Esercizi 21

3/2/2023

Si consideri la seguente grammatica (scritta in ANTLR)

```
prg :  'let' dec 'in' stm ;
dec :  ('int' Id ';')+ ;
exp :  Integers | Id | exp '+' exp ;
stm :  (Id '=' exp ';')+
```

dove

- gli `Integers` sono sequenze non vuote di cifre prefissate dal segno `+` o `-`;
- gli `Id` sono gli identificatori (sequenze non vuote di caratteri);

Esercizi

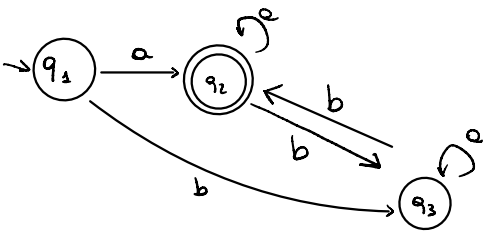
1. (**punti 2**) completare l'input di ANTLR con le regole per l'analizzatore lessicale che riguardano `Integers` e `Id`;

Integers := Sign Number2 Number*
Sign := '+' | '-'
Number := 0..9
Number2 := 1..9

Integer : Sign Number
Sign: +|-
Number : (0 | [1..9][0..9]*)

Id: letter+
letter: a-z | A-Z

Esercizio 1 (6 punti) Si definisca un analizzatore lessicale in `ANTLR` che accetta sequenze di token che a loro volta sono stringhe (non vuote) sull'alfabeto a, b che contengono un numero pari di occorrenze di b.



$\{a \mid b\, a^* b\}^+$

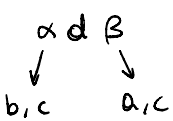
$q_1 : a q_2 \mid b q_3$

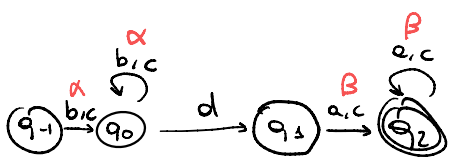
$q_2 : a q_2 \mid b q_3 \mid \varepsilon$ — q_2 finale

$q_3 : a q_3 \mid b q_2$

20/11/21

Esercizio 1 (6 punti). Sia L il linguaggio sull'alfabeto $\{a, b, c, d\}$ costituito da sequenze (non vuote) di token della forma $\alpha d \beta$ dove α è una qualunque stringa non vuota che contiene $\{b, c\}$ e β è una qualunque stringa non vuota che contiene $\{a, c\}$. Ad esempio `ccdc bdc` è una sequenza di token valida, mentre `cc ada` è sbagliata. Si definisca in `ANTLR` l'analizzatore lessicale per tokens in L senza utilizzare gli operatori $*$ o $+$.





$q_{-1} : b q_0 \mid c q_0$

$q_0 : b q_0 \mid c q_0 \mid d q_1$

$q_1 : a q_2 \mid c q_2$

$q_2 : a q_2 \mid c q_2 \mid \varepsilon$

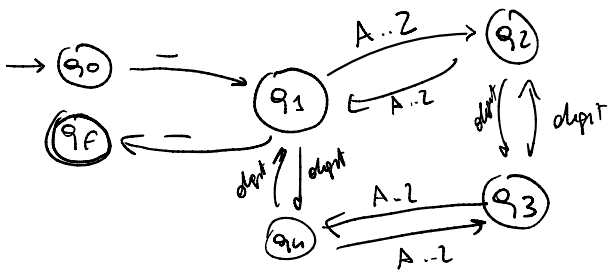
18/9/2020

Esercizio 1 (punti 6) Gli identificatori di un linguaggio di programmazione devono iniziare e terminare con "_" e tra questi due caratteri ci possono essere solo lettere maiuscole e cifre (in qualunque ordine) con il vincolo che il numero di lettere e quello delle cifre sia sempre pari. Definire l'analizzatore lessicale per questi identificatori in `ANTLR`.

15/6/2020

Esercizio 1 (6 punti). Definire un analizzatore lessicale in `ANTLR` che accetta sequenze di token che a loro volta sono stringhe non vuote sull?alfabeto {a, b} per cui non ci sono mai due occorrenze di `b` consecutive. Ad esempio `a abaa b aaaab` è un input riconosciuto.

18/9

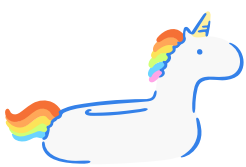


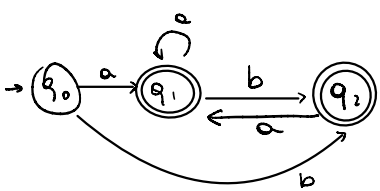
l : A|..|Z
d : 0|..|9

q_0 : $_\, q_1$
q_1 : $l\, q_2 \mid d\, q_4 \mid \_\, q_f$
q_2 : $l\, q_1 \mid d\, q_3$
q_3 : $d\, q_2 \mid l\, q_4$
q_4 : $d\, q_1 \mid l\, q_3$
q_f : ε

15/6/2020

Esercizio 1 (6 punti). Definire un analizzatore lessicale in `ANTLR` che accetta sequenze di token che a loro volta sono stringhe non vuote sull?alfabeto {a, b} per cui non ci sono mai due occorrenze di `b` consecutive. Ad esempio `a abaa b aaaab` è un input riconosciuto.





q_0 : $a\, q_1 \mid b\, q_2$
q_1 : $a\, q_1 \mid b\, q_2 \mid \varepsilon$
q_2 : $a\, q_1 \mid \varepsilon$

Analisi Sintattica

14/2/2019

Esercizio 2. Data la grammatica (le lettere minuscole sono simboli terminali)

$$\begin{array}{ll} S \to & Ab \mid Bc \\ A \to & aA' \\ A' \to & d \mid \varepsilon \\ B \to & aB' \\ B' \to & d \mid \varepsilon \end{array}$$

Si dimostri, costruendo l'opportuna tabella, che la grammatica è $LL(1)$. Nel caso in cui non lo sia, verificare se esiste k tale che la grammatica è $LL(k)$. Motivare la risposta.

5/6/2019

Esercizio 2. Data la grammatica (le lettere minuscole sono simboli terminali)

$$\begin{array}{ll} A \to & Ba \mid C \\ B \to & AA \\ C \to & Cc \mid b \end{array}$$

1. Trasformarla, rimuovendo la (mutua) ricorsione sinistra;
2. verificare se la grammatica ottenuta è $LL(1)$ costruendo l'opportuna tabella.

NULLABLE (S) = NULLABLE (Ab) ∨ NULLABLE (Bc) = F ∨ F = F
NULLABLE (A) = NULL (a) ∧ NULL (A') = F ∧ T = F
NULLABLE (A') = NULL (d) ∨ NULL (ε) = F ∨ T = T
NULLABLE (B) = NULL (a) ∧ NULL (B') = F
NULLABLE (B') = NULL (d) ∨ NULL (ε) = T

NULLABLE (G) = { A', B' }

FIRST (αγ) = { FIRST(α) se α ∉ NULL(G) ; FIRST(α) \ {ε} ∪ FIRST(γ) se α ∈ NULL(G) }

FI (S) = FI (Ab) ∪ FI (Bc)
= FI (A) ∪ FI (B)
= {a}

FI (A) = {a}
FI (A') = {d, ε}
FI (B) = {a}
FI (B') = {d, ε}

FO (S) = {$}
FO (A) = {b}
FO (A') = {b}
FO (B) = {c}
FO (B') = {c}

- FOLLOW (S) = $
- FOLLOW (X) = ∪ FIRST(γ) \ {ε} (Z → δXγ in G) ∪ FOLLOW (Z) (Z → δXγ in G ∧ NULL(γ))

FIRST dopo il mio termine γ ∈ NULL(G)

	a	b	c	d	$
S	S→Ab S→Bc				
A	A→aA'				
A'		A'→ε		A'→d	
B	B→aB'				
B'			B'→ε	B'→d	

14/2/2019

Esercizio 2. Data la grammatica (le lettere minuscole sono simboli terminali)

ad ad

$$\begin{array}{lll} S \to & Ab \mid Bc \\ A \to & aA' & \text{ad} \\ A' \to & d \mid \varepsilon \\ B \to & aB' & \text{ad} \\ B' \to & d \mid \varepsilon \end{array}$$

Si dimostri, costruendo l'opportuna tabella, che la grammatica è $LL(1)$. Nel caso in cui non lo sia, verificare se esiste k tale che la grammatica è $LL(k)$. Motivare la risposta.

5/6/2019

Esercizio 2. Data la grammatica (le lettere minuscole sono simboli terminali)

$$\begin{array}{lll} A \to & Ba \mid C \\ B \to & AA \\ C \to & Cc \mid b \end{array}$$

1. Trasformarla, rimuovendo la (mutua) ricorsione sinistra;
2. verificare se la grammatica ottenuta è $LL(1)$ costruendo l'opportuna tabella.

GG