INFORMATICA – a.a. 2016/17

Esercitazione di Laboratorio 12

Obiettivi dell'esercitazione

- Progettare base dati e algoritmi per la risoluzione di (piccoli) problemi

Contenuti tecnici

- Verifica dell'apprendimento e del pieno possesso dei contenuti del corso

---

Da risolvere preferibilmente in laboratorio

Esercizio 1. Si scriva un programma che legga da file (nome è ricevuto da linea di comando) un elenco di parole (max 1000) e generi una statistica. Tale statistica deve riportare per ciascuna parola (massimo 15 lettere) il numero di occorrenze e si deve basare su una struttura dati opportuna.

Esempio:

ecco un possibile file in input

Torino Lecce Bari Roma Terni Torino Roma Napoli
Caserta Firenze Reggio Catania
Firenze Bari Lecce
Milano Verona Milano Torino

e la statistica che ne deriva

Terni 1
Napoli 1
Caserta 1
Reggio 1
Catania 1
Verona 1
Lecce 2
Bari 2
Roma 2
Firenze 2
Milano 2
Torino 3

Esercizio 2. È una variante dell'esercizio 1. Il file contiene un generico testo (non formattato), compresi i caratteri di interpunzione. Un secondo file, *particel.dat*, contiene, una per riga, una serie di nomi (tipicamente particelle come *di*, *a*, *da*…) che sono da escludere dalla statistica. Il numero di particelle non supera i 100. I nomi del testo possono contenere caratteri maiuscolo e minuscolo, ma per la statistica devono essere riportati tutti a minuscolo.

Da risolvere a casa

3) Si realizzi un programma in C che sia in grado di gestire il calcolo automatico di una partita al Fantacalcio. Il programma deve ricevere in input (da riga di comando) due file, di uguale formato, contenenti le formazioni sfidanti. Ciascun file è composto da più righe, una per ogni giocatore schierato, così composte:
<nome_giocatore> <ruolo>
Si supponga che il nome del giocatore sia lungo al massimo 50 caratteri e che possa contenere spazi, mentre il ruolo sia un singolo carattere: 'P' per portiere, 'D' per difensore, 'C' per centrocampista ed 'A' per attaccante. Si supponga inoltre che il numero massimo di riserve sia pari a 7 e che ogni giocatore abbia compilato correttamente la rispettiva formazione (nessun doppione, da 11 a 18 giocatori in tutto, ecc).

Il programma deve calcolare il punteggio totale di ogni squadra secondo la seguente modalità operativa:
1. per ogni giocatore schierato, si chieda, a schermo, di inserire il punteggio del giocatore (calcolato secondo le peculiari regole del torneo), prevendo la possibilità che sia 'X' quando il giocatore non abbia giocato.
2. Ogni qual volta sia necessario saltare un giocatore, sia sostituito da un giocatore di pari ruolo tra le riserve, fino ad un massimo di 3 sostituzioni. Qualora non fosse possibile trovare un giocatore di pari ruolo, si riduca il numero di giocatori schierati di una unità.
3. Si sommi il totale di ogni squadra e lo si visualizzi nel modo più piacevole possibile.

Una volta realizzata la versione di base, si provi a implementare una nuova versione del programma che consenta:

- di gestire il risultato finale di una partita analizzando i totale delle due squadre ed applicando delle regole per permutare il punteggio in "goal".

- di gestire le sostituzioni "fuori ruolo": per ogni giocatore di cui non sia rintracciabile una riserva di pari ruolo, si selezioni comunque un giocatore decurtandone il punteggio totale in funzione di quanto sia "fuori luogo". Esempio: P al posto di A potrebbe corrispondere a -5 mentre un C al posto di un D ad un -2. Se particolarmente ispirati, si dia la possibilità di configurare tali punteggi all'interno di un terzo file.

- di gestire un intero campionato, salvando un file con la classifica che si aggiorni automaticamente dopo ogni partita analizzata.

Nota: gli esercizi che seguono non richiedono necessariamente l'uso di file, sono esercizi sugli algoritmi.

4) Realizzare un programma in C che, lette da tastiera le dimensioni di un rettangolo n x m, simuli il movimento di due robot. I due robot partono da due angoli opposti del rettangolo ed ognuno si muove di un passo per volta. Il primo compie un percorso a spirale in senso orario, toccando tutte le celle fino al centro; il secondo si muove percorrendo ad una ad una tutte le colonne, salendo e scendendo alternativamente. L'elaborazione termina quando i due robot arrivano ad occupare la stessa cella, o quando hanno compiuto il loro percorso; prima di terminare il programma deve stampare quale è stata la condizione di uscita.
Suggerimento: la base dati è rappresentata dalla posizione dei robot (riga e colonna), il rettangolo su cui si muovono può essere "virtuale".

5) Realizzare un programma di ausilio al gioco del 15 (se non lo si conosce, acculturarsi in Internet). All'inizio il programma deve disporre in modo casuale, in una matrice 4 x 4, tutti numeri tra 0 e 15. Il numero 0 verrà trattato e visualizzato come spazio vuoto (blank). Dopo la visualizzazione della matrice al centro dello schermo (in modo formattato), il programma attende che il giocatore segnali tramite tastiera qual è la tessera (numero) da spostare. Il programma deve verificare che la mossa sia valida (cioè che in una posizione contigua a quella indicata ci sia lo spazio vuoto, cioè lo zero), e in caso affermativo deve attuare la mossa e tornare alla visualizzazione della matrice. In caso negativo deve segnalare l'errore e tornare alla visualizzazione. Il programma

deve inoltre verificare ad ogni mossa se si è raggiunta la configurazione vincente, quella in cui i valori sono crescenti dall'alto in basso e da sinistra a destra, con lo spazio vuoto nell'angolo in basso a destra. Il gioco termina in caso di vittoria.
Si consiglia di parametrizzare la dimensione della matrice (quadrata), in modo da poter giocare con una matrice 3 x 3 per test.

Configurazione iniziale (esempio):

```
-------------
| 7|10|14|  |
-------------
| 5|12|11| 8|
-------------
| 3| 9| 1| 4|
-------------
|13| 6|15| 2|
-------------
```

Configurazione finale (vincente):

```
-------------
| 1| 2| 3| 4|
-------------
| 5| 6| 7| 8|
-------------
| 9|10|11|12|
-------------
|13|14|15|  |
-------------
```

6) Una possibile tecnica per gestire numeri interi di valore anche molto elevato consiste nell'utilizzare dei vettori di interi. In tale caso ogni cifra decimale del numero viene memorizzata in un diverso elemento del vettore. Ad esempio è possibile memorizzare la cifra corrispondente alle unità (cioè la meno significativa) nell'elemento di indice 0, la cifra corrispondente alle decine nell'elemento di indice 1 e così via.

Si realizzi un programma in linguaggio C che, utilizzando il metodo di rappresentazione indicato, sia in grado di:

- leggere da tastiera due numeri interi di dimensione N
- leggere un carattere indicante l'operazione da effettuare sui numeri (+ = somma, - = differenza, * = mltiplicazione)
- effettuare l'operazione indicata esprimendo il risultato nello stesso formato.

Si noti quanto segue:

- la costante N risulta essere predefinita (i.e., direttiva define)
- si gestiscono solo numeri positivi (non esiste simbolo per rappresentare il segno dei numeri)
- occorre indicare l'eventuale presenza di un overflow se il risultato non e` rappresentabile su un vettore di dimensione N.

7) Tra gli scherzosi linguaggi codificati della fanciullezza, ricorre spesso

in America il "piglatin". Si prende una parola inglese e se ne traspone il primo suono (di solito la prima lettera) in fine parola, aggiungendo poi la lettera 'a'. Cosi' "dog" diventa "ogda", "computer" diventa "omputerca", "piglatin" diventa "iglatinpa" e "pig latin" diventa "ipga atinla", ecc.
Realizzare un programma che accetti da tastiera una riga di testo in inglese e ne stampi il corrispondente piglatin. Ogni testo puo' essere battuto su una riga di al piu' 80 caratteri, con uno o piu' blank o altri separatori (virgola, punto, punto e virgola, ecc.) tra una parola e l'altra, che devono essere mantenuti nel testo in stampa. I caratteri si intendono tutti minuscoli.
Si consiglia di procedere così:
- leggere la riga da tastiera in un vettore di caratteri
- utilizzando un opportuno algoritmo, individuare la posizione dell'inizio e della fine di ciascuna parola
- spostare la prima lettera della parola al fondo della parola stessa (non deve cambiare il numero complessivo di lettere)
- inserire subito dopo la parola la lettera 'a' (il numero complessivo di lettere è incrementato di uno)
- stampare il vettore di caratteri modificato.

8) In molte applicazioni industriali si utilizzano i codici a barre per rappresentare le cifre. Ogni cifra è codificata mediante un pacchetto di bit, e successivamente gli 0 sono stampati come barre strette, gli 1 come barre larghe.
Realizzare un programma in C che generi tutti i codici 2 su 5 (codici di 5 bit, di cui 2 bit ad uno, gli altri a zero) e li memorizzi in un'opportuna base dati. Si rammenti che i codici binari possono essere visti come numeri interi senza segno, per cui è sufficiente generare tutti i codici di 5 bit (sono 32, da 0 a 31) e poi scartare quelli che non soddisfano la condizione di avere 2 bit ad uno e 3 a zero.
Successivamente il programma confronti i codici a due a due e, per ciascuna coppia, confronti i bit a parità di posizione, conteggiando il numero di bit diversi tra loro (il numero di bit diversi viene detto "distanza" tra i codici). Il programma calcoli e stampi il minimo della distanza tra i codici.