Esercitazione di laboratorio n. 5

(Caricamento sul portale entro le 23.59 del 02/12/2017 degli esercizi 1 e 3, 2 facoltativo)

Esercizio n. 1: Atleti

Competenze: Vettori di `struct`, strutture dati dinamiche, vettori dinamici, riallocazione dinamica (Puntatori e strutture dati dinamiche 2.5.5, 3.3.3, 3.2.3)

Sia dato un file di testo contenente l'anagrafica degli iscritti a una palestra (`atleti.txt`), organizzato come segue:

- sulla prima riga è presente un intero `N` rappresentante il numero di atleti
- sulle `N` righe successive sono presenti quattro stringhe quali un codice identificativo univoco, il nome e il cognome dell'atleta e la categoria sportiva in cui compete. Segue sulla stessa riga la data di nascita dell'atleta. La riga è terminata da un intero, positivo o nullo, a rappresentare il numero di ore settimanali di allenamento assegnate a ogni atleta
- il codice è nella forma `AXXXX`, dove `X` rappresenta una cifra nell'intervallo 0-9
- il nome, il cognome e la categoria di ogni atleta sono rappresentati da una stringa, priva di spazi, di massimo 25 caratteri alfabetici (maiuscoli o minuscoli)
- le date sono riportate nel formato gg/mm/aaaa (es: 08/11/1997)
- tutti i campi sono separati da uno o più spazi.

Si scriva un programma C tale per cui:

- i contenuti dell'anagrafica siano memorizzati in un vettore di strutture allocato dinamicamente della dimensione opportuna
- i dettagli di ogni atleta siano memorizzati in una apposita `struct` in cui tutte le stringhe reputate necessarie siano allocate dinamicamente.

Una volta memorizzate le informazioni contenute nel file, il programma rende disponibili le seguenti funzioni:

- stampa, a scelta se a video o su file, dei contenuti dell'anagrafica
- ordinamento del vettore per data di nascita ascendente
- ordinamento del vettore per codice atleta
- ordinamento del vettore per cognome (e nome in caso di omonimi)
- stampa a video degli atleti, divisi per categoria sportiva
- aggiornamento del monte ore settimanali
- ricerca di un atleta per codice
- ricerca di un prodotto per cognome (anche parziale).

Per quanto riguarda le ricerche, si richiede che siano implementate sia una funzione di ricerca dicotomica sia una funzione di ricerca lineare. Per quanto riguarda l'ordinamento, si presti attenzione alla stabilità dell'algoritmo prescelto nel caso di ordinamento per più chiavi successive a causa dell'omonimia. Si selezioni l'algoritmo di ricerca più opportuno: se la base dei dati è ordinata secondo la chiave di ricerca corrente si usi la ricerca dicotomica, altrimenti quella lineare. Si suggerisce a tal proposito di mantenere nel programma uno stato relativo all'ordinamento corrente della base dati (ossia, su quale chiave sia attualmente ordinato).

Esercizio n. 2: Analisi di puntatori
Competenze: tipo `struct`, matrici, definizione di puntatore
(Puntatori e strutture dati dinamiche 1.2, 1.3)

Si consideri il seguente tipo `struct`:

```
typedef struct item_s {
  int a, b;
  char c;
  float d;
  char s[MAXS];
} Item;
```

Si scriva un programma che utilizzi una matrice di Item: `M[N][N];`
`N` e `MAXS` siano definite mediante `#define`. Il contenuto della matrice viene acquisito da un file testo contenente `NxN` righe, ognuna delle quali contiene (con i campi separati da spazi) un Item: si consiglia una lettura formattata `fscanf(fp, "%d%d %c%f%s", …)`
Il programma acquisisca iterativamente da tastiera una coppia di interi `r`, `c` (compresi tra 0 e `N`-1). Visualizzi successivamente, per `M[r][c]`, il contenuto e l'indirizzo di tutti i campi della relativa `struct`.

Esercizio n. 3: Generatore di sigle
Tema d'esame del 02/02/2015 - Esercizio n°3 del percorso semplificato
Competenze: Problem-solving con modelli del Calcolo Combinatorio (Ricorsione e problem-solving 3.2, 3.3)

Una sigla alfanumerica di lunghezza `N` è composta selezionando per ognuna delle `N` posizioni `sigla[i]` un carattere che appartiene all'insieme `set[j]`. Ogni insieme `set[j]` è dato come stringa di caratteri alfanumerici e la sua cardinalità è al massimo 10. Un file contiene le informazioni relative alla lunghezza della sigla `N` (intero sulla prima riga) e alle stringhe che rappresentano i `set[j]` (`N` stringhe una per riga sulle `N` righe successive).

Si realizzi una funzione ricorsiva in C che, letto il file, generi tutte le possibili sigle e le memorizzi su di un secondo file. I nomi dei file siano passati come parametri alla funzione. Si realizzi un opportuno `main` per chiamare la funzione ricorsiva.

Esempio : se il primo file ha il seguente contenuto :
3
A
Xy
123
occorre scrivere nel secondo file le sigle: AX1, AX2, AX3, Ay1, Ay2, Ay3.

Suggerimento: si identifichi prima il modello del Calcolo Combinatorio, in seguito le strutture dati per le scelte e la soluzione, infine si scriva la funzione ricorsiva sulla base del modello.