Programmazione I
Prova di programmazione – 27 Luglio 2020 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma per l'assegnamento di docenti ad un insieme ordinato di corsi. L'insieme dei corsi è fissato, ed è: {*Programmazione I, Algoritmi e strutture dati, Architettura dei calcolatori*}. Non è richiesto di scrivere in programma in modo tale da poter gestire insiemi di corsi diversi da questo. Siccome l'insieme è ordinato, *Programmazione I* è il primo corso nell'insieme, *Algoritmi e strutture dati* il secondo corso ed *Architettura dei calcolatori* il terzo. Per ciascuno dei corsi, ciascun docente può esprimere una preferenza a tenere tale corso. Una preferenza è un numero intero tra 0 e 10. Ciascun docente è individuato dal suo cognome (una parola). L'insieme dei docenti, nonché le loro preferenze, sono definiti a tempo di esecuzione del programma. All'avvio del programma l'insieme dei docenti è vuoto. Il programma fornisce le seguenti funzionalità.

1. **inizializza_docenti(N)** Inizializza l'insieme dei docenti a contenere **N** docenti. L'eventuale precedente insieme è perso. Il valore massimo possibile per **N** non è noto a tempo di scrittura del programma. Per ciascun docente la funzionalità legge da *stdin* il suo cognome e tre numeri interi, intesi come le tre preferenze del docente. La prima preferenza è relativa al primo corso nell'insieme dei corsi, la seconda relativa al secondo, e al terza relativa al terzo. Se una sola preferenza non è compresa da 0 e 10, la funzionalità fallisce senza tentare di ottenere nuove preferenze.
2. **stampa_docenti** Stampa i nomi dei corsi, ed i nomi e le preferenze dei docenti. Utilizza il formato mostrao nel seguente esempio di stampa
   ```
   Corsi: Programmazione I, Algoritmi, Architettura
   Docenti e preferenze:
   Valente 1 5 7
   Marongiu 7 7 2
   Cavicchioli 2 8 6
   Montangero 10 1 9
   ```
3. **[2] salva_docenti** Salva l'insieme di docenti e preferenze in un file di testo dal nome predefinito.
4. **[3] carica_docenti** Carica l'insieme di docenti e preferenze dal file. L'eventuale precedente contenuto è perso.
5. **[6] assegna_docenti** Visita i corsi dell'insieme, seguendo l'ordinamento tra i corsi, e per ciascun corso visitato, assegna al corso il docente libero (ossia non ancora assegnato ad alcun corso) con la preferenza più alta per tale corso, se c'è ancora un docente libero. In particolare, mostra l'assegnamento stampando su *stdout* i nomi dei docenti assegnati ai corsi, nello stesso ordine dei corsi (primo docente assegnato al primo corso, secondo docente al secondo corso e così via). Ad esempio, per l'insieme utilizzato nell'esempio al punto 2, stamperebbe:
   ```
   Montangero Cavicchioli Valente
   ```
6. **[4] assegna_docenti2** Uguale alla funzionalità 5, ma con un controllo aggiuntivo: affinché l'assegnamento di un docente ad un corso abbia successo, tale docente non deve avere una preferenza superiore a nessun altro docente per nessuno dei corsi successivi a quello a cui verrebbe assegnato. Se l'assegnamento ha successo, allora l'output è identico alla funzionalità 5. Se l'insegnamento fallisce, allora l'output è
   ```
   Assegnamento fallito
   ```
   Ad esempio, l'assegnamento fallirebbe per l'insieme utilizzato nell'esempio al punto 2, perché Montangero sarebbe da assegnare a Programmazione I, ma ha una preferenza per Architetture superiore a tutti gli altri docenti. Se la preferenza di Montangero per Architettura dei calcolatori fosse stata ad esempio 6, allora l'assegnamento di tutti i docenti sarebbe stato possibile, ed uguale a quello dell'esempio per la funzionalità 5.

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*.

Per compatibilità col tester, configurare una lunghezza massima delle parole pari almeno a 20.

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