Esercizi 21-11-09

Riccardo Gennaro

October 2021

Esercizio 1

Scrivere la funzione

```
1    void outerProduct3(float p[][3], float a[], float b[]);
```

che ritorna il *prodotto esterno* dei primi tre elementi di **a** e **b**. Tale *prodotto esterno* è definito come la matrice p[3][3] i cui elementi $p_(ij)$ sono tali che

$$p_{ij} = a_i \cdot b_j$$

Esercizio 2

Scrivere un programma che, dato un file di input contente delle parole, generi un secondo file in output che contenga le stesse parole, ma in ordine inverso rispetto al file iniziale. Per essere copiata, una parola deve essere composta da un numero di caratteri **pari**.

Il programma dovrà accettare due parametri da riga di comando: il nome del file in input e il nome del file su cui effettuare l'output, in questo ordine.

Il programma dovrà anche implementare dei controlli sul numero di argomenti passati da riga di comando e sull'apertura dei file (in caso di un file di input non esistente).

Procediamo ora con un esempio. Dato in input il file input_A, contenente le seguenti parole:

```
Impara a risolvere tutti i problemi che sono stati risolti.  Richard
Feynman
```

se l'eseguibile è a.out, allora il comando `./a.out input_A output` genererà un file chiamato `output` che conterrà le seguenti parole:

```
risolti.  sono problemi Impara
```

Note

- Per semplicità, considerate come una *parola* qualsiasi stringa di caratteri compresa tra due spazi bianchi (e/o separatori di tabulazione, nuova linea e fine file). Quindi, sono da considerarsi parole anche stringhe come `"andato:"` o `"!esempio"`.
- Il file in input può contenere al massimo 10000 parole e ognuna di queste parole è formata da al massimo 100 caratteri.
- Non è consentito l'utilizzo di librerie diverse da `<iostream>` ed `<fstream>` pena **l'annullamento dell'esercizio**.
- E' consentito definire ed implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema.

Esercizio 3

Scrivere nel file esercizio3.cc:

- una funzione **ricorsiva** *calcola_norma_ricorsivo* che, dato un array di long e la sua dimensione, restituisca un numero *double* corrispondente alla **norma ad uno** dell'array stesso;
- una funzione **ricorsiva** *normalizza* che, dato un array di *long* e la sua dimensione, restituisca un **nuovo array** contenente, per ogni elemento, l'elemento corrispondente del primo array diviso per la **norma ad uno** dell'array in input, calcolata con la funzione di cui al punto precedente.

Le operazioni di calcolo della norma e di divisione degli elementi dell'array per il valore della norma vanno eseguite tramite funzioni ricorsive, **pena l'annullamento dell'esercizio**.

Non è ammesso l'uso di cicli, variabili globali o static. È ammesso l'uso di funzioni ausiliarie, purché ricorsive. È consentito l'uso della funzione `sqrt` della libreria `cmath`.

E vietata ogni modifica alla funzione main pena l'annullamento dell'esercizio.

NOTA 1: La norma ad uno di un vettore è la somma dei degli elementi del vettore.

$$||x|| := \sum_{i=1}^{n} x_i$$

Ad esempio:

- Array in input: [1, 2, 4]
- Norma = 1 + 2 + 4 = 7
- Array normalizzato = [1/7, 2/7, 4/7] = [0.142857, 0.285714, 0.571428]

NOTA 2: I valori dell'array iniziale vanno inseriti utilizzando la funzione `leggi(...)` definita in `array.h` e `array.o`.
La stampa dell'array normalizzato va eseguita usando la funzione `stampa(...)` definita in `array.h` e `array.o`.

Esercizio 4

Scrivere un programma che calcoli il *prodotto matriciale* di due matrici di numeri interi. Se le matrici non sono compatibili per il prodotto stampare un messaggio di errore.
La dimensione delle matrici deve essere specificata dall'utente. Il prodotto è calcolato con la funzione seguente:

```
1    int** compute_prod(int** m1, int r1, int c1, int** m2, int c2);
```

Dove

- `m1` è la prima matrice;
- `r1` è il numero di righe di m1;
- `c1` è il numero di colonne di m1;
- `m2` è la seconda matrice;
- `c2` è il numero di colonne di m2;

NOTA: la complessità di `compute_prod(...)` deve essere pari o inferiore a $O(n^3)$. Ciò significa che potete implementare la funzione con l'algoritmo che si basa sulla definizione del prodotto fra matrici, oppure, se siete dei pazzi maniaci, potete implementare l'algoritmo di Strassen.