

PROGRAMMAZIONE 1 e LABORATORIO (A,B)

a.a. 2015-2016 - Prova scritta del 6/07/2016

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

ESERCIZIO 1 (6 punti)

Si definisca una grammatica libera sull'alfabeto $\Lambda = \{ (,), [,], \{, \} \}$ che genera le stringhe di parentesi bilanciate in modo che le parentesi quadrate e graffe non stiano mai all'interno di parentesi tonde e le parentesi graffe non stiano mai all'interno di parentesi quadrate. Ad esempio, detto \mathcal{L} il linguaggio generato dalla grammatica:

$$([]()) \notin \mathcal{L} \quad \{ () [] (()) \} \in \mathcal{L}$$

ESERCIZIO 2 (6 punti)

Dato un albero binario, si definisce *livello* di un nodo nell'albero il numero di nodi che si incontrano nel cammino dalla radice al nodo medesimo. Quindi, ad esempio, la radice ha livello 1, i figli della radice hanno livello 2, e così via. Dato il tipo degli alberi binari visti a lezione

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

si scriva in CAML una funzione

```
leaves : 'a btree -> 'a list * 'a list
```

che, dato un albero binario, restituisce una coppia di liste (l_1, l_2) in cui l_1 contiene i valori dei nodi non foglia di livello pari e l_2 contiene i valori dei nodi non foglia di livello dispari.

ESERCIZIO 3 (6 punti)

Si definisca in CAML, senza utilizzare ricorsione esplicita, una funzione

```
minfirst : 'a list -> 'a list
```

in modo che $(\text{minfirst } xs)$ sia una lista che contiene tutti gli elementi di xs , in cui le occorrenze dell'elemento minimo sono tutte in testa (l'ordine in cui gli elementi non minimi di xs occorrono nella lista risultato è irrilevante). Ad esempio

```
minfirst [2;3;4;1;3;1;6] = [1;1] @ L
```

dove L è una qualunque permutazione della lista $[2;3;4;3;6]$

ESERCIZIO 4 (6 punti)

Si scriva in C una procedura che, presi attraverso opportuni parametri una lista di interi ed un intero n , cancella dalla lista i primi n elementi maggiori di 0. Si suppongano predefiniti i tipi:

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

ESERCIZIO 5 (6 punti)

Si scriva in C una funzione che, dato un array di interi a e la sua dimensione dim , restituisce il valore di verità della seguente formula

$$\exists i, j \in [0, dim). a[i] \neq a[j] \wedge \#\{k \mid k \in [0, dim) \wedge a[k] = a[i]\} = \#\{h \mid h \in [0, dim) \wedge a[h] = a[j]\}$$

Si ricorda che dato un insieme finito A , $\#A$ indica il numero di elementi di A .