Corso di Linguaggi di Programmazione — Parziale di fine modulo
Prova scritta **B** del 20 Dicembre 2013.

Tempo a disposizione: 2 ore e 30 minuti.

1. Considerare l'espressione regolare $a^*\epsilon b^*$. Costruire l'associato NFA seguendo la costruzione canonica vista a lezione.
2. Prendere l'NFA costruito al punto 1) e renderlo deterministico attraverso il procedimento di costruzione dei sottoinsiemi.
3. Osservare che il DFA prodotto al punto 2) non è minimo, usando l'algoritmo a tabella iterativo, ovvero verificare che almeno due stati sono tra loro equivalenti. Costruire il DFA minimo.
4. Dato il DFA minimo del punto 3), costruire l'associata grammatica regolare. Rimuovere gli eventuali simboli inutili e ricostruire dalla grammatica risultante l'espressione regolare associata.
5. Dati due DFA $M_1 = (\Sigma, Q_1, \delta_1, q_{01}, F_1)$ e $M_2 = (\Sigma, Q_2, \delta_2, q_{02}, F_2)$ tali che $Q_1 \cap Q_2 = \emptyset$, sia $M = (\Sigma, Q_1 \times Q_2, \delta, (q_{01}, q_{02}), F_1 \times F_2)$ con $\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$. Dire se M è un DFA. Qual è il linguaggio riconosciuto da M se $L_1 = L[M_1]$ e $L_2 = L[M_2]$?
6. Una delle seguenti due affermazioni non è corretta: motivare la risposta. (i) Ogni linguaggio regolare è generabile, per ogni $k \geq 1$, da una grammatica di classe LR(k). (ii) Ogni linguaggio libero nondeterministico può essere generato solo da grammatiche ambigue.
7. Mostrare che $L_1 = \{a^n b^{2n} c^m \mid n, m \geq 0\}$ è libero deterministico, costruendo un opportuno DPDA.
8. Sapendo che anche $L_2 = \{a^n b^m c^{2m} \mid n, m \geq 0\}$ è libero deterministico, è libero il linguaggio $L_1 \cap L_2$, dove L_1 è il linguaggio dell'esercizio sopra?
9. Classificare il linguaggio $L = \{ww^R \mid w \in a^*\}$, ovvero dire se L è regolare, oppure libero ma non regolare, oppure non libero, giustificando adeguatamente la risposta.
10. Data la seguente grammatica G con simbolo iniziale S:

$$\begin{array}{lcl} S & \rightarrow & \epsilon \mid AB \\ A & \rightarrow & \mathtt{a}A\mathtt{b} \mid \epsilon \\ B & \rightarrow & \mathtt{b} \mid \mathtt{b}B \end{array}$$

si determini il linguaggio $L(G)$ generato da G. Si semplifichi G rimuovendo le produzioni epsilon, ottenendo la grammatica G'. Si discuta se la grammatica risultante G' sia del tutto equivalente alla grammatica G.

11. Si consideri la grammatica G' ottenuta al passo precedente. Si verifichi se G' è di classe LL(1). Se non lo è, si manipoli opportunamente G' per trasformarla in una grammatica equivalente G'' di classe LL(1). Si costruisca quindi la tabella di parsing LL(1). Si mostri il funzionamento del parser LL(1) sull'input $aabb$.
12. Data la grammatica G del punto 10), si verifichi che non è di classe LR(0), ma è di classe SLR(1). Si mostri il funzionamento del parser SLR(1) sull'input $aabb$.
13. Si consideri la grammatica G

$$C \rightarrow \ \mathtt{a} \mid \textbf{if } \mathtt{b} \textbf{ then } \mathrm{C} \mid \textbf{if } \mathtt{b} \textbf{ then } \mathrm{C} \textbf{ else } \mathrm{C}$$

che esprime comandi composti in modo condizionale a partire da una istruzione elementare a e da una espressione booleana b, non meglio specificate. Si dimostri che la grammatica G è ambigua. Definire le regole di semantica operazionale strutturata (SOS) per il costrutto **if** b **then** c_1 **else** c_2.