Esercizi di programmazione*

Lezione 17

Esercizio 1 Si scriva il codice LUNGHEZZA_RIGA.C, che riceve da linea di comando il nome di un file di testo e un numero intero `n` e stampa a video la lunghezza (numero di caratteri escluso l'a capo finale) della riga `n`-esima di tale file. Se il file contiene meno di `n` righe, si deve stampare 0.

Esercizio 2 Si scriva il codice APRIBILE.C, che riceve da linea di comando un numero qualsiasi di nomi di file di testo e stampa a video l'elenco di quelli che si possono aprire e di quelli che non si possono aprire. Il codice deve restituire al sistema operativo il valore `EXIT_SUCCESS` se tutti i file si possono aprire, `EXIT_FAILURE` se almeno uno dei file non si può aprire.

Esercizio 3 Si scriva il codice MAIUSCOLO.C, che riceve da linea di comando il nome di un file di testo e stampa a video il file stesso, convertendo tutte le lettere minuscole in maiuscole. Si considerino le due varianti nelle quali si legge una parola alla volta o una riga alla volta.

Esercizio 4 Si scriva il codice CONCATENA.C, che riceve da linea di comando i nomi di diversi file di testo esistenti e di un file da creare (l'ultimo) e salva nell'ultimo file la concatenazione del contenuto dei file iniziali, nell'ordine con cui compaiono nella linea di comando. Si usi un solo puntatore a file e l'apertura in accodamento.

Esercizio 5 Si scriva il codice LEGGEPROBLEMA.C, che riceve da linea di comando il nome di un file di testo, che contiene i dati per il seguente problema: una fabbrica produce n prodotti utilizzando m componenti. Ciascuna unità del prodotto j garantisce un profitto p_j, ma consuma una quantità a_{ij} del componente i. In magazzino sono presenti b_i unità del componente i. I dati n, m, p_j, a_{ij} e b_i sono rappresentati secondo le specifiche del linguaggio di modellazione *MathProg*. Un esempio di tale linguaggio è riportato qui di seguito:

```
param NumProdotti := 3 ;
param NumComponenti := 4 ;

param Profitto :=
1 50
2 30
3 30
;
```

*tratti o ispirati dal testo di K.N. King

```
param Consumo :=
[1,1] 2 [1,2] 1 [1,3] 1
[2,1] 1 [2,2] 0 [2,3] 1
[3,1] 0 [3,2] 1 [3,3] 1
[4,1] 1 [4,2] 2 [4,3] 1
;

param Magazzino :=
1 1000
2 400
3 700
4 1200
;

end;
```

Il codice da scrivere deve caricare i valori di n e m in opportune variabili, quelli di p_j e b_i in opportuni vettori e quelli di a_{ij} in un'opportuna matrice. Vettori e matrice devono essere allocati dinamicamente. Quindi il programma deve stampare i dati a video.

Suggerimento: Si veda anche la dispensa `Leggere.pdf` sul sito del corso.