Corso di Laurea Magistrale in Informatica

Compito di Compilatori e Interpreti

15 Giugno 2020

Nota Bene. Alla fine del compito, fare una foto a tutto il compito col cellulare e inviare le foto per email a `cosimo.laneve@unibo.it` .

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 $\{\texttt{a}, \texttt{b}\}$ per cui non ci sono mai due occorrenze di `b` consecutive. Ad esempio `a abaa b aaaab` è un input riconosciuto.

Esercizio 2 (7 punti). Data la grammatica (le lettere minuscole sono simboli terminali, A è il simbolo iniziale)

$$\begin{array}{lllllll} S \rightarrow & Aa & | & bAc & |Bc & | & bBa \\ A \rightarrow & aA & | & \varepsilon & & & \\ B \rightarrow & bB & | & \varepsilon & & & \end{array}$$

Verificare se la grammatica è LL(1) costruendo l'opportuna tabella. Nel caso non lo sia, esiste un k per cui essa è LL(k)? Motivare la risposta.

Esercizio 3 (10 punti). Definire la funzione `code_gen` per il comando

```
for id := E to E' do S
```

La semantica del for è: (1) si calcolano il valore delle espressioni `E` e `E'` e siano esse v e v'; (2) quindi si inizializza `id` a v e si esegue `S` se `id` $\leq v'$; (3) dopo l'esecuzione di `S`, si incrementa `id` e si riverifica se `id` $\leq v'$. L'iterazione termina quando `id` $> v'$.

Si applichi tale regola al comando

```
for x := y to z do z := x+1
```

assumendo che le variabili `x` e `y` si trovino nel record di attivazione corrente ad offset 8 e 12 del `$fp`, mentre `z` si trovi nell'ambiente statico immediatamente precedente a offset 8.