CORSO DI FONDAMENTI DI INFORMATICA

Prof. Maristella Matera - A.A. 2022 / 2023

Laboratorio di Programmazione in C – Laboratorio 4

Esercizio 1 – Gioco di prestigio

Il programma definisce la struttura **Carta** che rappresenta una carta da gioco. La struttura è costituita dal valore della carta (singolo carattere) e dal seme (singolo carattere). Se il valore della carta è una lettera, questa deve essere maiuscola (per semplicità, rappresentare il 10 come D). I semi delle carte sono indicati con la notazione italiana (C, Q, F, P).

Il programma esegue un gioco di prestigio all'utente: estrae casualmente 21 carte da un mazzo di 52 carte da gioco. Le mostra a video e chiede all'utente di pensare a una delle carte. Successivamente le divide in tre mazzetti sullo schermo e chiede all'utente di indicare in quale di questi è presente la carta pensata. Questo procedimento viene ripetuto tre volte in totale e al termine il programma mostra a video la carta pensata dall'utente.

Il trucco: partendo dalla prima carta in cima al mazzo, i mazzetti sono ottenuti disponendo una carta alla volta da sinistra a destra su tre colonne. A ogni iterazione, il mazzetto contenente la carta scelta deve essere posizionato in mezzo agli altri due, ottenendo un nuovo mazzo. Al termine delle tre iterazioni, la carta scelta occuperà l'undicesima posizione nel mazzo.

Esempio: estraggo 21 carte dal mazzo:

DP QF 4P 6P 2C KP QP 9C QQ 7C KQ 3P 6Q 8F 9F 3C KF AQ 3F DF 5Q

Lo spettatore sceglie l'**asso di quadri** (AQ):

Prima iterazione: **AQ finisce in colonna 3**	**Seconda iterazione:** **AQ finisce in colonna 1**	**Terza iterazione:** **AQ finisce in colonna 3**
DP QF 4P 6P 2C KP QP 9C QQ 7C KQ 3P 6Q 8F 9F 3C KF AQ 3F DF 5Q	DP 6P QP 7C 6Q 3C 3F 4P KP QQ 3P 9F AQ 5Q QF 2C 9C KQ 8F KF DF	6P 6Q 4P 3P 5Q 9C KF DP 7C 3F QQ AQ 2C 8F QP 3C KP 9F QF KQ DF
Mazzo finale: AQ finisce in posizione 11 6P 3P KF 3F 2C 3C QF 4P 9C 7C AQ QP 9F DF 6Q 5Q DP QQ 8F KP KQ		

Il programma, implementando opportune funzioni, deve:

- aprire il mazzo di 52 carte da gioco contenuto nel file binario **mazzoPolimi.bin** fornito con questo testo e memorizzarlo in una struttura adeguata
- estrarre a caso 21 carte dal mazzo inserendole in un array di carte
- salvare il mazzo di carte ottenuto in un file binario chiamato **mazzoStudente.bin**
- eseguire il gioco di prestigio all'utente secondo il procedimento riportato sopra

Suggerimento: *per la generazione di numeri pseudocasuali è possibile utilizzare la funzione* ***rand()*** *unitamente alla funzione* ***srand().*** *Entrambe le funzioni sono contenute in* ***stdlib.h.***

La funzione ***srand()*** *prende come unico parametro un numero intero che fa da "seme" per l'algoritmo di generazione di numeri pseudocasuali (come seme può essere utilizzato il numero di secondi trascorsi dal 1 gennaio 1970, restituito dalla funzione* ***time()*** *contenuta in* ***time.h****) e va eseguita prima della chiamata a* ***rand()****.*

Esercizio 2 - Libreria Matematica

Creare una libreria matematica contenente le funzioni per il calcolo della potenza e del massimo comune divisore.

Scrivere un `main()` che, ricevuti due valori interi da riga di comando (usando `argc` e `argv[]`), richiami entrambe le funzioni e stampi il risultato.

Esercizio 3 – Copia dispari v. 2.0

Modificare il programma dello scorso laboratorio **utilizzando le funzioni** e in modo che la matrice `Mat1` di interi `NxN` (sempre con `N` costante opportunamente definita) **venga letta da un file di testo chiamato 'matrice.txt' scritto in formato CSV (*Comma Separated Values*):**

```
1,2,3
4,5,6
7,8,9
```

La funzione copia i soli elementi dispari in una nuova matrice `Mat2` della stessa dimensione, senza lasciare posizioni vuote intermedie.

Il programma, prima di procedere con la copia degli elementi, **verifica che la matrice definita nel file abbia le dimensioni richieste.** Se la matrice non è di dimensioni `NxN` allora deve comparire un messaggio di errore a video e il programma viene terminato.

Implementare un `main()` che, dopo aver richiamato la funzione implementata, **scriva in un file di testo 'matrice_dispari.txt' la nuova matrice Mat2,** rispettando il formato riportato sopra.

Esercizio 4 – Sudoku

Si realizzi un programma che verifichi la corretta soluzione della griglia di un Sudoku.

Un Sudoku è una griglia 9x9 che rispetta queste caratteristiche:

- tutti i valori devono essere singoli numeri tra 1 e 9
- su ciascuna delle 9 **righe** non devono esserci valori ripetuti
- su ciascuna delle 9 **colonne** non devono esserci valori ripetuti
- in ciascuno dei 9 **blocchi 3x3** che compongono la griglia non devono esserci valori ripetuti

L'esercizio è da risolvere svolgendo un livello dopo l'altro, secondo questa sequenza:

Livello 1 – Utilizzare il file `sudoku_template.c` fornito assieme a questo testo come punto di partenza: il file contiene la dichiarazione di due matrici Sudoku da verificare (una corretta e l'altra sbagliata). Il programma deve verificare la validità di queste due matrici e stampare a video il risultato del controllo. In questo livello il programma va scritto **usando le funzioni** (quindi completando solo il `main()` già esistente) e dividendo il problema assegnato in sottoproblemi.

Livello 2 – Modificare il programma già scritto in modo che la matrice da verificare venga acquisita **numero per numero,** e che venga eseguito il controllo di validità **già in fase di inserimento.**

Livello 3 – Modificare il programma già scritto in modo che la matrice da verificare venga letta dal file di testo `sudoku.txt` fornito assieme a questo testo.