FONDAMENTI DI PROGRAMMAZIONE - CdL in MATEMATICA
PROVA SCRITTA DEL 11/7/2011

Scrivere **in stampatello** COGNOME, NOME e MATRICOLA su ogni foglio consegnato

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). Infine non è consentito l'uso di variabili statiche.
Laddove è utilizzato, il tipo `boolean` è definito da `typedef enum {false, true} boolean;`

ESERCIZIO 1 (7 punti)
Dato l'alfabeto $\Sigma = \{1, 2, 3, 4\}$, sia L il linguaggio

$$L = \{a\mathbf{w}b | a, b \in \Sigma \wedge a < b \wedge \mathbf{w} \in \Sigma^*\}$$

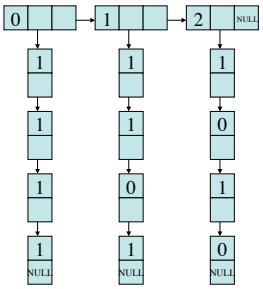
1. Progettare l'automa a stati finiti non deterministici (NFA) che riconosce L.
2. Indipendentemente dal punto (1), definire la grammatica *libera* (non necessariamente regolare) che genera L.

ESERCIZIO 2 (6 punti)
Si scriva una procedura che legge una sequenza di interi terminati da 0 e produce in output `true` se il numero di elementi multipli di 3 è strettamente inferiore al numero di elementi multipli di 5, produce `false` altrimenti. Si risolva il problema, senza utilizzare

- costrutti iterativi; né
- strutture di appoggio, tipo liste o vettori.

ESERCIZIO 3 (7 punti)
Siano dati tre array di interi: i primi due, di dimensione `dim`, sono array ordinati, mentre il terzo,di dimensione `2*dim`, ha tutte le posizioni inizializzate a 0. Progettare una procedura `merge` che dati in ingresso i tre array, popoli il terzo array con gli elementi dei primi due, mantenendo l'ordine e includendo eventuali ripetizioni. Se ad esempio il primo array contenesse `2 3 4 7 13`, e il secondo `5 6 9 11 13 19`, il terzo array alla fine dovrebbe contenere i numeri `2 3 4 5 6 7 9 11 13 19`.

ESERCIZIO 4 (11 punti)
Si vuole rappresentare un sistema di prenotazione posti su treni. Vogliamo quindi avere una lista concatenata, dove ogni elemento rappresenta una carrozza, con il suo numero di posti liberi e la sua sottolista dei posti in dotazione. Ad ogni posto è associato un campo che ci dice se il posto è libero oppure no. Di seguito riportiamo un esempio.



- (1 punti) Definire i tipi opportuni per la rappresentazione indicata.
- (5 punti) Scrivere una funzione *iterativa* che cerchi la prima carrozza, in cui si trovano n posti liberi contigui e che restituisca il puntatore al primo posto libero individuato, oppure `NULL`, in caso di insuccesso.
- (6 punti) Supponendo che nell'elemento carrozza esista un ulteriore campo che ci dica se la carrozza ha tutti i posti liberi oppure no, scrivere una procedura *ricorsiva* che, data la lista delle carrozze, individui ed elimini la prima carrozza, i cui posti risultino tutti liberi.