TECNICHE DI PROGRAMMAZIONE, A.A. 2022/2023

Esercitazione di Laboratorio 2

Valutazione: **gli esercizi 1 e 3** saranno oggetto di valutazione.
Scadenza: caricamento di quanto valutato - entro le 23:59 dell'4/4/2023 (insieme a lab01 e lab03).
Le modalità di caricamento sul portale della didattica, sezione "elaborati", sono descritte nel documento "istruzioni per consegna".

Obiettivi

- Risolvere problemi gestendo problemi iterativi su dati scalari (*Dal problema al programma: Cap. 3*), di tipo numerico (3.1) e di codifica (3.2)

Contenuti tecnici

- Basi di Input Output
- Utilizzo di funzioni
- Costrutti condizionali e iterativi
- Manipolazioni elementari di numeri (int e float) e caratteri (char)

ATTENZIONE
Gli esercizi 3 e 4 sono relativi a problemi di "codifica di testi", affrontati come primo argomento nella lezione di giovedì 17/3 (Capitolo 3 - Problem solving dati scalari – slide pag. 39-43: Crittografia semplice). I problemi dovrebbero essere ampiamente alla portata di tutti/e. Si consiglia tuttavia di leggere le poche slide sopra citate prima di affrontare gli esercizi stessi.

Di questi esercizi si forniscono esempi di file di ingresso. Ciò non esclude per nulla la possibilità di creare propri file. Deve essere poi chiaro che il programma realizzato dovrebbe agire correttamente con qualunque file conforme al testo del problema (quindi non limitarsi a "funzionare" correttamente solo sugli esempi forniti): questo, ovviamente, nei limiti del buon senso, senza pretendere che si faccia un test esaustivo (irrealizzabile in pratica) su tutti i possibili file di ingresso.

Da risolvere durante il laboratorio oppure prima/dopo il laboratorio stesso

Esercizio 1.
Competenze: IO formattato, manipolazioni di numeri;
Categoria: problemi numerici (Dal problema al programma: 3.1), con affinità ai problemi di codifica di numeri (3.2.1)

Scrivere un programma C che, acquisiti 2 numeri interi positivi ne calcoli il massimo comune divisore utilizzando la formula di Eulero.

Formula di Eulero o metodo dei resti: si procede per divisioni successive del numero maggiore per quello minore, sostituendo ad ogni passo il valore maggiore con il minore ed il minore col resto della divisione. Il processo termina quando il resto è 0.

Esempio: A = 34 , B = 18

passo 1: 34 % 18 = 16
passo 2: 18 % 16 = 2

passo 3: 16 % 2 = 0 ← stop!

Risultato: MCD = 2

Esercizio 2.
Competenze: IO formattato, manipolazioni di numeri;
Categoria: problemi numerici (Dal problema al programma: 3.1)

Scrivere un programma in linguaggio C che visualizzi i primi N numeri della serie di Fibonacci, con N acquisito in input da tastiera.

Suggerimento: ecco i primi numeri appartenenti alla serie 0 1 1 2 3 5 8 … In modo formale la serie si costruisce considerando la seguente relazione: $X_i = X_{i-1} + X_{i-2}$, con $X_0 = 0$ e $X_1 = 1$;

Approfondimento (variante dell'esercizio): si modifichi la serie come segue:

$X_i = X_{i-1} * X_{i-2}$, con $X_0 = 1$ e $X_1 = 2$;

Si determini sperimentalmente (osservando i risultati ottenuti) quanti sono gli elementi di questa serie rappresentabili con variabili di tipo intero (int) e di tipo intero senza segno (unsigned int)

Esercizio 3.
Competenze: IO su file, manipolazioni di caratteri;
Categoria: problemi di codifica applicati a testi/caratteri (3.2.2)

Un file (`sorgente.txt`) contiene un testo composto da un numero indefinito di righe.
Notare che il testo NON contiene il carattere '$'.

Lo scopo del programma è di ricodificare il testo sostituendo sequenze di caratteri ripetuti da 2 a 9 volte (ATTENZIONE: il numero non comprende il primo carattere, cioè AA contiene una ripetizione, BBB due ripetizioni, ecc.) con la terna di caratteri <carattere ripetuto>$<numero di ripetizioni> (ATTENZIONE: per il <numero di ripetizioni> è sufficiente un carattere). Se un carattere fosse ripetuto più di 9 volte, le ripetizioni sarebbero spezzate in più intervalli.

Ad esempio, la sequenza di caratteri "il numero 1000000 e' grande", viene compressa in "il numero 10$5 è grande". La sequenza, "ci sono 15 = ripetuti: ========== e 4 punti…." viene ricodificata in: "ci sono 15 = ripetuti: =$9=$4 e 4 punti.$3"

Il risultato della ricodifica sia salvato su un secondo file (`compresso.txt`).

Esempio:
Il contenuto del file `sorgente.txt` è:

```
 Partenza      Destinazione    Costo
   Parigi          New York     1000
     Roma            Londra      700
   Sidney       Los Angeles     2222
```

Il file di uscita `ricodificato.txt` conterrà:

```
Partenza $5Destinazione $3Costo
$2Parigi $9New York $410$2
$4Roma $9  Londra $5700
$2Sidney $6Los Angeles $42$3
```

Si scrivano due funzioni, in grado di effettuare, rispettivamente, la compressione e la decompressione. I prototipi delle funzioni siano
int comprimi(FILE *fin, FILE *fout);
int decomprimi(FILE *fin, FILE *fout);

In caso di errore le funzioni ritornano come risultato 0, diversamente ritornano il numero di caratteri scritto nel file in uscita. Il main permette all'utente di selezionare (in base a un opportune input) codifica oppure decodifica, apre i file quello in input e quello in output. chiama la funzione selezionata. Si consiglia di usare un terzo file per il testo de-compresso (evitando così di sovrascrivere, perdendolo, il file originale `sorgente.txt`)

Esercizio 4.
Competenze: IO su file, manipolazioni di caratteri;
Categoria: problemi di codifica applicati a testi/caratteri (3.2.2)

Un file contiene un testo composto da un numero non noto di caratteri.

Lo scopo del programma è di ricodificare il testo, generando un secondo file, nel quale i caratteri sono stati ricodificati secondo le regole seguenti:

I caratteri numerici sono ricodificati in questo modo:

- I caratteri numerici ('0'..'9') sono ricodificati nel carattere numerico posto k posizioni più avanti, con k che, partendo da 0, viene incrementato ad ogni carattere numerico ricodificato (ATTENZIONE: gli incrementi sono effettuati MODULO 10, cioè, arrivati a 9 si riparte da 0). Se ad esempio il file iniziasse con la riga: "Il numero 248 è pari", il '2' verrebbe ricodificato (k parte da 0) in '2'+0 = '2' (k diventa 1), il '4' in '4'+1 = '5' (k diventa 2) e '8' viene ricodificato in '8'+2 = '0' (perché arrivati a '9' si riparte da '0').
  I caratteri alfabetici sono invece ricodificati in questo modo:
- Se un carattere alfabetico è preceduto da un carattere non alfabetico, resta inalterato
- Se è preceduto da un carattere alfabetico (sia c0 il carattere precedente), il suo codice si sposta di h posizioni in avanti, nell'insieme dei caratteri alfabetici (con h = c0-'A', se c0 è maiuscolo, h=c0-'a', se c**0** è minuscolo. L'incremento di h è modulo 26, cioè arrivati alla 'z' o 'Z' (a seconda che il carattere ricodificato sia maiuscolo o minuscolo) si riparte da 'a' o 'A'.

Il risultato della ricodifica sia salvato su un secondo file (I nomi dei file siano opportunamente acquisiti da tastiera).

Esempio:
Se il contenuto del file è:

```
Apelle figlio di Apollo
fece una palla di pelle di pollo
tutti i pesci vennero a galla
per vedere la palla di pelle di pollo
```

```
fatta da Apelle figlio di Apollo.
```

Il file di uscita `ricodificato.txt` conterrà:

```
Aptept fntema dl Apdozn
fjlp uhh ppall dl ptept dl pdozn
tngzh i ptlnv vzmzdui a ggrcc
ptk vzcgxb ll ppall dl ptept dl pdozn
ffyrr dd Aptept fntema dl Apdozn.
```

Si scrivano due funzioni, in grado di effettuare, rispettivamente, la codifica e la decodifica.
I prototipi delle funzioni siano:

```
int codifica(FILE *fin, FILE *fout);
int decodifica(FILE *fin, FILE *fout);
```

In caso di errore le funzioni ritornano come risultato 0, diversamente ritornano il numero di caratteri scritto nel file in uscita. Il main permette all'utente di selezionare (in base a un opportuno input) codifica oppure decodifica, apre i file quello in input e quello in output. chiama la funzione selezionata.