

PROGRAMMAZIONE 1 e LABORATORIO (A,B) - a.a. 2013/14

Prova scritta del 3 febbraio 2014

Scrivere **IN STAMPATELLO** COGNOME, NOME, MATRICOLA e CORSO su ogni foglio consegnato

ESERCIZIO 1 (punti 5)

Si supponga di avere una grammatica $G = \langle \Lambda, V, S, P \rangle$, con $\Lambda = \{a, b, c\}$, che genera il linguaggio

$$\mathcal{L}_G = \{a^n b^n c^n \mid n > 0\}$$

Definire una grammatica

$$G' = \langle \Lambda, V \cup V', S', P \cup P' \rangle$$

con $V' \cap V = \{\}$ che genera il linguaggio

$$\mathcal{L}_{G'} = \{a^m b^n c^n \mid m > n > 0\} \cup \{a^n b^n c^m \mid m > n > 0\}$$

ESERCIZIO 2 (punti 5)

Definire un automa **deterministico** che riconosce il seguente linguaggio sull'alfabeto $\Lambda = \{a, b, c\}$

$$\mathcal{L} = \{\alpha abc\beta \mid \alpha \in \Lambda^*, \beta \in \{a, b\}^*\}$$

ESERCIZIO 3 (punti 5)

Dato il tipo degli alberi binari

```
type 'a btree = Void | Node of 'a * 'a btree * 'a btree
```

si definisca in CAML una funzione `foo` con tipo

```
foo : 'a btree -> int * int
```

in modo che `(foo bt)` sia la coppia costituita dalla profondità dell'albero e dal numero di nodi dell'albero.

ESERCIZIO 4 (punti 5)

Si definisca in C una funzione

```
int check (int a[], int dim)
```

che restituisce il valore di verità della seguente formula:

$$\exists j \in [1, dim - 1]. \#\{i \mid i \in [0, j) \wedge a[j] < a[i]\} = \#\{i \mid i \in [j + 1, dim) \wedge a[j] > a[i]\}$$

N.B. : si ricorda che, dato un insieme A , la notazione $\#A$ indica la cardinalità (ovvero il numero di elementi) di A .

ESERCIZIO 5 (punti 5)

Senza utilizzare ricorsione esplicita, definire in CAML una funzione

```
foo : 'a list -> 'a * 'a
```

in modo che `(foo lis)` restituisca la coppia `(p, m)` dove `p` è il primo elemento della lista e `m` è il valore massimo nella lista. La funzione non è definita su liste vuote.

ESERCIZIO 6 (punti 5)

Date le seguenti definizioni:

```
struct el {int info; struct el *next;};  
typedef struct el ElementoDiLista;  
typedef ElementoDiLista *ListaDiElementi;
```

scrivere in C una procedura che, dati in ingresso attraverso opportuni parametri una lista di interi e due interi `x` e `y`, inserisce nella lista un elemento che contiene `x` come predecessore dell'ultimo elemento che contiene `y`. Se nessun elemento della lista contiene `y`, la procedura inserisce un elemento che contiene `x` in fondo alla lista.