Programmazione I
Prova di programmazione – 11 Settembre 2020 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma per l'analisi dell'utilizzo di oggetti allocati in memoria dinamica. Ciascun oggetto è identificato univocamente da un indice, dato da numero naturale. Sono previste tre operazioni: *Allocazione*, *Accesso* e *Deallocazione*. Tali operazioni hanno per operando l'indirizzo dell'oggetto per cui eseguire l'operazione, e sono indicate, rispettivamente, con le espressioni: *A <indice>*, *X <indice>, D <indice>*. Il programma permette di memorizzare ed analizzare una generica sequenza di tali operazioni, come mostrato in dettaglio nella descrizione delle funzionalità. All'avvio del programma, la sequenza è vuota. Il programma fornisce le seguenti funzionalità.

1. **inizializza_sequenza(N)** Inizializza la sequenza a contenere **N** operazioni. L'eventuale precedente contenuto della sequenza è perso. Il valore massimo possibile per **N** non è noto a tempo di scrittura del programma. Se **N** == 0, allora la sequenza avrà lunghezza nulla. Se **N**>0, allora, per ciascuna operazione la funzionalità legge da *stdin* la lettera che rappresenta l'operazione---che deve essere *A*, *X* o *D*---e l'indice dell'oggetto su cui è eseguita l'operazione, che deve essere un numero naturale compreso tra 1 ed **N**. Alla lettura di input non conforme a queste specifiche, la funzionalità smette di leggere da *stdin* e fallisce. La sequenza risulterà vuota. La funzionalità non effettua alcun altro controllo sull'ingresso oltre a questo.
2. **stampa_sequenza** Stampa la sequenza delle operazioni, scrivendo una operazione per ogni riga. Come mostrato nel seguente esempio, per una sequenza con N=6:
   ```
   A 2
   A 4
   X 2
   D 2
   A 6
   D 4
   ```
3. **[2] salva_sequenza** Salva la sequenza in un file di testo dal nome predefinito.
4. **[3] carica_sequenza** Carica la sequenza dal file. L'eventuale precedente contenuto è perso.
5. **[4] stampa_occupazione_memoria** Stampa il numero massimo di oggetti che potrebbero essere presenti contemporaneamente in memoria in base alla sequenza di operazioni. Ad esempio, per la sequenza di cui all'esempio al punto 2, stamperebbe 2.
6. **[6] controlla_sequenza** Controlla che la sequenza sia corretta, ossia non contenga nessuna delle seguenti tre operazioni illegali: accesso ad un oggetto non allocato, allocazione di un oggetto già allocato, deallocazione di un oggetto non allocato. Se la sequenza è corretta, stampa `Sequenza corretta`, altrimenti stampa `Sequenza non corretta`. Ad esempio, stampa sequenza corretta per la sequenza nell'esempio al punto 2, mentre stamperebbe sequenza non corretta per, ad esempio, le sequenze: `A 2 X 2 A 4 D 2 X 4 X 2`, oppure `A 2 X 2 A 4 D 2 X 4 D 2`, o infine `A 2 X 2 A 4 A 2`.

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 e 2. 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