Programmazione I
Prova di programmazione – 15 gennaio 2018 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma di gestione dello stato, e solo dello stato, di una sequenza di N blocchi di memoria. All'avvio del programma la sequenza è vuota, ossia N=0. Il programma deve fornire le seguenti funzionalità.

1. **inizializza_sequenza(N)** Inizializza, oppure, se già inizializzata precedentemente, reinizializza la sequenza affinché contenga **N** blocchi. I blocchi sono tutti liberi. L'eventuale precedente stato della sequenza è perso.
2. **alloca_blocchi(idx, n)** Prova ad allocare **n** blocchi consecutivi, a partire dal blocco di indice **idx**. Se tali blocchi sono tutti liberi, allora li fa passare tutti in stato occupato, e ritorna vero. Altrimenti lascia lo stato della sequenza inalterato, e ritorna falso.
3. **stampa_stato** Stampa lo stato della sequenza di blocchi, attraverso una sequenza di due possibili simboli, un **–** oppure un *****. Se nella posizione *i*-esima della sequenza di simboli appare un **–**, allora il blocco di indice *i* è libero, altrimenti il blocco è occupato. Ad esempio, per una sequenza di 15 blocchi, lo stato potrebbe essere
   –––*–––***––**
   ossia, il primo blocco è libero, i successivi due sono occupati, poi ci sono due blocchi liberi, uno occupato, e così via.
4. **salva_stato** Salva lo stato della sequenza in un file di testo dal nome definito a tempo di scrittura del programma.
5. **carica_stato** Carica lo stato della sequenza dal file. Il contenuto precedente è perso.
6. **blocchi_liberi(idx)** Ritorna il numero di blocchi liberi consecutivi presenti a partire dall'indice **idx**. Ad esempio, se lo stato della sequenza è come riportato nell'esempio al punto 3, ecco dei possibili valori di ritorno, in funzione dell'indice: idx=0 → 1, idx=2 → 0, idx=3 → 2, idx=6 → 4.
7. **cerca_alloca_blocchi(n)** Cerca un indice a partire dal quale vi sono almeno **n** blocchi liberi consecutivi, e, se lo trova, alloca **n** blocchi consecutivi a partire da tale indice, ossia fa passare tali blocchi in stato occupato, e ritorna vero. Se non trova un tale indice, ritorna falso. Ad esempio, se questa funzionalità è eseguita, con n=3, sulla sequenza il cui stato è riportato nell'esempio al punto 3, la funzionalità scoprirebbe che ci sono almeno 3 blocchi liberi consecutivi a partire dall'indice 6, ed il nuovo stato della sequenza diverrebbe
   –––****–***––**
8. **cerca_alloca_blocchi2(n)** Identica alla precedente, ma cerca l'indice a partire dal quale vi è il numero più grande di blocchi libero consecutivi possibile. Ad esempio, se questa funzionalità è eseguita, con n=1 o con n=2, sulla sequenza il cui stato è riportato nell'esempio al punto 3, allocherebbe in entrambi i casi i blocchi a partire dall'indice 6.

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 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