FONDAMENTI DI INFORMATICA

Prof. Maristella Matera - A.A. 2024/2025

Laboratorio di Programmazione in C

Laboratorio 5

Es. 1 Generatore di frasi per biscotti della fortuna

■ *Liste, allocazione dinamica, gestione dei file*

La *Golden Gate Fortune Cookie Factory* di San Francisco esporta biscotti della fortuna in tutto il mondo dal 1962: dopo tanto tempo, in fabbrica non sanno più quali frasi inserire nei loro biscotti.

Scrivere un programma C che generi casualmente delle frasi a partire dal contenuto di alcuni file di testo `.txt` forniti con il testo dell'esercizio:

- `momenti.txt` (es. "stasera", "domattina", etc...)
- `azioni.txt` (es. "troverai", "porterai", etc...)
- `cose.txt` (es. "un libro", "l'amore", etc...)
- `luoghi.txt` (es. "in città", "al bar", etc...)

Ogni file è composto da più righe, e ciascuna riga può contenere spazi.

Per la generazione della frase è richiesto che da ciascun file venga estratta casualmente una riga secondo l'ordine `momenti–azioni–cose–luoghi`.

All'avvio del programma, il contenuto di ciascun file va inserito all'interno di una lista dinamica dedicata. Il singolo elemento della lista è definito tramite uno struct di tipo `Nodo` contenente la riga letta dal file e il puntatore all'elemento successivo.

Il programma all'avvio deve chiedere all'utente quante frasi ha bisogno di generare.

Le frasi generate vanno salvate man mano in una lista dinamica. La lista va stampata a video e poi salvata all'interno di un file di testo chiamato `frasi.txt`.

Suggerimenti:

- Potete utilizzare la funzione `snprintf()` definita in `stdio.h` per stampare una stringa in un buffer rispettando una certa format string e una lunghezza di massimo `n` caratteri (utile per comporre e salvare la frase generata all'interno di una stringa).

Es. 2 Il filo di Arianna

■ *Ricorsione, gestione dei file, matrici*

All'interno di un file di testo in formato CSV (Comma-Separated Values) e chiamato `labirinto.csv`, definire una matrice quadrata `NxN` composta esclusivamente da `1` e `0` che rappresenti un labirinto (dove `1` = muro, `0` = percorso).

Esempio di labirinto in formato CSV:

```
0,1,0,0,1
0,0,1,0,1
1,0,1,0,0
1,0,0,1,0
1,1,0,0,0
```

Scrivere un programma C che:

- legge il file `labirinto.csv`
- controlla che la matrice sia formalmente corretta secondo le specifiche riportate in precedenza
- verifica attraverso una **funzione ricorsiva** se esiste un percorso per uscire dal labirinto partendo dalla posizione `(0,0)` (ingresso) fino alla posizione `(n-1, n-1)` (uscita).

Es. 3 "I topi non avevano nipoti"

■ *Ricorsione, stringhe*

Scrivere un programma C che, attraverso una **funzione ricorsiva**, verifichi se una stringa è un palindromo. Un palindromo è qualsiasi verso, frase, parola o cifra che letta in senso inverso mantiene immutato il significato.

La funzione deve confrontare i caratteri iniziali e finali della stringa riducendola ad ogni chiamata. La stringa può contenere spazi bianchi, la funzione non deve essere case-sensitive e deve ignorare tutto ciò che non è alfanumerico (punteggiatura, spazi...).

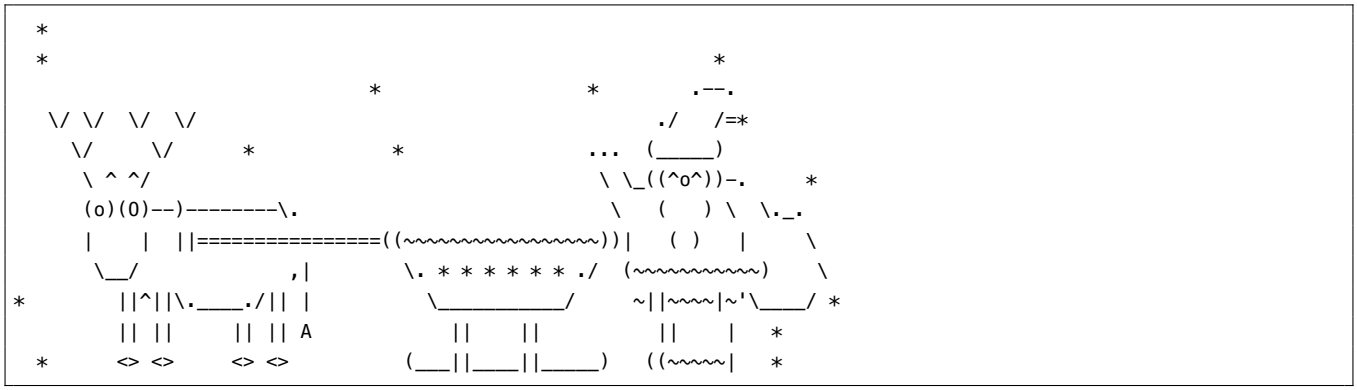
Esempio:

Le stringhe **Ingegno c'era nell'allenare congegni** e **Erede, vai a vedere!** sono palindromi.

Le stringhe **Non ricordo se c'ero e se c'ero dormivo** e **This is not a bug, it's a feature** non sono palindromi.

Es. 4 La slitta di Babbo Natale

■ *Liste, gestione dei file, allocazione dinamica, ricorsione, funzioni, programmazione modulare, struct*



Babbo Natale ha ricevuto l'elenco dei regali da consegnare la notte del 25 dicembre: la lista è stata compilata dai suoi elfi unendo il contenuto delle letterine spedite dai bambini di tutto il mondo. I regali richiesti sono molti, e quindi ha bisogno di voi per ottimizzare il caricamento dei regali sulla sua slitta.

Gli elfi aiutanti di Babbo Natale sono notoriamente molto disordinati: la loro lista potrebbe contenere elementi ripetuti e non nell'ordine richiesto da Babbo Natale.

L'elenco ricevuto da Babbo Natale è contenuto in un file di testo in formato `.txt`, composto da più righe e fornito con il testo dell'esercizio. Ciascuna riga contiene in sequenza il nome del regalo, il peso del singolo regalo in grammi e la quantità da caricare sulla slitta. Tutti i valori sono separati da spazi, per esempio:

```
Trenino 300 5
Slime 100 12
Drone 500 23
Bambola 150 34
Trenino 300 23
Trenino 300 5
Bambola 150 34
Koala 350 17
```

Svolgere i seguenti passaggi:

1. Definire il tipo `Regalo` che contiene:
   (a) il nome del regalo desiderato (`nome`), di lunghezza massima 50 caratteri e senza spazi;
   (b) un numero intero che indica il peso in grammi del singolo regalo (`peso`);
   (c) un numero intero che indica la quantità (`qty`);
   (d) il puntatore di tipo `Regalo` all'elemento successivo (`next`);
2. Definire il tipo `ListaRegali` che indica una lista composta da elementi di tipo `Regalo`: la lista rappresenta il contenuto del sacco di Babbo Natale nell'ordine in cui andrà riempito (il primo regalo della lista è il primo a entrare nel sacco).
3. Implementare un `main()` che riceva da Babbo Natale il nome del file contenente la lista preparata dagli elfi come argomento a riga di comando - è possibile utilizzare il file `listaElfi.txt` fornito con il testo dell'esercizio per testare il programma.
4. Caricare il contenuto del file all'interno di una lista dinamica composta da elementi di tipo `Regalo` leggendo il file riga per riga, garantendo sempre l'ordinamento discendente in base al peso totale di ogni riga (i regali complessivamente più pesanti dovranno essere inseriti per primi nel sacco).
5. Successivamente, il programma cancella gli eventuali elementi ripetuti contenuti nella lista (confrontando il nome del regalo, la quantità e il peso unitario) e stampa il numero di regali nella lista finale e il loro peso totale.
6. Il programma chiede a Babbo Natale se vuole eliminare i regali troppo pesanti: acquisisce un valore di peso soglia massimo per un singolo regalo e cancella tutti i regali che hanno un peso complessivo superiore. Babbo Natale può continuare ad alleggerire il sacco fino a quando non inserisce `-1`.
7. Dopo ogni modifica, il `main()` deve sempre stampare la lista di regali aggiornata: ogni riga deve contenere il nome del regalo, il suo peso unitario, la quantità e il peso totale (`quantità * peso unitario`).

Le funzionalità del programma devono essere implementate attraverso l'uso delle seguenti funzioni (utilizzare i prototipi richiesti dagli elfi di Babbo Natale):

- `void apriListaDaFile(ListaRegali* lista, char nomeFile[])`
  apre il file TXT contenente l'elenco dei regali e carica il suo contenuto nella lista di elementi di tipo Regalo.
- `void stampaLista(ListaRegali lista)`
  stampa gli elementi della lista.
- `void inserisciInOrdine(ListaRegali* lista, char nomeRegalo[], int pesoUnitario, int qty)`
  inserisce un nuovo regalo nella lista in modo che questa risulti in ordine dopo l'operazione.
- `int calcolaNumeroRegali(ListaRegali lista)`
  calcola e restituisce il numero totale di regali presenti nella lista.
- `int calcolaPesoTotale(ListaRegali lista)`
  calcola e restituisce il peso totale dei regali nella lista.
- `void cancellaRegalo(ListaRegali* lista, char nomeRegalo[], int pesoUnitario, int qty)`
  cancella tutti i regali della lista che hanno le caratteristiche passate come input alla funzione.
  Questa funzione può essere ricorsiva.
- `void cancellaRegaliOltreSoglia(ListaRegali* lista, int sogliaPeso)`
  cancella tutti i regali della lista che hanno un peso complessivo superiore a quello della soglia (passata come input alla funzione).
  Questa funzione può essere ricorsiva.
- `void cancellaRipetizioni(ListaRegali* lista)`
  cancella eventuali ripetizioni dalla lista lasciando una sola occorrenza del regalo presente più volte
  Questa funzione può essere ricorsiva.

Le funzioni richiamate dal `main()` per la gestione della lista devono essere implementate all'interno di una o più librerie che il programma dovrà includere (ad esempio, separando le funzioni ricorsive dalle altre).