Programmazione I
Prova di programmazione – 17 luglio 2017 – 2 ore

Partendo dal frammento di codice fornito, realizzare un programma di gestione di compiti, da svolgere nell'arco di una giornata. Ciascun compito è caratterizzato da un *tempo*, ossia un numero naturale che rappresenta l'ora in cui tale compito va svolto, una *descrizione* ed un *luogo*, rappresentati semplicemente da una parola (stringa senza spazi) ciascuno. La lunghezza massima delle descrizioni e dei luoghi è definita a tempo di scrittura del programma. Si può memorizzare al più un compito per ciascuna ora della giornata. Chiamiamo *agenda* l'insieme dei compiti per la giornata. All'avvio del programma, l'agenda ha dimensione massima nulla, ossia non può contenere alcun compito. Il programma deve fornire le seguenti funzionalità.

1. **reinizializza_agenda(ora_min, ora_max)** Reinizializza l'agenda per contenere compiti i cui tempi vanno da **ora_min** ad **ora_max**. L'eventuale precedente contenuto dell'agenda è perso. Sia ora_min che ora_max devono essere compresi tra 8 e 23. Implementare questa funzionalità in maniera tale che l'agenda occupi **solo la quantità di memoria necessaria** per memorizzare un insieme di compiti i cui tempi vanno da **ora_min** ad **ora_max**.
2. **aggiungi_compito(t, d, l)** Aggiunge all'agenda un compito caratterizzato dal tempo **t**, dalla descrizione **d** e dal luogo **l**, a patto che il tempo **t** sia compatibile con i tempi memorizzabili nell'agenda, e che non vi sia ancora alcun compito memorizzato al tempo **t**.
3. **stampa_compiti** Stampa i compiti attualmente memorizzati nell'agenda, come sequenze di terne *tempo descrizione luogo*. Ad esempio:
   ```
   10 dentista modena
   14 meccanico carpi
   15 lavanderia carpi
   18 cinema modena
   ```
4. **salva_stato** Salva il contenuto dell'agenda in un file di testo dal nome definito a tempo di scrittura del programma.
5. **carica_stato** Carica il contenuto dell'agenda dal file. Il contenuto precedente è perso.
6. **aggiungi_compito2(t, d)** Aggiunge all'agenda un compito caratterizzato dal tempo **t** e dalla descrizione **d**, a patto che il tempo **t** sia compatibile con i tempi memorizzabili nell'agenda. In particolare, se c'è già un compito memorizzato a tale tempo **t**, controlla se è possibile ritardare il compito già memorizzato al tempo **t**, ed eventualmente altri compiti successivi già memorizzati, in maniera tale da fare posto al nuovo compito. Se questo è possibile, allora effettua tali spostamenti ed aggiunge il nuovo compito. Si ottiene il punteggio massimo se: i compiti spostati sono ritardati al più di un'ora, viene spostato solo il numero minimo indispensabile di compiti, si realizzano queste operazioni con un numero di passi lineare rispetto alla dimensione dell'agenda. Ad esempio, se si volesse aggiungere il compito *(14, piscina ,carpi)* all'agenda riportata nell'esempio al punto 3, si otterrebbe
   ```
   10 dentista modena
   14 piscina carpi
   15 meccanico carpi
   16 lavanderia carpi
   18 cinema modena
   ```

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