Programmazione I
Prova di programmazione – 13 Gennaio 2019 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma di gestione dell'ordine di esecuzione di un insieme di generici compiti. Il numero *N* di compiti nell'insieme può variare durante l'esecuzione del programma, come illustrato nella descrizione delle funzionalità. I compiti sono indicizzati, da 0 ad *N*-1, ed il compito di indice *i* ha priorità maggiore dei compiti di indice maggiore di *i*. I compiti possono essere *accesi* oppure *spenti*. All'avvio del programma l'insieme dei compiti è vuoto. Il programma fornisce le seguenti funzionalità.

1. **inizializza_insieme(N)** Inizializza l'insieme dei compiti a contenere **N** processi, tutti spenti. L'eventuale contenuto precedente dell'insieme è perso.
2. **accendi_compito(i)** Accende il compito di indice **i**.
3. **stampa_stato** Stampa il compito in esecuzione e l'insieme dei compiti accesi ma non in esecuzione. Stampa -1 per l'indice del compito in esecuzione se nessun compito è in esecuzione. Il formato di stampa deve essere come nel seguente esempio, in cui il compito in esecuzione è quello di indice 3 (si può avere un compito in esecuzione solo se si è realizzata anche la prossima funzionalità):
   ```
   Esecuzione: 3
   Accesi: 4 6
   ```
4. **esegui_prossimo_compito()** Manda in esecuzione il compito acceso a priorità massima. Se c'è già un compito in esecuzione, ne interrompe l'esecuzione e lo spegne. Ad esempio, se si invoca questa funzionalità a partire dallo stato di cui all'esempio al punto 3, il nuovo stato diviene
   ```
   Esecuzione: 4
   Accesi: 6
   ```
5. **salva_stato** Salva lo stato del programma in un file di testo dal nome predefinito.
6. **carica_stato** Carica lo stato dal file. L'eventuale precedente insieme dei compiti è perso.
7. **esegui_prossimo_compito2()** Uguale al punto 4, a parte la seguente differenza: ogni quattro esecuzioni manda in esecuzione il compito acceso che aspetta da più tempo, indipendentemente dalla sua priorità.

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 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
  c) sono state seguite eventuali altre indicazioni presenti nella traccia in merito al voto finale