Parte2: Semantica

Sintassi

- Un programma si presenta come la "lunga" sequenza dei caratteri che formano i simboli e le strutture del linguaggio con cui il programma è espresso.

  `#include<stdio.h> #include<stdlib.h> voidXSwap(intA[], intB[]){A[0] = ...`

- **Lessico**. La definizione di **come** e **quali** sequenze di caratteri formino simboli (inclusi i separatori di simboli) è affidata al Lessico.

- **Sintassi**. La definizione di **come** e **quali** sequenze di simboli formino costrutti e la struttura del programma è affidata alla Sintassi.

Grammatiche

- **Grammatiche**. Lessico e Sintassi possono essere completamente definite mediante grammatiche:
  - Lineari, per il lessico, $\mathcal{G}^{\mathcal{R}}$.
  - Libere da contesto (Context Free), per la sintassi, $\mathcal{G}^{\mathcal{F}}$.
  - $\mathcal{G}^{\mathcal{R}} \subset \mathcal{G}^{\mathcal{F}}$.

- Sia A insieme finito di simboli (caratteri):
  - $\mathbf{A}^\star$ insieme di tutte le sequenze finite di simboli su A.
  - $\mathtt{A}^\star$ è chiuso rispetto all'operatore binario, associativo, $\_.\_$, chiamato *giustapposizione*[1]: esempio Pi.sa = Pisa.
  - $\epsilon \in \mathtt{A}^\star$ seq. vuota: $\epsilon.\mathtt{q} = \mathtt{q} = \mathtt{q}.\epsilon, \forall \mathtt{q} \in \mathtt{A}^\star$

Definition (Linguaggio (formale) su A)

Un linguaggio (formale) su A è un sottoinsieme di $\mathtt{A}^\star$

[1] la sua notazione è omessa quando non vi è ambiguità

Grammatiche Libere, $\mathcal{G}^{\mathcal{F}}$ /1

Definition (Grammatica Libera)

Una grammatica libera è una quadrupla (NT,T,R,S) tale che:

- **NT** insieme finito dei non terminali.
- **T** insieme finito dei terminali.
- **S** simbolo iniziale, S∈NT.
- **R** insieme finito delle produzioni,

$$\mathtt{R} \equiv \{\mathtt{V_i} \rightarrow \mathtt{w_i} \mid \mathtt{V_i} \in \mathtt{NT}, \mathtt{w_i} \in (\mathtt{T} \cup \mathtt{NT})^*, \mathtt{i} \in [1, \mathtt{k}]\}$$

Esempio

$\mathtt{G} = (\{\mathtt{E}\}, \{\mathtt{I}, +, *, (,)\}, \mathtt{E}, \mathtt{R})$

$\mathtt{R} = \{\mathtt{E} \rightarrow \mathtt{E} + \mathtt{E}, \mathtt{E} \rightarrow \mathtt{E} * \mathtt{E}, \mathtt{E} \rightarrow \mathtt{I}, \mathtt{E} \rightarrow (\mathtt{E})\}$

Notazione. $\mathtt{V} \rightarrow \alpha_1 | ... | \alpha_\mathtt{k}$ invece di $\mathtt{V_1} \rightarrow \alpha_1, ..., \mathtt{V_k} \rightarrow \alpha_\mathtt{k}$ quando, $\mathtt{V} = \mathtt{V_1} = ... = \mathtt{V_k}$.

Esempio

$\mathtt{R} = \{\mathtt{E} \rightarrow \mathtt{E} + \mathtt{E} \mid \mathtt{E} * \mathtt{E} \mid \mathtt{I} \mid (\mathtt{E})\}$

Grammatiche Libere, $\mathcal{G}^{\mathcal{F}}$ /2

Esempio

$\mathtt{G} = (\{\mathtt{E}\}, \{\mathtt{I}, +, *, (,)\}, \mathtt{E}, \mathtt{R})$

$\mathtt{R} = \{\mathtt{E} \to \mathtt{E} + \mathtt{E}, \mathtt{E} \to \mathtt{E} * \mathtt{E}, \mathtt{E} \to \mathtt{I}, \mathtt{E} \to (\mathtt{E})\}$

Qual'è il significato di G? [2]

Definition (La relazione $\Rightarrow$)

Ogni grammatica libera $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ definisce una relazione binaria $\Rightarrow_{\mathtt{G}}$ (o, semplicemente $\Rightarrow$) su $(\mathtt{T} \cup \mathtt{NT})^*$ tale che:

$$\forall\, \alpha.\mathtt{V}.\rho,\ \alpha.\gamma.\rho \in (\mathtt{T} \cup \mathtt{NT})^*, \quad \alpha\mathtt{V}\rho \Rightarrow \alpha\gamma\rho \text{ sse } \mathtt{V} \to \gamma \in \mathtt{R}$$

Definition (Linguaggio generato)

Sia $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ libera. Il linguaggio definito da $\mathtt{G}$:

$$\mathcal{L}(\mathtt{G}) = \{\alpha \in \mathtt{T}^\star | \mathtt{S} \Rightarrow^\star \alpha\}$$

dove, $\Rightarrow^\star$ è la chiusura transitiva (e riflessiva) di $\Rightarrow_{\mathtt{G}}$

[2] "sse" sta per "se e solo se"

Grammatiche Libere, $\mathcal{G}^{\mathcal{F}}$ /3

Definition (Derivazione)

Sia $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ libera. Siano $\mathtt{v}, \mathtt{w} \in (\mathtt{T} \cup \mathtt{NT})^*$. Sia $\mathtt{v} \Rightarrow^\star \mathtt{w}$. Una derivazione, in $\mathtt{G}$, da $\mathtt{v}$ a $\mathtt{w}$, è una sequenza $\mathtt{v} \Rightarrow \mathtt{w}_0 \Rightarrow \mathtt{w}_1 \Rightarrow \ldots \Rightarrow \mathtt{w}$

Esempio
La derivazione ci fornisce una prova di appartenenza. Si dimostri che $\mathtt{I} * \mathtt{I} + \mathtt{I} \in \mathcal{L}(\mathtt{G})$.

$\mathtt{G} = (\{\mathtt{E}\}, \{\mathtt{I}, +, *\}, \mathtt{E}, \mathtt{R})$
$\mathtt{R} = \{\mathtt{E} \to \mathtt{E} + \mathtt{E}, \mathtt{E} \to \mathtt{E} * \mathtt{E}, \mathtt{E} \to \mathtt{I}, \mathtt{E} \to (\mathtt{E})\}$

Procediamo applicando una $\Rightarrow$ alla volta a partire da $\mathtt{E}$

$\mathtt{E} \Rightarrow_2 \mathtt{E} * \mathtt{E}$ [3]
$\Rightarrow_3 \mathtt{I} * \mathtt{E}$
$\Rightarrow_1 \mathtt{I} * \mathtt{E} + \mathtt{E}$
$\Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{E}$
$\Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{I}$

Aggiungendo induzione (su ...) possiamo dimostrare che:

$$\{\mathtt{I}.\alpha_1.\cdots.\alpha_\mathtt{n} \mid \alpha_\mathtt{i} \in \{+\mathtt{I}, *\mathtt{I}\}, \mathtt{n} \geq 0\} \subset \mathcal{L}(G).$$

[3] Un pedice può indicare la produzione usata. In questo caso, usiamo la posizione a partire da 1 per la produzione più a sinistra

Grammatiche Libere: La relazione $\Rightarrow$ su alberi /1

Esempio

$$G = (\{E\}, \{I, +, *, (,)\}, E, R)$$
$$R = \{E \to E + E, E \to E * E, E \to I, E \to (E)\}$$

- La $\Rightarrow$ su sequenze di simboli (stringhe) grammaticali ci dice proprio tutto?

$$E \Rightarrow_2 E * E \Rightarrow_3 I * E \Rightarrow_1 I * E + E \Rightarrow_3 I * I + E \Rightarrow_3 I * I + I$$
$$E \Rightarrow_2 E * E \Rightarrow_1 E * E + E \Rightarrow_3 I * E + E \Rightarrow_3 I * I + E \Rightarrow_3 I * I + I$$
$$E \Rightarrow_1 E + E \Rightarrow_2 E * E + E \Rightarrow_3 I * E + E \Rightarrow_3 I * I + E \Rightarrow_3 I * I + I$$

- In realtà NO.
  - 2 delle 3 derivazioni sopra ci dicono la stessa cosa, 1 dice una cosa differente.
  - Quali? In cosa, sono diverse?
  - Useremo la struttura degli alberi per rispondere.

- La sequenza di simboli grammaticali non è abbastanza espressiva

Grammatiche Libere: La relazione $\Rightarrow$ su alberi /2

Definition

- $[] \in \texttt{Tree}^{\texttt{A}}$
- $[\texttt{a} - (\texttt{t}_1, ..., \texttt{t}_\texttt{n})] \in \texttt{Tree}^{\texttt{A}}$, per $\texttt{a} \in \texttt{A}, \texttt{t}_\texttt{i} \in \texttt{Tree}^{\texttt{A}}, \texttt{n} \geq 0$

Notazione. [a - ()] può essere scritto [a].

- Gli alberi possono essere utilizzati come relazioni sulla struttura di un termine
  - Ogni arco è una coppia della relazione *sottotermine*
- Hanno una rappresentazione grafica.
  - mostriamo quella di [E - ([E],[+],[E])]

Grammatiche Libere: La relazione $\Rightarrow$ su alberi /3

Definition (La relazione $\Rightarrow^{\mathtt{Tree}}$)

Ogni grammatica libera $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ definisce una relazione binaria $\Rightarrow^{\mathtt{Tree}}_{\mathtt{G}}$ (o, semplicemente $\Rightarrow^{\mathtt{Tree}}$, ovvero $\Rightarrow$) su $\mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}}$ tale che:

- $[\mathtt{V}] \Rightarrow [\mathtt{V} - ([\mathtt{u}_1], ..., [\mathtt{u}_n])]$
  sse $\mathtt{V} \rightarrow \mathtt{u}_1 . \ldots . \mathtt{u}_n \in \mathtt{R}$ and $\mathtt{u}_i \in (\mathtt{T} \cup \mathtt{NT})$
- $[\mathtt{V} - (\mathtt{t}_1, ..., \mathtt{t}_j, ..., \mathtt{t}_n)] \Rightarrow [\mathtt{V} - (\mathtt{t}_1, ..., \mathtt{r}_j, ..., \mathtt{t}_n)]$
  sse $1 \leq j \leq n \wedge (\forall i) \mathtt{t}_i \in \mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}} \wedge \mathtt{t}_j \Rightarrow \mathtt{r}_j$

Definition (Linguaggio generato su $\mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}}$ - Parse Tree set)

Sia $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ libera. L'insieme dei parse tree di $\mathtt{G}$ è:

$$\mathcal{PT}(\mathtt{G}) = \{\alpha \in \mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}} \mid [\mathtt{S}] \Rightarrow^{\star} \alpha\}$$

dove, $\Rightarrow^{\star}$ è la chiusura transitiva (e riflessiva) di $\Rightarrow$

Grammatiche Libere: La relazione $\Rightarrow$ su alberi /4

Definition (Derivazione su $\mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}}$)

Sia $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ libera. Siano $\mathtt{u}, \mathtt{q} \in \mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}}$. Sia $\mathtt{u} \Rightarrow^{\star} \mathtt{q}$. Una derivazione, in $\mathtt{G}$, da $\mathtt{u}$ a $\mathtt{q}$, è una sequenza

$\mathtt{u} \Rightarrow \mathtt{q}_0 \Rightarrow \mathtt{q}_1 \Rightarrow \ldots \Rightarrow \mathtt{q}$

Esempio

$\mathtt{G} = (\{\mathtt{E}\}, \{\mathtt{I}, +, *, (,)\}, \mathtt{E}, \mathtt{R})$

$\mathtt{R} = \{\mathtt{E} \to \mathtt{E} + \mathtt{E}, \mathtt{E} \to \mathtt{E} * \mathtt{E}, \mathtt{E} \to \mathtt{I}, \mathtt{E} \to (\mathtt{E})\}$

- La $\Rightarrow$ su stringhe ci dice proprio tutto?

$\mathtt{E} \Rightarrow_2 \mathtt{E} * \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{E} \Rightarrow_1 \mathtt{I} * \mathtt{E} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{I}$

$\mathtt{E} \Rightarrow_2 \mathtt{E} * \mathtt{E} \Rightarrow_1 \mathtt{E} * \mathtt{E} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{E} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{I}$

$\mathtt{E} \Rightarrow_1 \mathtt{E} + \mathtt{E} \Rightarrow_2 \mathtt{E} * \mathtt{E} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{E} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{E} \Rightarrow_3 \mathtt{I} * \mathtt{I} + \mathtt{I}$

- La $\Rightarrow$ su alberi?

$[\mathtt{E}] \Rightarrow_2 [\mathtt{E} - ([\mathtt{E}], [*], [\mathtt{E}])] \Rightarrow_3 [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E}])]$

$\Rightarrow^{\star} [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [+], [\mathtt{E} - ([\mathtt{I}])])])]$

$[\mathtt{E}] \Rightarrow^{\star} \ldots$

$[\mathtt{E}] \Rightarrow_1 [\mathtt{E} - ([\mathtt{E}], [+], [\mathtt{E}])] \Rightarrow_2 [\mathtt{E} - ([\mathtt{E} - ([\mathtt{E}], [*], [\mathtt{E}])], [+], [\mathtt{E}])]$

$\Rightarrow^{\star} [\mathtt{E} - ([\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E} - ([\mathtt{I}])])], [+], [\mathtt{E} - ([\mathtt{I}])])]$

La relazione $\Rightarrow$ su alberi: Una vista grafica

Esempio

- La $\Rightarrow$ su alberi?

$$[\mathtt{E}] \Rightarrow_2 [\mathtt{E} - ([\mathtt{E}], [*], [\mathtt{E}])] \Rightarrow_3 [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E}])]$$
$$\Rightarrow^\star [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [+], [\mathtt{E} - ([\mathtt{I}])])])]$$

$$[\mathtt{E}] \Rightarrow^\star \ldots$$
$$[\mathtt{E}] \Rightarrow_1 [\mathtt{E} - ([\mathtt{E}], [+], [\mathtt{E}])] \Rightarrow_2 [\mathtt{E} - ([\mathtt{E} - ([\mathtt{E}], [*], [\mathtt{E}])], [+], [\mathtt{E}])]$$
$$\Rightarrow^\star [\mathtt{E} - ([\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E} - ([\mathtt{I}])])], [+], [\mathtt{E} - ([\mathtt{I}])])]$$

Grammatiche Libere: Frontiera di un albero /5

Definition (Frontiera di un albero)

La frontiera di un albero $\mathtt{q} \in \mathtt{Tree}^{\mathtt{A}}$ è una stringa di $\mathtt{A}^\star$, ovvero $\mathtt{fron(q)} \in \mathtt{A}^\star$ per ogni q, così definita:

- $\mathtt{fron}([]) = \epsilon$
- $\mathtt{fron}([\mathtt{a}]) = \mathtt{a}$
- $\mathtt{fron}([\mathtt{a} - (\mathtt{t}_1, ..., \mathtt{t}_n)]) = \mathtt{fron}(\mathtt{t}_1).\cdots.\mathtt{fron}(\mathtt{t}_n), \quad \mathtt{n} > 0$

Esempio

$$\begin{aligned}
&\mathtt{fron}([\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [*], [\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [+], [\mathtt{E} - ([\mathtt{I}])])])]) = \\
&\quad = \mathtt{fron}([\mathtt{E} - ([\mathtt{I}])]).\mathtt{fron}([*]).\mathtt{fron}([\mathtt{E} - ([\mathtt{E} - ([\mathtt{I}])], [+], [\mathtt{E} - ([\mathtt{I}])])]) \\
&\quad = \mathtt{fron}([\mathtt{I}]).\mathtt{fron}([*]).\mathtt{fron}([\mathtt{E} - ([\mathtt{I}])]).\mathtt{fron}([+]).\mathtt{fron}([\mathtt{E} - ([\mathtt{I}])]) \\
&\quad = \mathtt{fron}([\mathtt{I}]).\mathtt{fron}([*]).\mathtt{fron}([\mathtt{I}]).\mathtt{fron}([+]).\mathtt{fron}([\mathtt{I}]) \\
&\quad = \mathtt{I}.*.\mathtt{I}.+.\mathtt{I} = \mathtt{I} * \mathtt{I} + \mathtt{I}
\end{aligned}$$

Grammatiche Libere: Ambiguità /6

Definition (Linguaggio generato utilizzando Parse Tree)

Sia $\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ libera. Il linguaggio definito da $\mathtt{G}$ è:

$$\mathcal{L}(\mathtt{G}) = \{\mathtt{fron(q)} \in \mathtt{T}^\star \mid [\mathtt{S}] \Rightarrow^\star \mathtt{q}\}$$

dove, $\Rightarrow^\star$ è la chiusura transitiva (e riflessiva) di $\Rightarrow_{\mathtt{G}}^{\mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}}}$

Definition (Grammatica Ambigua)

$\mathtt{G} \equiv (\mathtt{NT}, \mathtt{T}, \mathtt{R}, \mathtt{S})$ è ambigua se $\exists \mathtt{q}_1 \neq \mathtt{q}_2 \in \mathtt{Tree}^{\mathtt{T} \cup \mathtt{NT}}$:

- $[\mathtt{S}] \Rightarrow^\star \mathtt{q}_1, [\mathtt{S}] \Rightarrow^\star \mathtt{q}_2$
- $\mathtt{fron}([\mathtt{q}_1]) = \mathtt{fron}([\mathtt{q}_2]) \in \mathtt{T}^*$

Grammatiche Libere: Ambiguità /7

- La **sintassi di un LP** deve essere espressa mediante una grammatica libera **non ambigua**
- Ogni **termine** di LP deve avere **un'unica struttura sintattica** che ne mostra i sottotermini componenti

Esempio Grammatica Ambigua

$$G = (\{E\}, \{I, +, *\}, E, R)$$
$$R = \{E \to E + E, E \to E * E, E \to I, E \to (E)\}$$

Esempio Grammatica non Ambigua

$$G' = (\{E, F, T\}, \{I, +, *\}, E, R')$$
$$R' = \{E \to E + F \mid F$$
$$F \to F * T \mid T$$
$$T \to I \mid (E)\}$$

$$\mathcal{L}(G) = \mathcal{L}(G')$$

Grammatiche: Lessico e Sintassi

- Lessico e Sintassi si integrano nella definizione di un Linguaggio

Esempio sottoLinguaggio delle Espressioni Aritmetiche su Interi

Sintassi: Espressione

$$\mathtt{G} = (\{\mathtt{E}, \mathtt{F}, \mathtt{T}\}, \{\mathtt{N}, +, -, *\}, \mathtt{E}, \mathtt{R_G})$$

$$\begin{aligned}\mathtt{R_G} = \{\ & \mathtt{E} \to \mathtt{E} + \mathtt{F} \mid \mathtt{F} \\ & \mathtt{F} \to \mathtt{F} * \mathtt{T} \mid \mathtt{T} \\ & \mathtt{T} \to -\mathtt{T} \mid \mathtt{N} \mid (\mathtt{E})\}\end{aligned}$$

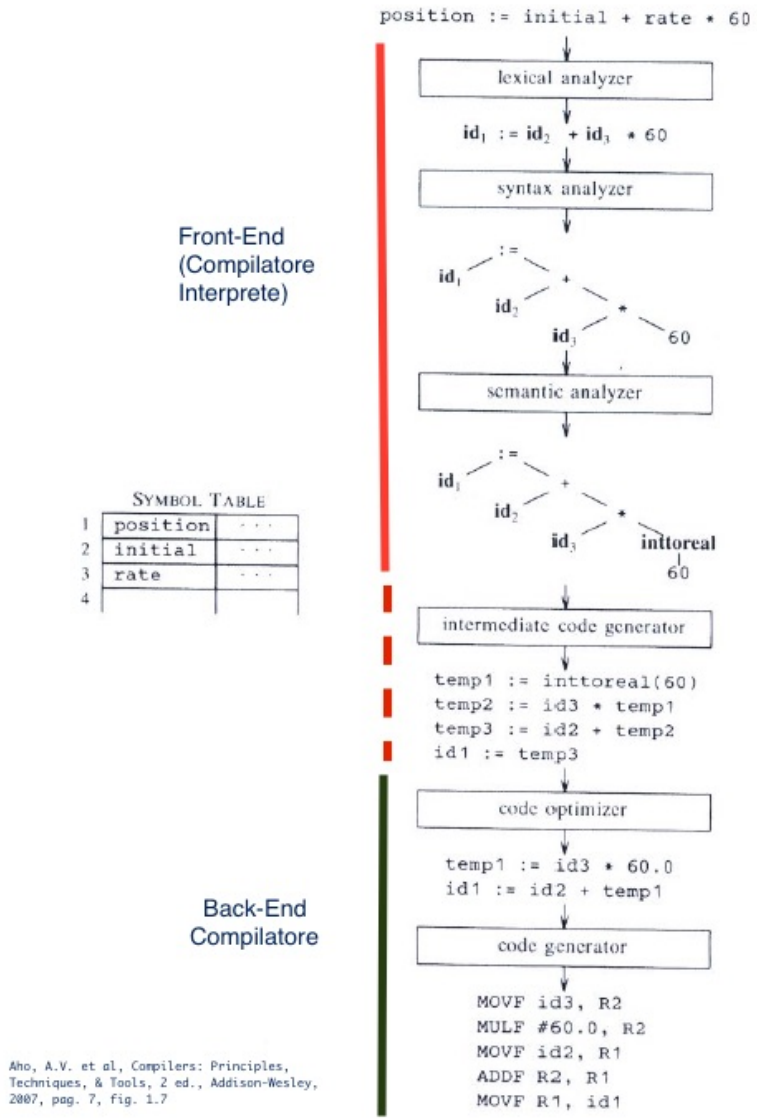
Lessico: Naturale (Identificatore)

$$\mathtt{H} = (\{\mathtt{L}, \mathtt{N}, \mathtt{D}\}, \{-, +, *, 0, \cdots, 9\}, \mathtt{L}, \mathtt{R_H})$$

$$\begin{aligned}\mathtt{R_H} = \{\ & \mathtt{L} \to \mathtt{N} \mid * \mid - \mid + \\ & \mathtt{N} \to \mathtt{D}\ \mathtt{N} \mid \mathtt{D} \\ & \mathtt{D} \to 0 \mid 1 \mid \cdots \mid 9\}\end{aligned}$$

Riconosciamo la stringa, 13*-5+007$\in \mathcal{L}(<\mathtt{G},\mathtt{H}>)$.

Compilatore/Interprete: Front-End e Back-End



Aho, A.V. et al, Compilers: Principles, Techniques, & Tools, 2 ed., Addison-Wesley, 2007, pag. 7, fig. 1.7

Vincoli Sintattici Contestuali: Problema

- **Programmi (sintatticamente) Legali** sono i programmi sintatticamente corretti a cui la semantica del linguaggio può associare la funzione calcolabile descritta dal programma.

- La sintassi espressa da una grammatica non ambigua non è sufficiente a identificare tutti e solo i programmi legali.

- Ad esempio: `x = 7` è un assegnamento
  - sintatticamente corretto in tutti i programmi C [4]
  - ma non è legale se `x` non è stato *dichiarato* [5]

[4] e di tutti i L.P. dove `x`, `_=_`, `7` siano la sintassi per una variabile, per l'operatore di assegnamento, per un'espressione, rispettivamente.

[5] opportunamente, nel programma

Vincoli Sintattici Contestuali: Soluzione

- **Proprietà contestuali** (delle regole di composizione) non possono essere espresse da una grammatica libera (da contesto, i.e. Context-Free):
  - identificatori usati devono essere dichiarati prima ...;
  - numero di parametri attuali e formali devono coincidere
  - compatibilità nell'assegnamento, tra il tipo di una variabile e il tipo dell'espressione assegnata
  - ...
- Queste proprietà contestuali **fanno parte della definizione del linguaggio**, ma devono essere espresse con strumenti adatti.
- **Analisi Statica** Sono procedimenti di analisi e formalismi specifici (ad es., Sistema dei Tipi) per controllare che il programma soddisfi tutte le proprietà contestuali del linguaggio.

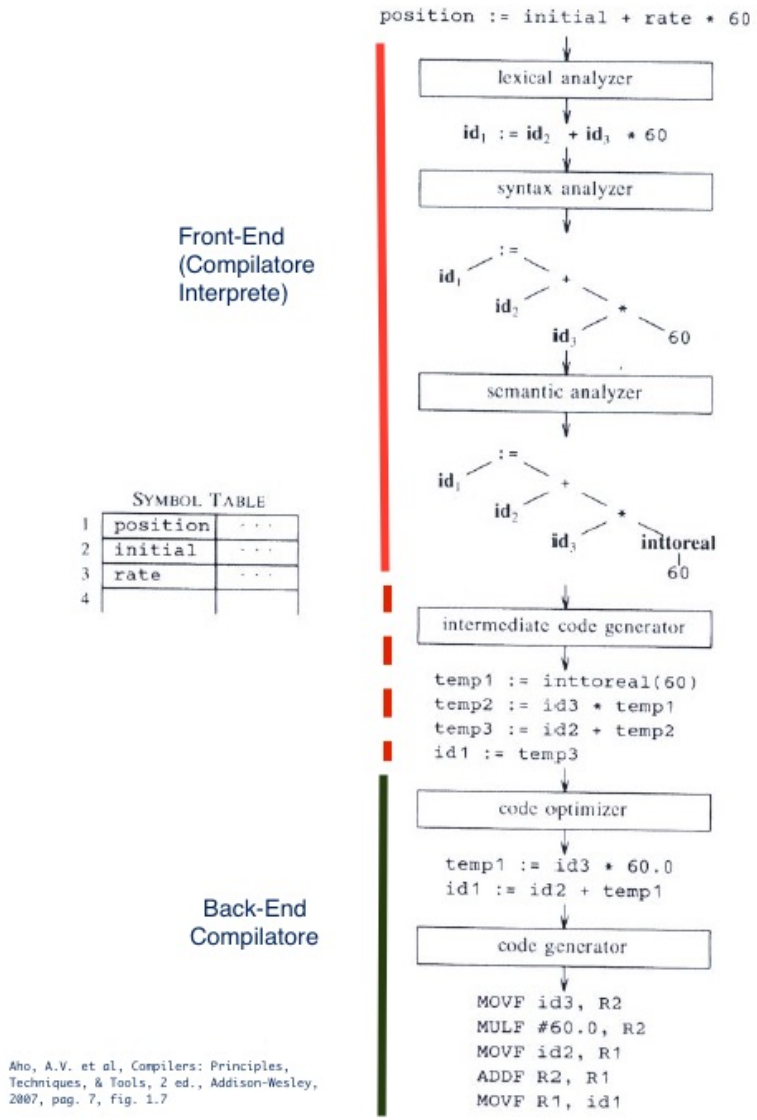
Compilatore, Interprete: Front-End

- **Programmi (sintatticamente) Legali.** Compilatori e Interpreti trattano tutti questi aspetti nel front-end.
- Il **Front-End** di C./I.[6] provvede alla:
  - **Analisi Lessicale** in accordo al lessico
  - **Analisi Sintattica** in accordo alla sintassi
  - **Analisi Statica** in accodo alle proprietà contestuali
  - **Costruzione Abstract Tree** `AT(p)` che fornisce una rappresentazione interna del programma sorgente `p`.

- `AT(p)` è successivamente utilizzato dal **Back-End** di C./I. per completare la realizzazione dell'esecutore.
- Vediamo tutto questo nella tipica struttura (in fasi) di un Compilatore.

[6] Compilatori e Interpreti

Compilatore/Interprete: Front-End e Back-End



Aho, A.V. et al, Compilers: Principles, Techniques, & Tools, 2 ed., Addison-Wesley, 2007, pag. 7, fig. 1.7

Sintassi Concreta e Sintassi Astratta /1

Definition (Sintassi Concreta)

Insieme delle sequenze di simboli, dato un lessico, che formano i costrutti ed i programmi esprimibili in un Linguaggio di Programmazione.

- Interfaccia per il programmatore
- Molto più facile scrivere un programma in S. Concreta che in S. Astratta.
- Ma occorre conoscere Sintassi del linguaggio utilizzato.

```c
#include <stdio.h>
#include <stdlib.h>

int main (void){
    int x = 10, y = 5, z = 7;
    if (x>y) if (y-z>0) z=x;
    else z = z+y;
    printf("x = %d; y = %d; z = %d;\n", x, y, z);
    return(1);
}
```

- Conoscete la Sintassi del Linguaggio C?
- Allora spiegate la struttura dei condizionali usati nel programma, ovvero quali valori sono stampati dalla sua esecuzione.

Sintassi Concreta e Sintassi Astratta /1

Definition (Sintassi Astratta)

Insieme degli alberi etichettati che rappresentano i costrutti ed i programmi esprimibili in un Linguaggio di Programmazione. La rappresentazione deve mostrare in modo univoco, la relazione tra il termine e i sotto-termini che lo compongono.

- Interfaccia per il Back-End
- Faticoso scrivere un programma in S. Astratta: Ma MAI AMBIGUA
- Contiene tutte e solo le proprietà del programma (niente fronzoli mnemonici sui simboli, o parentesi per associare/grouping termini, o convenzioni o sottintesi)

- Durante lo studio di un LP la definizione della S. Concreta è data come l'ultima attività
- Per gran parte dello sviluppo di un LP, i programmi del nuovo linguaggio sono scritti direttamente in S. Astratta.
- Preparatevi a farlo in Laboratorio.

Sintassi Astratta: Equazioni su Alberi

Definition ("Grammatica" per Sintassi Astratta)

La S. Astratta è espressa con formalismi che utilizzano notazioni che ricordano una grammatica. Usano definizioni che ricordano le produzioni ed hanno forma:

$$\mathtt{A} ::= [\mathtt{csr}]\ \mathtt{A}_1\ \ldots\ \mathtt{A}_n$$

dove:

- $\mathtt{A}$, $\mathtt{A}_1$, ... $\mathtt{A}_n$ sono tipi di alberi: Ogni tipo associato univocamente ad una categoria della sintassi concreta
- [csr] è un costruttore di alberi con segnatura: $\mathtt{A}_1\ \ldots\ \mathtt{A}_n \rightarrow\ \mathtt{A}$
- I costruttori costruiscono alberi su $\mathtt{Tree}^{\mathtt{T}}$, con $\mathtt{T}$ sottoinsieme dei Terminali della Sintassi Concreta: L'etichetta di ogni nodo dipende dal costruttore.

Esempio sottoLinguaggio delle Espressioni Aritmetiche (omettiamo il lessico per $\mathtt{N}$)

Sintassi Astratta: Una grammatica su $\mathtt{Tree}^{\mathtt{T}}$ per le Espressioni

$$\mathtt{G}^{\mathtt{A}} = (\{\mathtt{E}\}, \{\mathtt{N}, +, -, *\}, \mathtt{E}, \mathtt{R}_{\mathtt{G}^{\mathtt{A}}})$$

$$\mathtt{R}_{\mathtt{G}^{\mathtt{A}}} = \{\mathtt{E} ::= [+]\ \mathtt{E}\ \mathtt{E},\ \mathtt{E} ::= [*]\ \mathtt{E}\ \mathtt{E},\ \mathtt{E} = [-]\mathtt{E},\ \mathtt{E} = [\mathtt{N}]\}$$

Somma di fattori: $[+]([\mathtt{N}], [*]([\mathtt{N}], [\mathtt{N}]))$
Prodotti di addendi: $[*]([+]([\mathtt{N}], [\mathtt{N}]), [\mathtt{N}])$
Sintassi concreta: $\mathtt{N}\ +\ \mathtt{N}\ *\ \mathtt{N}$

Esercizi L4

1. Una grammatica $\mathtt{G}$ calcola un linguaggio. Si dica:
   (a) Quali sono gli oggetti di tale linguaggio;
   (b) Di quale insieme è sotto-insieme tale linguaggio
   (c) Quale relazione (fornire definizione) permette di definire tale linguaggio?
   (e) Si fornisca la definizione di tale linguaggio
2. Sia $\mathtt{G} \equiv \{\{\mathtt{T}\}, \{\mathtt{A}, \mathtt{E}\}, \mathtt{T}, \{\mathtt{T} \rightarrow \mathtt{A}, \mathtt{T} \rightarrow \mathtt{A}, \mathtt{T} \rightarrow \mathtt{A}\,\mathtt{E}\,\mathtt{T}\}\}$. Si applichino a $\mathtt{G}$ le formulazione date nei punti (a)-(e) di esercizio L4.1.
3. (a) Si formalizzi la relazione tra $\Rightarrow_{\mathtt{G}}$ e $\Rightarrow_{\mathtt{G}}^{\mathtt{Tree}}$
   (b) Quando usare l'una e quando l'altra?
4. (a) Si completi la definizione di una grammatica le cui produzioni sono sotto.
   (b) Si mostri che la grammatica ottenuta è ambigua.
   $$\mathtt{E} \rightarrow \mathtt{T} \mid \mathtt{T} + \mathtt{E} \mid \mathtt{T} - \mathtt{E}$$
   $$\mathtt{T} \rightarrow \mathtt{A} \mid \mathtt{A} * \mathtt{E}$$
5. Si dia una grammatica non ambigua che generi tutte e sole le sequenze di parentesi angolate bilanciate
6. Si chiama regolare una grammatica ... (v. Esercizi4Lezione4Parte1).
   Si mostri che il linguaggio $\mathtt{L} \equiv \{\mathtt{a}^{\mathtt{n}}\mathtt{b}^{\mathtt{m}} \mid \mathtt{n}, \mathtt{m} \geq 1\}$ è regolare.
7. Si dia una grammatica non ambigua per il linguaggio $\mathtt{L} \equiv \{\mathtt{a}^{\mathtt{n}}\mathtt{b}^{\mathtt{m}} \mid \mathtt{n} \leq \mathtt{m}\}$.
8. (a) A cosa serve la sintassi concreta di un L. di Programmazione?
   (b) Cosa la distingue dalla sintassi astratta.

Altri esercizi in:
* Esercizi4Lezione4Parte1

* Cap. 2 [GM]

Semantica: Denotazionale e Operazionale

- Associare significato ai termini del linguaggio.
- Vari formalismi con differenti accezioni di significato e differenti usi.
- Due da ricordare:
  - **Semantica Denotazionale**
    - *Significato*: Funzione Calcolata
    - *Usi*: Molteplici incluso definizione di C. e di I.
    - *Laboratorio*: Interpreti di $\mathcal{L}$, derivabili dalla S. Den. di $\mathcal{L}$, riscritta in un Linguaggio di P. Funzionale, OCaml
    - *Caratteristica*: Orientata allo studio di proprietà di $\mathcal{L}$, Astratta dall'implementazione di $\mathcal{L}$.
  - **Semantica Operazionale**

Semantica: Operazionale

- Associare significato ai termini del linguaggio.
- Vari formalismi con differenti accezioni di significato e differenti usi.
- Due da ricordare:
  - **Semantica Denotazionale**
  - **Semantica Operazionale**
    - *Significato*: Computazione (su Macchina)
    - *Usi*: Definizione di I., anche didattici e per studio di programmi e programmazione in $\mathcal{L}$
    - *Caratteristica*: Basata sulla nozione di Stato, transizione di stato, computazione
- Useremo una Semantica Operazionale *Strutturata* in cui la Macchina è una Macchina Astratta.

Semantica Operazionale Strutturata, SOS

- La definizione di Stato, Transizione di stato e Computazione di una S.O.S, dipende dal Linguaggio.
- Vediamole per il Linguaggio sotto.
- Assumiamo che sintassi (concreta) e lessico siano già stati trattati, analizzati, fornendoci la sintassi astratta dei termini del linguaggio: "(AExp-AExp)" va letto [7] come un albero con radice "-" e due alberi "AExp" come figli (i simboli "(",",")" rimarcano questa lettura).

$$
\begin{array}{lll}
\mathit{Num} & ::= & 1 \mid 2 \mid 3 \mid \ldots \\
\mathit{Var} & ::= & X_1 \mid X_2 \mid X_3 \mid \ldots
\end{array}
$$

$$
\begin{array}{lll}
\mathit{AExp} & ::= & \mathit{Num} \mid \mathit{Var} \mid (\mathit{AExp} + \mathit{AExp}) \mid (\mathit{AExp} - \mathit{AExp}) \\
\mathit{BExp} & ::= & \mathbf{tt} \mid \mathbf{ff} \mid (\mathit{AExp} == \mathit{AExp}) \mid \neg \mathit{BExp} \mid (\mathit{BExp} \wedge \mathit{BExp})
\end{array}
$$

$$
\begin{array}{lll}
\mathit{Com} & ::= & \mathbf{skip} \mid \mathit{Var} := \mathit{AExp} \mid \mathit{Com}; \mathit{Com} \mid \\
 & & \quad \mathbf{if}\ \mathit{BExp}\ \mathbf{then}\ \mathit{Com}\ \mathbf{else}\ \mathit{Com} \mid\ \mathbf{while}\ \mathit{BExp}\ \mathbf{do}\ \mathit{Com}
\end{array}
$$

[7] **Esercizio (aggiuntivo).**
Fornire la definizione formale di $\mathcal{L}(\mathtt{G})$, quando $\mathtt{G}$ sia una grammatica per sintassi astratta (ovvero la definizione che manca in slide 23 di questi lucidi.

SOS: Lo stato

$$
\begin{array}{lll}
\mathit{Num} & ::= & 1 \mid 2 \mid 3 \mid \ldots \\
\mathit{Var} & ::= & X_1 \mid X_2 \mid X_3 \mid \ldots
\end{array}
$$

$$
\begin{array}{lll}
\mathit{AExp} & ::= & \mathit{Num} \mid \mathit{Var} \mid (\mathit{AExp} + \mathit{AExp}) \mid (\mathit{AExp} - \mathit{AExp}) \\
\mathit{BExp} & ::= & \mathbf{tt} \mid \mathbf{ff} \mid (\mathit{AExp} == \mathit{AExp}) \mid \neg \mathit{BExp} \mid (\mathit{BExp} \wedge \mathit{BExp})
\end{array}
$$

$$
\begin{array}{lll}
\mathit{Com} & ::= & \mathbf{skip} \mid \mathit{Var} := \mathit{AExp} \mid \mathit{Com}; \mathit{Com} \mid \\
& & \mathbf{if}\ \mathit{BExp}\ \mathbf{then}\ \mathit{Com}\ \mathbf{else}\ \mathit{Com} \mid \ \mathbf{while}\ \mathit{BExp}\ \mathbf{do}\ \mathit{Com}
\end{array}
$$

- La definizione di Stato.
  - Identificatori di un unico tipo: Variabili (valori modificabili)
  - Memoria simbolica per trattare le variabili (valori modificabili)
  - Non abbiamo I/O, file systems,.... Lo stato è la sola memoria
  - Lo stato è rappresentato da sequenza finita di coppie $(\mathtt{X_i}, \mathtt{n_i})$, indicante ...
  - Lo stato è denotato con le (meta)variabili σ, τ (anche con pedici)

SOS: Transizione

$$
\begin{array}{lll}
Num & ::= & 1 \mid 2 \mid 3 \mid \ldots \\
Var & ::= & X_1 \mid X_2 \mid X_3 \mid \ldots
\end{array}
$$

$$
\begin{array}{lll}
AExp & ::= & Num \mid Var \mid (AExp + AExp) \mid (AExp - AExp) \\
BExp & ::= & \mathbf{tt} \mid \mathbf{ff} \mid (AExp == AExp) \mid \neg BExp \mid (BExp \wedge BExp)
\end{array}
$$

$$
\begin{array}{lll}
Com & ::= & \mathbf{skip} \mid Var := AExp \mid Com; Com \mid \\
 & & \mathbf{if}\ BExp\ \mathbf{then}\ Com\ \mathbf{else}\ Com \mid\ \mathbf{while}\ BExp\ \mathbf{do}\ Com
\end{array}
$$

- La definizione di Transizione.
  Relazione binaria $\rightarrow$ che definisce:
  - Esecuzione di un costrutto c nello stato σ della Macchina Astratta
  - La esprimiamo con:
    - $\langle c, \sigma \rangle \rightarrow \tau$, indicante ...
    - $\langle c, \sigma \rangle \rightarrow \langle c', \sigma' \rangle$, indicante ...
    - $\langle c_1, \sigma_1 \rangle \rightarrow \langle c'_1, \sigma'_1 \rangle, \cdots, \langle c_k, \sigma_k \rangle \rightarrow \langle c'_k, \sigma'_k \rangle \ / \ \langle c, \sigma \rangle \rightarrow \langle c', \sigma' \rangle$, k-premesse/1-conclusione, indicante ...
  - Le transizioni sono espresse con regole di inferenza di una teoria del I ordine:

$$\frac{\langle a_2, \sigma \rangle \rightarrow \langle a'', \sigma \rangle}{\langle (a_1 + a_2), \sigma \rangle \rightarrow \langle (a_1 + a''), \sigma \rangle}$$

    con le **variabili** introdotte a sinistra di $\rightarrow$, nella conclusione, quantificate **Universali** e quelle introdotte a destra tutte **Esistenziali**.

  - La semantica di $\mathcal{L}$ fornisce le formule $\langle c, \sigma \rangle \rightarrow \langle c', \sigma' \rangle$ che sono provate vere nella teoria definita per $\mathcal{L}$.

SOS: Computazione

$$
\begin{array}{lll}
Num & ::= & 1 \mid 2 \mid 3 \mid \ldots \\
Var & ::= & X_1 \mid X_2 \mid X_3 \mid \ldots
\end{array}
$$

$$
\begin{array}{lll}
AExp & ::= & Num \mid Var \mid (AExp + AExp) \mid (AExp - AExp) \\
BExp & ::= & \mathbf{tt} \mid \mathbf{ff} \mid (AExp == AExp) \mid \neg BExp \mid (BExp \wedge BExp)
\end{array}
$$

$$
\begin{array}{lll}
Com & ::= & \mathbf{skip} \mid Var := AExp \mid Com; Com \mid \\
 & & \mathbf{if}\ BExp\ \mathbf{then}\ Com\ \mathbf{else}\ Com \mid \ \mathbf{while}\ BExp\ \mathbf{do}\ Com
\end{array}
$$

- La definizione di Computazione di $\mathtt{p} \in \mathcal{L}$.
  - Sequenza degli stati attraversati effettivamente dalle transizioni usate nell'esecuzione del programma $\mathtt{p}$ in uno stato iniziale σ_0
  - Ovvero, la prova dell'esistenza di uno stato finale $\sigma_{\mathtt{n}}$ tale che la formula
    $$\langle \mathtt{P}, \sigma_0 \rangle \rightarrow \sigma_{\mathtt{n}}$$
    è vera nella teoria data come Semantica del Linguaggio.
  - Nel Laboratorio, ricorreremo a queste proprietà:
    - Nel **Progettare** il Linguaggio Small21;
    - Nella **Guida e Verifica dell'Implementazione** della Macchina Astratta di Small21;
    - Nelle **Modifiche e Varianti del Progetto** di Esame di Fine Corso.

SOS: Notazione usata

$$
\begin{array}{lll}
\mathit{Num} & ::= & 1 \mid 2 \mid 3 \mid \ldots \\
\mathit{Var} & ::= & X_1 \mid X_2 \mid X_3 \mid \ldots
\end{array}
$$

$$
\begin{array}{lll}
\mathit{AExp} & ::= & \mathit{Num} \mid \mathit{Var} \mid (\mathit{AExp} + \mathit{AExp}) \mid (\mathit{AExp} - \mathit{AExp}) \\
\mathit{BExp} & ::= & \mathbf{tt} \mid \mathbf{ff} \mid (\mathit{AExp} == \mathit{AExp}) \mid \neg \mathit{BExp} \mid (\mathit{BExp} \wedge \mathit{BExp})
\end{array}
$$

$$
\begin{array}{lll}
\mathit{Com} & ::= & \mathbf{skip} \mid \mathit{Var} := \mathit{AExp} \mid \mathit{Com}; \mathit{Com} \mid \\
& & \mathbf{if}\ \mathit{BExp}\ \mathbf{then}\ \mathit{Com}\ \mathbf{else}\ \mathit{Com} \mid\ \mathbf{while}\ \mathit{BExp}\ \mathbf{do}\ \mathit{Com}
\end{array}
$$

- Notazione.
  $(\mathtt{X}_1, \mathtt{n}_1), \cdots, (\mathtt{X}_k, \mathtt{n}_k)$ stato con k variabili legate
  σ, τ (anche con pedici) sono stati della macchina
  $\sigma(\mathtt{X}_i) = \mathtt{n}_i$ quando $\sigma = (\mathtt{X}_1, \mathtt{n}_1), \cdots, (\mathtt{X}_i, \mathtt{n}_i), \cdots, (\mathtt{X}_k, \mathtt{n}_k)$,
  $\sigma[\mathtt{X}_i \leftarrow \mathtt{m}_i] = (\mathtt{X}_1, \mathtt{n}_1), \cdots, (\mathtt{X}_i, \mathtt{m}_i), \cdots, (\mathtt{X}_k, \mathtt{n}_k)$
  quando $\sigma = (\mathtt{X}_1, \mathtt{n}_1), \cdots, (\mathtt{X}_i, \mathtt{n}_i), \cdots, (\mathtt{X}_k, \mathtt{n}_k)$
  $\mathtt{n}, \mathtt{n}_i \in \mathtt{Num}$ valori numerici
  $\mathtt{a}, \mathtt{a}_i \in \mathtt{AExp}$ espressioni aritmetiche
  $\mathtt{b}, \mathtt{b}_i \in \mathtt{BExp}$ espressioni booleane
  tt, **ff** i valori true e false
  $\mathtt{c}, \mathtt{c}_i \in \mathtt{Com}$ comandi del linguaggio

SOS: Semantica Espressioni Aritmetiche

$$
\begin{aligned}
Num &::= 1 \mid 2 \mid 3 \mid \ldots \\
Var &::= X_1 \mid X_2 \mid X_3 \mid \ldots
\end{aligned}
$$

$$AExp ::= Num \mid Var \mid (AExp + AExp) \mid (AExp - AExp)$$

Semantica delle Espressioni `AExp`.

$$\langle X, \sigma \rangle \to \langle \sigma(X), \sigma \rangle$$

$$\frac{}{\langle (n+m), \sigma \rangle \to \langle p, \sigma \rangle} \text{ where } p = n + m \qquad \frac{}{\langle (n-m), \sigma \rangle \to \langle p, \sigma \rangle} \text{ where } p = n - m \text{ e } n \geq m$$

$$\frac{\langle a_1, \sigma \rangle \to \langle a', \sigma \rangle}{\langle (a_1 + a_2), \sigma \rangle \to \langle (a' + a_2), \sigma \rangle} \qquad \frac{\langle a_2, \sigma \rangle \to \langle a'', \sigma \rangle}{\langle (a_1 + a_2), \sigma \rangle \to \langle (a_1 + a''), \sigma \rangle}$$

$$\frac{\langle a_1, \sigma \rangle \to \langle a', \sigma \rangle}{\langle (a_1 - a_2), \sigma \rangle \to \langle (a' - a_2), \sigma \rangle} \qquad \frac{\langle a_2, \sigma \rangle \to \langle a'', \sigma \rangle}{\langle (a_1 - a_2), \sigma \rangle \to \langle (a_1 - a''), \sigma \rangle}$$

- L'ordine di valutazione degli argomenti delle operazioni è inessenziale
- Se lo volessimo da sinistra a destra: Come modificarla?

SOS: Semantica Espressioni Aritmetiche - Applichiamola

$$AExp = Var \mid Num \mid (AExp + AExp) \mid (AExp - AExp)$$

var: $\dfrac{\mathtt{X} \in \mathtt{Var}}{\langle \mathtt{X}, \sigma\rangle \to \langle \sigma(\mathtt{X}), \sigma\rangle}$	
primPlus: $\dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \oplus \mathtt{m}}{\langle (\mathtt{n} + \mathtt{m}), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}$	primMinus: $\dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \ominus \mathtt{m}}{\langle (\mathtt{n} - \mathtt{m}), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}$
plusL: $\dfrac{\langle \mathtt{a}_1, \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}' + \mathtt{a}_2), \sigma\rangle}$	plusR: $\dfrac{\langle \mathtt{a}_2, \sigma\rangle \to \langle \mathtt{a}'', \sigma\rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}_1 + \mathtt{a}''), \sigma\rangle}$
minusL: $\dfrac{\langle \mathtt{a}_1, \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}' - \mathtt{a}_2), \sigma\rangle}$	minusR: $\dfrac{\langle \mathtt{a}_2, \sigma\rangle \to \langle \mathtt{a}'', \sigma\rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}_1 - \mathtt{a}''), \sigma\rangle}$

Applichiamo le regole all'espressione $\mathtt{a} \equiv ((\mathtt{x} - \mathtt{y}) + 7)$ nello stato $\sigma \equiv (\mathtt{x}, 12), (\mathtt{y}, 3)$

Step 1: PlusL

$$\text{R(PlusL)}^{8}\ \frac{\langle ((\mathtt{x} - \mathtt{y}) + 7), \sigma\rangle \to \langle (\mathtt{a}' + 7), \sigma\rangle}{\langle (\mathtt{x} - \mathtt{y}), \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}$$

Step 1: PlusL

Step 2: MinusL

$$\text{R(plusL)}\ \frac{\langle ((\mathtt{x} - \mathtt{y}) + 7), \sigma\rangle \to \langle (\mathtt{a}' + 7), \sigma\rangle \quad \mathtt{a}' \equiv (\mathtt{x}' - \mathtt{y})}{\text{R(minusL)}\ \dfrac{\langle (\mathtt{x} - \mathtt{y}), \sigma\rangle \to \langle (\mathtt{x}' - \mathtt{y}), \sigma\rangle}{\langle \mathtt{x}, \sigma\rangle \to \langle \mathtt{x}', \sigma\rangle}}$$

– continua

[8] R(c) si legge reversed-c ed indica l'uso invertito della regola c: Dalle conclusioni alle sufficienti premesse.

SOS: Semantica Espressioni Aritmetiche - Applichiamola/2

$$AExp = Var \mid Num \mid (AExp + AExp) \mid (AExp - AExp)$$

var: $\dfrac{\mathtt{X} \in \mathtt{Var}}{\langle \mathtt{X}, \sigma \rangle \to \langle \sigma(\mathtt{X}), \sigma \rangle}$	
primPlus: $\dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \oplus \mathtt{m}}{\langle (\mathtt{n} + \mathtt{m}), \sigma \rangle \to \langle \mathtt{p}, \sigma \rangle}$	primMinus: $\dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \ominus \mathtt{m}}{\langle (\mathtt{n} - \mathtt{m}), \sigma \rangle \to \langle \mathtt{p}, \sigma \rangle}$
plusL: $\dfrac{\langle \mathtt{a}_1, \sigma \rangle \to \langle \mathtt{a}', \sigma \rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}' + \mathtt{a}_2), \sigma \rangle}$	plusR: $\dfrac{\langle \mathtt{a}_2, \sigma \rangle \to \langle \mathtt{a}'', \sigma \rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}_1 + \mathtt{a}''), \sigma \rangle}$
minusL: $\dfrac{\langle \mathtt{a}_1, \sigma \rangle \to \langle \mathtt{a}', \sigma \rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}' - \mathtt{a}_2), \sigma \rangle}$	minusR: $\dfrac{\langle \mathtt{a}_2, \sigma \rangle \to \langle \mathtt{a}'', \sigma \rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}_1 - \mathtt{a}''), \sigma \rangle}$

Step 1: PlusL
Step 2: MinusL
Step 3: Var

$$\mathrm{R(plusL)}\ \frac{\langle ((\mathtt{x} - \mathtt{y}) + 7), \sigma \rangle \to \langle (\mathtt{a}' + 7), \sigma \rangle \quad \mathtt{a}' \equiv (\mathtt{x}' - \mathtt{y})}{\mathrm{R(minusL)}\ \dfrac{\langle (\mathtt{x} - \mathtt{y}), \sigma \rangle \to \langle (\mathtt{x}' - \mathtt{y}), \sigma \rangle}{\mathrm{R(Var)}\ \dfrac{\langle \mathtt{x}, \sigma \rangle \to \langle \mathtt{x}', \sigma \rangle}{\mathtt{x} \in \mathtt{Var},\ \mathtt{x}' = \sigma(\mathtt{x}) = 12}}}$$

Questo conduce alla seguente transizione:
$\langle ((\mathtt{x} - \mathtt{y}) + 7), \sigma \rangle$
$(\mathtt{plusL}, \mathtt{minusL}, \mathtt{Var}) \to \langle ((12 - \mathtt{y}) + 7), \sigma \rangle$

SOS: Semantica Espressioni Aritmetiche - Applichiamola/3

$$AExp = Var \mid Num \mid (AExp + AExp) \mid (AExp - AExp)$$

$\text{var}: \dfrac{\mathtt{X} \in \mathtt{Var}}{\langle \mathtt{X}, \sigma \rangle \to \langle \sigma(\mathtt{X}), \sigma \rangle}$	
$\text{primPlus}: \dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \oplus \mathtt{m}}{\langle (\mathtt{n} + \mathtt{m}), \sigma \rangle \to \langle \mathtt{p}, \sigma \rangle}$	$\text{primMinus}: \dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \ominus \mathtt{m}}{\langle (\mathtt{n} - \mathtt{m}), \sigma \rangle \to \langle \mathtt{p}, \sigma \rangle}$
$\text{plusL}: \dfrac{\langle \mathtt{a}_1, \sigma \rangle \to \langle \mathtt{a}', \sigma \rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}' + \mathtt{a}_2), \sigma \rangle}$	$\text{plusR}: \dfrac{\langle \mathtt{a}_2, \sigma \rangle \to \langle \mathtt{a}'', \sigma \rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}_1 + \mathtt{a}''), \sigma \rangle}$
$\text{minusL}: \dfrac{\langle \mathtt{a}_1, \sigma \rangle \to \langle \mathtt{a}', \sigma \rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}' - \mathtt{a}_2), \sigma \rangle}$	$\text{minusR}: \dfrac{\langle \mathtt{a}_2, \sigma \rangle \to \langle \mathtt{a}'', \sigma \rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma \rangle \to \langle (\mathtt{a}_1 - \mathtt{a}''), \sigma \rangle}$

Step 1: PlusL
Step 2: MinusR
Step 3: Var

$$\text{R(plusL)}\ \frac{\langle ((12 - \mathtt{y}) + 7), \sigma \rangle \to \langle (\mathtt{a}' + 7), \sigma \rangle \quad \mathtt{a}' \equiv (12 - \mathtt{y}')}{\text{R(minusR)}\ \dfrac{\langle (12 - \mathtt{y}), \sigma \rangle \to \langle (12 - \mathtt{y}'), \sigma \rangle}{\text{R(Var)}\ \dfrac{\langle \mathtt{y}, \sigma \rangle \to \langle \mathtt{y}', \sigma \rangle}{\mathtt{y} \in \mathtt{Var},\ \mathtt{y}' = \sigma(\mathtt{y}) = 3}}}$$

Questo conduce alla seguente sequenza di transizioni:

$\langle ((\mathtt{x} - \mathtt{y}) + 7), \sigma \rangle$
$(\mathtt{plusL}, \mathtt{minusL}, \mathtt{Var}) \to \langle ((12 - \mathtt{y}) + 7), \sigma \rangle$
$(\mathtt{minusR}, \mathtt{Var}) \to \langle ((12 - 3) + 7), \sigma \rangle$

SOS: Semantica Espressioni Aritmetiche - Applichiamola/4

$$AExp = Var \mid Num \mid (AExp + AExp) \mid (AExp - AExp)$$

var: $\dfrac{\mathtt{X} \in \mathtt{Var}}{\langle \mathtt{X}, \sigma\rangle \to \langle \sigma(\mathtt{X}), \sigma\rangle}$	
primPlus: $\dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \oplus \mathtt{m}}{\langle (\mathtt{n} + \mathtt{m}), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}$	primMinus: $\dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \ominus \mathtt{m}}{\langle (\mathtt{n} - \mathtt{m}), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}$
plusL: $\dfrac{\langle \mathtt{a}_1, \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}' + \mathtt{a}_2), \sigma\rangle}$	plusR: $\dfrac{\langle \mathtt{a}_2, \sigma\rangle \to \langle \mathtt{a}'', \sigma\rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}_1 + \mathtt{a}''), \sigma\rangle}$
minusL: $\dfrac{\langle \mathtt{a}_1, \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}' - \mathtt{a}_2), \sigma\rangle}$	minusR: $\dfrac{\langle \mathtt{a}_2, \sigma\rangle \to \langle \mathtt{a}'', \sigma\rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}_1 - \mathtt{a}''), \sigma\rangle}$

Step 1: PlusL
Step 2: primMinus

$$\text{R(plusL)}\ \frac{\langle((12-3)+7),\sigma\rangle \to \langle(\mathtt{a}'+7),\sigma\rangle}{\text{R(primMinus)}\ \dfrac{\langle(12-3),\sigma\rangle \to \langle \mathtt{p},\sigma\rangle}{12,3 \in \mathtt{Num},\ \mathtt{p} = 12 \ominus 3}} \qquad \mathtt{a}' \equiv \mathtt{p}$$

Questo conduce alla seguente sequenza di transizioni:

$\langle((\mathtt{x} - \mathtt{y}) + 7), \sigma\rangle$
$(\mathtt{plusL}, \mathtt{minusL}, \mathtt{Var}) \to \langle((12 - \mathtt{y}) + 7), \sigma\rangle$
$(\mathtt{minusR}, \mathtt{Var}) \to \langle((12 - 3) + 7), \sigma\rangle$
$(\mathtt{primMinus}) \to \langle(9 + 7), \sigma\rangle$

$$AExp = Var \mid Num \mid (AExp + AExp) \mid (AExp - AExp)$$

$\text{var}: \dfrac{\mathtt{X} \in \mathtt{Var}}{\langle \mathtt{X}, \sigma\rangle \to \langle \sigma(\mathtt{X}), \sigma\rangle}$	
$\text{primPlus}: \dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \oplus \mathtt{m}}{\langle (\mathtt{n} + \mathtt{m}), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}$	$\text{primMinus}: \dfrac{\mathtt{n}, \mathtt{m} \in \mathtt{Num} \qquad \mathtt{p} = \mathtt{n} \ominus \mathtt{m}}{\langle (\mathtt{n} - \mathtt{m}), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}$
$\text{plusL}: \dfrac{\langle \mathtt{a}_1, \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}' + \mathtt{a}_2), \sigma\rangle}$	$\text{plusR}: \dfrac{\langle \mathtt{a}_2, \sigma\rangle \to \langle \mathtt{a}'', \sigma\rangle}{\langle (\mathtt{a}_1 + \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}_1 + \mathtt{a}''), \sigma\rangle}$
$\text{minusL}: \dfrac{\langle \mathtt{a}_1, \sigma\rangle \to \langle \mathtt{a}', \sigma\rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}' - \mathtt{a}_2), \sigma\rangle}$	$\text{minusR}: \dfrac{\langle \mathtt{a}_2, \sigma\rangle \to \langle \mathtt{a}'', \sigma\rangle}{\langle (\mathtt{a}_1 - \mathtt{a}_2), \sigma\rangle \to \langle (\mathtt{a}_1 - \mathtt{a}''), \sigma\rangle}$

Step 1: primPlus

$$\text{R(primPlus)} \ \frac{\langle (9 + 7), \sigma\rangle \to \langle \mathtt{p}, \sigma\rangle}{9, 7 \in \mathtt{Num}, \mathtt{p} = 9 \oplus 7}$$

Questo conduce alla seguente sequenza di transizioni:

$\langle (\mathtt{x} - \mathtt{y}) + 7), \sigma\rangle$

$(\mathtt{plusL}, \mathtt{minusL}, \mathtt{Var}) \to \langle ((12 - \mathtt{y}) + 7), \sigma\rangle$

$(\mathtt{minusR}, \mathtt{Var}) \to \langle ((12 - 3) + 7), \sigma\rangle$

$(\mathtt{primMinus}) \to \langle (9 + 7), \sigma\rangle$

$(\mathtt{primPlus}) \to \langle 16, \sigma\rangle$

SOS: Semantica dei Comandi

$$Com ::= \textbf{skip} \mid Var := AExp \mid Com; Com \mid \textbf{if } BExp \textbf{ then } Com \textbf{ else } Com \mid \textbf{while } BExp \textbf{ do } Com$$

Semantica dei Comandi `Com`.

$$\langle \textbf{skip}, \sigma \rangle \to \sigma \quad (c1)$$

$$\langle X := n, \sigma \rangle \to \sigma[X \leftarrow n] \quad (c2) \qquad \frac{\langle a, \sigma \rangle \to \langle a', \sigma \rangle}{\langle X := a, \sigma \rangle \to \langle X := a', \sigma \rangle} \quad (c3)$$

$$\frac{\langle c_1, \sigma \rangle \to \sigma'}{\langle c_1; c_2, \sigma \rangle \to \langle c_2, \sigma' \rangle} \quad (c4) \qquad \frac{\langle c_1, \sigma \rangle \to \langle c_1', \sigma' \rangle}{\langle c_1; c_2, \sigma \rangle \to \langle c_1'; c_2, \sigma' \rangle} \quad (c5)$$

$$\langle \textbf{if tt then } c_1 \textbf{ else } c_2, \sigma \rangle \to \langle c_1, \sigma \rangle \quad (c6)$$
$$\langle \textbf{if ff then } c_1 \textbf{ else } c_2, \sigma \rangle \to \langle c_2, \sigma \rangle \quad (c7)$$

$$\frac{\langle b, \sigma \rangle \to \langle b', \sigma \rangle}{\langle \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2, \sigma \rangle \to \langle \textbf{if } b' \textbf{ then } c_1 \textbf{ else } c_2, \sigma \rangle} \quad (c8)$$

$$\langle \textbf{while } b \textbf{ do } c, \sigma \rangle \to \langle \textbf{if } b \textbf{ then } c; \textbf{while } b \textbf{ do } c \textbf{ else skip}, \sigma \rangle \quad (c9)$$

SOS: Semantica dei Comandi

$$\langle \textbf{skip}, \sigma \rangle \to \sigma \quad (c1)$$

$$\langle X := n, \sigma \rangle \to \sigma[X \leftarrow n] \quad (c2) \qquad \frac{\langle a, \sigma \rangle \to \langle a', \sigma \rangle}{\langle X := a, \sigma \rangle \to \langle X := a', \sigma \rangle} \quad (c3)$$

$$\frac{\langle c_1, \sigma \rangle \to \sigma'}{\langle c_1; c_2, \sigma \rangle \to \langle c_2, \sigma' \rangle} \quad (c4) \qquad \frac{\langle c_1, \sigma \rangle \to \langle c_1', \sigma' \rangle}{\langle c_1; c_2, \sigma \rangle \to \langle c_1'; c_2, \sigma' \rangle} \quad (c5)$$

$$\langle \textbf{if tt then } c_1 \textbf{ else } c_2, \sigma \rangle \to \langle c_1, \sigma \rangle \quad (c6)$$
$$\langle \textbf{if ff then } c_1 \textbf{ else } c_2, \sigma \rangle \to \langle c_2, \sigma \rangle \quad (c7)$$

$$\frac{\langle b, \sigma \rangle \to \langle b', \sigma \rangle}{\langle \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2, \sigma \rangle \to \langle \textbf{if } b' \textbf{ then } c_1 \textbf{ else } c_2, \sigma \rangle} \quad (c8)$$

$$\langle \textbf{while } b \textbf{ do } c, \sigma \rangle \to \langle \textbf{if } b \textbf{ then } c; \textbf{while } b \textbf{ do } c \textbf{ else skip}, \sigma \rangle \quad (c9)$$

Sia `c` la sequenza di comandi: `X := 1;` **while** $\neg$`(X == 0)` **do** `X := X - 1`

$$\begin{aligned}
&\langle c, \sigma \rangle \\
&\to \langle c', \sigma[X \leftarrow 1] \rangle \\
&\to \langle \textbf{if } \neg(X == 0) \textbf{ then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 1] \rangle \\
&\to \langle \textbf{if } \neg(1 == 0) \textbf{ then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 1] \rangle \\
&\to \langle \textbf{if } \neg \textbf{ff then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 1] \rangle \\
&\to \langle \textbf{if tt then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 1] \rangle \\
&\to \langle X := (X - 1); c', \sigma[X \leftarrow 1] \rangle \\
&\to \langle X := (1 - 1); c', \sigma[X \leftarrow 1] \rangle \\
&\to \langle X := 0; c', \sigma[X \leftarrow 1] \rangle \\
&\to \langle c', \sigma[X \leftarrow 0] \rangle \\
&\to \langle \textbf{if } \neg(X == 0) \textbf{ then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 0] \rangle \\
&\to \langle \textbf{if } \neg(0 == 0) \textbf{ then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 0] \rangle \\
&\to \langle \textbf{if } \neg \textbf{tt then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 0] \rangle \\
&\to \langle \textbf{if ff then } X := (X - 1); c' \textbf{ else skip}, \sigma[X \leftarrow 0] \rangle \\
&\to \langle \textbf{skip}, \sigma[X \leftarrow 0] \rangle \\
&\to \sigma[X \leftarrow 0]
\end{aligned}$$

Esercizi L5

1. Significato, Uso, Caratteristica di semantica Denotazionale e semantica Operazionale dei L. di Programmazione
2. Quali sono le 3 strutture che devono essere definite per fornire la SOS di un L. di Programmazione?
3. Si modifichino le regole di inferenza delle espressioni aritmetiche, date a lezione, in modo da prescrivere una valutazione degli argomenti da sinistra a destra per somma e sottrazione.
4. Si indichino le regole di inferenza e si mostrino le transizioni ottenute applicando all'espressione (y-(7+2)) la semantica SOS, data a lezione.
5. Si annoti, a sinistra di ogni freccia nella figura sotto (v. testo completo), le regole di inferenza (dei comandi c1-c9) utilizzate nella transizione, come abbiamo visto a lezione nel caso delle espressioni.
6. Siano `c` e `d` i seguenti comandi:
   `c:` `X:=1;`
   `d:` **while** `(X==1)` **do skip**
   Si dia la sequenza di transizioni ottenute per `c;d` a partire dallo stato $\sigma = (\mathtt{X}, 0)$.

Altri esercizi in:
* Esercizi4Lezione4Parte2

* Cap. 2 [GM]