Programmazione I
Prova di programmazione – 30 gennaio 2018 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma di gestione del noleggio di un insieme di *N* auto, in cui ogni auto è individuata da un *identificatore* unico, costituito a sua volta da un numero da 0 ad *N*-1. All'avvio del programma, l'insieme è vuoto, ossia *N*=0. Il programma deve fornire le seguenti funzionalità.

1. **inizializza_insieme(N)** Inizializza, oppure, se già inizializzato precedentemente, reinizializza l'insieme sequenza affinché contenga **N** auto, tutte disponibili per il noleggio. L'eventuale precedente stato dell'insieme è perso.
2. **noleggia_auto()** Estrae un'auto qualsiasi dell'insieme (la scelta è lasciata al programmatore). Si ottiene il voto massimo se si realizza questa funzionalità con costo computazionale costante rispetto ad *N*, e si imposta la struttura dati in maniera tale da rendere possibile la realizzazione a costo costante anche dei punti 6 e 7. Non sono necessarie queste ottimizzazioni per ottenere la sufficienza.
3. **stampa_stato** Stampa gli identificatori delle auto disponibili per il noleggio. Ad esempio, per un insieme da 5 auto, lo stato potrebbe essere il seguente (qualsiasi ordine di stampa degli identificatori delle auto va bene):
   ```
   3 1 4
   ```
4. **salva_stato** Salva lo stato dell'insieme in un file **binario** dal nome definito a tempo di scrittura del programma. In merito, possono esservi utili le seguenti note
   NOTA: se si trasferisce su file il contenuto di una struct che a sua volta contiene un campo puntatore ad un array, allora, in quanto a tale campo, si trasferisce nel file il contenuto del **puntatore**, e non il contenuto dell'array.
   NOTA: Dato un campo **a**, di tipo array, contenuto in un oggetto **b** di tipo struct, l'indirizzo del primo elemento dell'array è dato dall'espressione **b.a**.
   NOTA: dato un campo **a**, non di tipo array, contenuto in un oggetto **b** di tipo struct, l'indirizzo del campo a è dato dall'espressione **&b.a**.
5. **carica_stato** Carica lo stato dell'insieme dal file. Il contenuto precedente è perso.
6. **auto_disponibile(id)** Ritorna vero se l'auto di identificatore **id** è disponibile per il noleggio. Si ottiene il voto massimo se si realizza questa funzionalità con costo computazionale costante rispetto ad *N*.
7. **restituisci_auto(id)** Reinserisce l'auto di identificatore **id**. Ritorna vero solo se tale auto non era già presente nell'insieme delle auto disponibili. Si ottiene il voto massimo se si realizza questa funzionalità con costo computazionale costante rispetto ad *N*.

I parametri di ingresso delle funzionalità sono solo indicativi. Gestire opportunamente le situazioni di errore, tranne l'*overflow* e l'inserimento di dati in formato errato o di parole troppo lunghe da *stdin*.

REGOLE

- Si può utilizzare ogni genere di manuale e di materiale didattico
- Per superare la prova, bisogna svolgere almeno i punti 1, 2 e 3. Se si svolgono solo tali punti, il programma deve essere perfettamente funzionante. Il voto ottenuto in questo caso è 18.
- Ciascuna funzionalità DEVE essere implementata mediante almeno una funzione.
- Il voto massimo (almeno 30) si ottiene se
  a) il programma è perfettamente funzionante in ogni sua parte
  b) tutti i principi di ingegneria del codice visti nel corso sono stati applicati