Programmazione I
Prova di Programmazione – 24 febbraio 2016 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma di gestione del carico di *N* furgoni per il trasporto di pacchi. *N* è fissato a tempo di scrittura. Ciascun furgone può contenere un numero di pacchi che **varia durante l'esecuzione del programma** e può diventare **arbitrariamente grande** (il numero massimo non è noto a tempo di scrittura del programma). Ogni pacco è identificato da un *codice*, costituito da una parola la cui lunghezza massima è fissata a tempo di scrittura del programma. All'avvio del programma tutti i furgoni sono vuoti. Realizzare le seguenti funzionalità.

1. **aggiungi_pacco(fur, p)** Aggiunge il pacco di codice **p** al furgone **fur** (**fur** è un indice).
2. **stampa_carico()** Stampa lo stato di carico dei furgoni. In particolare, per ciascun furgone stampa l'indice del furgone e la sequenza di codici dei pacchi nel furgone, separati da spazi. Ad esempio, un possibile output con *N*=3 potrebbe essere:
   ```
   0 d2o se1w uyuy2
   1 dh2h qwiuy2
   2 akjsh akjhsd 827dha
   ```
3. **salva_stato()** Salva lo stato dei furgoni in un file di testo definito a tempo di scrittura del programma.
4. **carica_stato()** Ricarica lo stato dal file. L'eventuale precedente stato è perso.
5. **copia_carico(f1, f2)** Copia nel furgone di indice **f2** il carico del furgone di indice **f1**. In altre parole, i furgoni avranno poi lo stesso carico. L'eventuale precedente carico di **f2** è perso. Per chiarezza, questa funzionalità va realizzata in modo tale che, se si aggiungono altri pacchi al furgone **f2** dopo aver invocato questa funzionalità, tali pacchi non si ritrovano poi automaticamente presenti anche nel furgone **f1**.
6. **sposta_carico(f1, f2)** Sposta nel furgone di indice **f2** il carico del furgone di indice **f1**. Quindi il furgone **f1** risulterà poi vuoto. L'eventuale precedente carico di **f2** è perso. Per chiarezza, questa funzionalità va realizzata in modo tale che, se si aggiungono altri pacchi al furgone **f2** dopo aver invocato questa funzionalità, tali pacchi non si ritrovano poi automaticamente presenti anche nel furgone **f1**. Si ottiene il punteggio massimo se si realizza questa funzionalità con costo computazionale minimo.

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

REGOLE

- Si può utilizzare ogni genere di manuale o di materiale didattico di altra natura
- Per superare la prova, il programma deve essere perfettamente funzionante nelle parti 1 e 2. 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