Fondamenti di Programmazione - CdL in MATEMATICA

I Appello del 22/1/2015

num. eserc.	1	2	3	4
punt. tot	8	6	7	10

N.B.:

- Negli esercizi di programmazione, viene valutata anche la leggibilità del codice proposto.
- Inoltre, non è consentito l'uso di istruzioni che alterino il normale flusso dell'esecuzione (come, ad esempio, `continue`, `break` e istruzioni di `return` all'interno di cicli che ne provochino l'uscita forzata).
- Non è consentito l'uso di variabili statiche.
- Laddove è utilizzato, il tipo `boolean` è definito da `typedef enum {false,true} boolean`.

ESERCIZIO 1 (8 punti)
Sia L il linguaggio regolare su $\Sigma = \{1, 2, 3\}$ tale che le stringhe w di L:

- cominciano sempre per 1
- non contengono mai le sequenze $21, 22, 31, 33$;
- finiscono sempre per 3.

Si forniscano allora:

- l'automa a stati finiti non deterministico A in grado di riconoscere L;
- la corrispondente espressione regolare, applicando l'algoritmo di eliminazione degli stati all'automa A;
- la corrispondente grammatica *regolare*, usando l'algoritmo per passare dalle transizioni dell'automa alle produzioni della grammatica.

ESERCIZIO 2 (6 punti)
Scrivere in C una funzione `boolean check (int a[], int b[], int dimA, int dimB)` che dati due array `a` e `b` di dimensioni `dimA` e `dimB`, rispettivamente, restituisce il valore di verità della seguente formula:

$$\forall i \in [0, dimA).\ (\#\{j | j \in [0, dimB) \wedge a[i] = b[j]\} = 0\ \vee\ \#\{j | j \in [0, dimB) \wedge a[i] = b[j]\} \geq 3)$$

dove con $\#S$ indichiamo la cardinalità (ovvero il numero di elementi) di un insieme S.

ESERCIZIO 3 (7 punti)
Scrivere in C una funzione *ricorsiva* che dato un array di caratteri `c` di dimensioni `dim`, due caratteri c_1 e c_2 (non necessariamente distinti), e un numero naturale `n`, controlli se il numero di caratteri dell'array che sono *immediatamente* preceduti da c_1 e *immediatamente* seguiti da c_2 è uguale *esattamente* ad `n`.
Se ad esempio l'array fosse

'a'	'a'	'b'	'c'	'a'	'd'	'b

e i caratteri fossero c_1 = 'a' e c_2= 'b', e `n` = 2, la funzione restituirebbe `true`, dato che sono esattamente due i caratteri con la proprietà desiderata (`a[1]` e `a[5]`).

ESERCIZIO 4 (10 punti)
Data una lista di interi definita come:

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

- Scrivere una procedura *iterativa* C che, data una lista di interi in cui tutti gli elementi con valore negativo precedono tutti gli elementi con valore positivo, e dato un intero `v`, inserisce nella lista un elemento che contiene `v`, in modo che anche nella lista risultante tutti gli elementi con valore negativo precedano tutti gli elementi con valore positivo.
- Scrivere in C una funzione *ricorsiva* che, dati in ingresso attraverso opportuni parametri una lista di interi, elimini dalla lista tutti gli elementi con valore pari e restituisca il numero degli elementi dispari.