Esercitazione di laboratorio n. 4

(Caricamento nella Sezione Elaborati del Portale entro e non oltre le 23:59 del 03/11/2015)

Esercizio n. 1: Anagrafica studenti

Sia dato un file di testo contenente l'anagrafica di un insieme di studenti, organizzato come segue:

- sulla prima riga è presente un intero N rappresentante il numero di studenti
- sulle N righe successive sono presenti quattro stringhe, rappresentanti la matricola, il nome, il cognome e la data di nascita dello studente, e un carattere (F o M) ad indicarne il sesso
- la matricola è nella forma sXXXXXX, dove X rappresenta una cifra nell'intervallo 0-9
- il nome e il cognome di ogni studente sono rappresentati da una stringa, priva di spazi, di massimo 35 caratteri alfabetici (maiuscoli o minuscoli)
- la data di nascita è riportata nel formato gg/mm/aaaa (es: 03/05/1993)
- tutti i campi sono separati da uno o più spazi.

Si scriva un programma che:

- acquisisca i contenuti in un vettore, di dimensione opportuna, allocato dinamicamente
- i dettagli di ogni studente 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 deve rendere disponibili le seguenti funzioni:

- ordinamento del vettore per matricola
- ordinamento del vettore per cognome, e successivamente per nome a fronte di cognomi uguali
- ordinamento del vettore per data di nascita (dal più giovane al meno giovane).

Esercizio n.2: Anagrafica studenti estesa

A partire dalle specifiche dell'esercizio n.1, estendere le funzionalità del programma per permettere:

- aggiunta di un nuovo studente alla base dati
- cancellazione di uno studente dalla base dati
- ricerca di uno studente per matricola
- ricerca di uno studente per cognome
- stampa a video della base dati

Per permettere di introdurre nuovi studenti, a fronte di vettore completamente occupato, si proceda a riallocare il vettore stesso in maniera opportuna.

Per quanto riguarda le ricerche, si richiede che siano implementate sia una funzione di ricerca dicotomica sia una funzione di ricerca lineare.
Il programma deve mantenere uno stato relativo all'ordinamento corrente della base dati (ossia, su quale chiave sia attualmente ordinato) e usare di conseguenza la funzione di ricerca più opportuna.

Esercizio n. 3: Matrici: allocazione e gestione

Un file di testo, il cui nome sia passato al programma da linea di comando, contiene le informazioni relative ad una matrice di interi. Sulla prima riga del file sono riportate le dimensioni R e C della matrice stessa. Sulle R righe successive, sono riportati i valori delle C colonne per ogni data riga. Si assuma che il formato del file sia corretto.

Si scriva un programma che allochi dinamicamente la matrice, effettui la lettura del file e infine calcoli la somma degli elementi delle diagonali e delle antidiagonali della matrice stessa.

Per quanto riguarda l'allocazione della matrice, si implementino due diverse funzioni. La prima faccia uso del valore di ritorno per restituire la matrice al `main`. La seconda faccia invece uso di parametri passati per riferimento. Nello specifico, si implementino due funzioni i cui prototipi siano nella forma:

```
int **alloca2d(...);
```

e

```
void alloca2d(int***, ...);
```