Programmazione I
Prova di programmazione – 12 settembre 2017 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma di gestione di una sequenza di comandi per un dispositivo di lettura dati. I comandi possono essere di due tipi: comando di *controllo*, con cui si passa un *codice di controllo* (valore intero) al dispositivo, e comando di *lettura*, con cui si legge una parola di al più *N* caratteri dal dispositivo. Ogni comando può essere in due stati: *da eseguire* o *eseguito*. Il numero totale massimo *M* di comandi presenti, tra quelli di *controllo* e di *lettura*, e tra quelli in stato *da eseguire* ed *eseguito*, è deciso a tempo di scrittura del programma. Ciascun comando di lettura è associato ad un buffer di caratteri, in cui sarà memorizzata la parola letta dal dispositivo. All'avvio del programma, non ci sono comandi, in alcuno stato. **E' obbligatorio** che ciascun comando occupi memoria per il suo buffer di caratteri solo se è un comando di lettura, e solo dopo che il comando stesso è stato effettivamente inserito per l'esecuzione (vedi il primo punto sotto). Va benissimo utilizzare lo stesso tipo di dato per entrambi i tipi di comando, purché si rispetti il precedente vincolo. **Si ottiene il voto massimo** se la struttura dati in cui sono memorizzati i comandi non occupa più memoria di quella necessaria per memorizzare *M* comandi. **Non si ottiene nessun incremento di voto** se si realizza un versione, più complessa, del programma in cui la memoria occupata è ulteriormente limitata, in ogni istante, a quella necessaria per contenere solo il numero di comandi effettivamente presenti. Il programma deve fornire le seguenti funzionalità.

1. **inserisci_comando(t)** Inserisce un comando *da eseguire* di tipo **t**, ove **t** può essere *controllo* oppure *lettura*. Se si tratta di un comando di controllo, legge da *stdin* il codice di controllo per il comando.
2. **esegui_prossimo_comando** Esegue il primo comando inserito tra quelli ancora in stato da eseguire. Il comando passa in stato *eseguito*. L'esecuzione del comando è simulata come segue: non si fa nulla se si tratta di comando di controllo, si legge da *stdin* una parola, e la si mette nel buffer del comando, se si tratta di comando di lettura.
3. **stampa_comandi** Stampa i comandi eseguiti, nell'ordine in cui sono stati eseguiti, seguiti dai comandi ancora da eseguire, nell'ordine in cui andranno eseguiti. Per i comandi di controllo, stampa il codice di controllo. Per i comandi di lettura eseguiti, stampa il contenuto del buffer. Ad esempio:

```
Eseguiti
lettura pluto
controllo 12
lettura pippo

Da eseguire
lettura
```

4. **salva_stato** Salva lo *stato del sistema,* ossia le sequenze di comandi da eseguire ed eseguiti, nonché i contenuti dei buffer, in un file di testo dal nome definito a tempo di scrittura del programma.
5. **carica_stato** Carica lo stato del sistema dal file. Il contenuto precedente è perso.
6. **cerca_parola(p)** Cerca la parola **p** nella concatenazione delle parole contenute nei buffer dei comandi eseguiti, e ritorna vero se la parola è presente. Ad esempio, ritornerebbe vero se si cercasse la parola *topi*, e lo stato del sistema fosse quello illustrato nell'esempio al punto 3. Si ottiene il punteggio massimo se si implementa questa funzionalità senza creare array dinamici temporanei per contenere la concatenazione delle parole contenute nei buffer dei comandi.

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 e di parole troppo lunghe 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