ANDREA BARBIN **5°E** **10/10/24**

RELAZIONE TECNICA: PROGETTO LIBRERIA DIGITALE IN C

RICHIESTA

L'obiettivo era quello di creare una libreria virtuale in cui ogni libro avrà diverse informazioni e sarà assegnato a una categoria specifica.
Ogni libro è identificato da titolo, autore, anno di pubblicazione e prezzo, ogni libro viene tracciato da queste informazioni.
I libri sono considerati divisi in categorie determinate da un nome, che indicherà di che tipo di libri si tratta, come "Narrativa" o "Scienze", e conterrà una lista di libri che appartengono a quella specifica categoria.
L'obbiettivo è quello di creare un programma che gestire tutte queste informazioni. Inoltre, dovrete scrivere delle funzioni che vi permettano di stampare tutti i libri che appartengono a una determinata categoria, cercare un libro specifico in base al titolo e trovare tutti i libri che rientrano in una certa categoria.

TEMPISTICHE

Assegnazione: 3/10/2024
Consegna: 10/10/2024

FUNZIONAMENTO DEL SOFTWARE

I libri prelevati da un database CSV vengono memorizzati in un array che li conterrà e permetterà di esaminarli e compiere le opportune operazioni con essi. Il programma è strutturato in modo tale che l'utente possa interagire con chiarezza con il software tramite un menù. È possibile cercare il libro che desidera utilizzando il titolo, visualizzare i libri di una determinata categoria, oppure utilizzare altri criteri per la ricerca.
L'utente potrà infatti visualizzare i libri in base a:

- Titolo, ottenendo il libro preciso
- Autore, ottenendo una lista di tutti i libri scritti da un determinato autore;
- Intervallo di prezzo, inserendo un prezzo minimo e un prezzo massimo, in modo tale da poter visualizzare tutti i libri in questo intervallo di prezzo;

- Anno di pubblicazione, in modo tale da visualizzare tutti i libri pubblicati in un determinato anno.

ASPETTI TECNICI DELL'IMPLEMENTAZIONE

DEFINIZIONE STRUTTURA LIBRO

Nel passaggio dall'ideazione su come realizzare il progetto e la stesura del codice, è stato considerato opportuno utilizzare una struttura chiamata Libro per modellare appunto ogni singolo libro. Questa struttura è così composta:

```
typedef struct
{
   char titolo[MAX_LUNGH_STR];
   char autore[MAX_LUNGH_STR];
   int annoPubblicazione;
   float prezzo;
   Categoria cat;

} Libro;
```

Una stringa per il titolo, una stringa per il nome dell'autore, un membro intero per l'anno di pubblicazione del libro, un membro float per il prezzo, e un membro enumeratore di tipo Categoria per poter memorizzare il genere del libro ("Narrativa, "Scienza", "Arte", "Saggistica", o "Romanzo).

DEFINIZIONE ENUMERATORE CATEGORIA

Per definire la categoria del libro è stato creato il seguente enumeratore:

```
typedef enum
{
   Narrativa,
   Scienza,
   Saggistica,
   Arte,
   Romanzo

} Categoria;
```

Per una più semplice gestione di questo membro della struttura **Libro** durante la fase di ricerca dei libri per categoria, è stata implementata una funzione che ritorna il corrispondente valore in stringa in modo tale da poterlo confrontare con la categoria inserita in input dall'utente, oltre che per gli output. Per evitare malfunzionamenti e garantire un'esperienza ottimale all'utente, ogni stringa inserita in input accetta gli spazi, in modo tale da consentire l'inserimento corretto del nome degli autori e i titoli dei libri per le ricerche.

RICERCA DEI LIBRI

Quando l'utente cerca i libri in base all'autore, all'intervallo di prezzo, o all'anno di pubblicazione, vengono creati degli array dinamici tramite delle funzioni malloc. Dopo aver stampato in output i libri trovati, queste allocazioni di memoria vengono liberate per evitare lo spreco di memoria.

LETTURA DEI DATI DAL FILE

Per la lettura dei libri dal file CSV, questo viene aperto in modalità lettura binaria e viene prestata attenzione al fatto che le colonne sono separate da un punto e virgola. Il nome del file è specificato dall'utente come argomento della riga di comando, accessibile tramite argv[1]. Il programma inizia verificando se l'argomento è presente e, in caso contrario, stampa un messaggio di utilizzo e termina. Successivamente, apre il file in modalità lettura usando fopen. Se l'apertura del file fallisce, viene emesso un messaggio di errore e il programma termina. Una volta aperto con successo il file, il programma legge i dati riga per riga attraverso un ciclo while, utilizzando fgets per acquisire ciascuna riga, che rappresenta un libro. Ogni riga viene quindi elaborata tramite la funzione strtok, che suddivide la riga in campi separati da virgole, corrispondenti agli attributi del libro come titolo, autore e ISBN. Questi attributi vengono memorizzati in una struttura di tipo Libro, che contiene campi per ogni attributo necessario. I dati estratti vengono quindi memorizzati in un array di strutture Libro, incrementando un contatore per tenere traccia del numero di libri letti. Infine, una volta completata la lettura di tutti i libri, il file viene chiuso con fclose per liberare le risorse utilizzate. Questo flusso di lavoro consente di importare e gestire i dati dei libri in modo efficace, a partire da un semplice file CSV. Per ogni token, cioè per ogni parte in cui vengono suddivise le righe separandole in base ai ; , viene controllato che non sia NULL per evitare di inserire nei campi delle allocazioni dell'array libreria dei valori NULL, creando problemi per la gestione dell'intera libreria.
Per garantire una corretta gestione delle righe vuote all'interno del file CSV, il programma include un controllo aggiuntivo. Prima di procedere con l'elaborazione dei dati di ciascuna riga, viene verificato se la lunghezza della riga acquisita è maggiore di zero, utilizzando la funzione strlen. Se la riga è vuota, il ciclo salta direttamente alla lettura della riga successiva, evitando di elaborare dati errati o inconsistenti. Questo controllo viene realizzato con un'istruzione continue, che consente di ignorare la riga vuota e passare alla successiva senza interrompere il flusso di lettura. Questo accorgimento garantisce che solo le righe effettivamente contenenti dati validi siano prese in considerazione, evitando così errori durante l'inserimento dei valori nella libreria.

FUNZIONI IMPLEMENTATE

char* categoriaStringa(Categoria catCercata)

Parametri: **Categoria catCercata**: una variabile di tipo Categoria che indica la categoria da convertire in stringa.

Scopo: Restituisce una stringa che rappresenta il nome della categoria passata come argomento.
Ritorna: Una char* che rappresenta la categoria in formato stringa ("Narrativa", "Scienze", "Saggistica", "Arte").

Categoria stringaACategoria(const char* categoriaStr)

Parametri: **const char* categoriaStr**, una stringa contiene la categoria cercata dall'utente.

Scopo: Converte una stringa in un valore di tipo Categoria.

Valore ritornato: Un valore di tipo Categoria corrispondente alla stringa passata come parametro, con Narrativa come valore di default se non corrisponde nessuna categoria.

void stampaLibro(Libro libro)

Parametri: **Libro libro**, una struttura Libro che contiene le informazioni di un libro.

Scopo: Stampa le informazioni di un singolo libro organizzando l'output dei vari membri della struttura in un formato corretto e facilmente leggibile.

Valore ritornato: void.

int cercaConTitolo(char titolo[])

Parametri: **char titolo[]**, una stringa che rappresenta il titolo del libro da cercare.

Scopo: Cerca un libro nella libreria usando il titolo.

Valore ritornato: Un intero che rappresenta l'indice del libro trovato; ritorna -1 se il libro non è presente.

Libro* cercaConAutore(char autore[], int* size, int nLibri)

Parametri:

- **char autore[]**, una stringa che rappresenta il nome dell'autore.
- **int* size**, un puntatore a intero che restituirà la dimensione dell'array di libri trovati.
- **int nLibri**, il numero di libri presenti nella libreria.

Scopo: Cerca tutti i libri scritti da un determinato autore.

Valore ritornato: Un puntatore a un array dinamico di Libro contenente i libri scritti dall'autore specificato. Aggiorna per riferimento *size con il numero di libri trovati.

void stampaArray(Libro *arrLibri, int size)

Parametri:

- **Libro* arrLibri**, un array di libri.
- **int size**, la dimensione dell'array di libri.

Scopo: Stampa le informazioni di tutti i libri presenti nell'array passato come argomento.

Valore ritornato: void.

void stampaCategoria(char categoria[], int nLibri)

Parametri:

- **char categoria[]**, una stringa che rappresenta la categoria dei libri desiderata.
- **int nLibri**, il numero di libri presenti nella libreria.

Scopo: Stampa tutti i libri appartenenti alla categoria specificata.

Valore ritornato: void.

Libro* CompresiNeiPrezzi(float prezzoMinimo, float prezzoMassimo, int* size, int nLibri)

Parametri:

- **float prezzoMinimo**, l'estremo minore dell'intervallo di prezzo.
- **float prezzoMassimo**, , l'estremo maggiore dell'intervallo di prezzo.
- **int* size**, un puntatore a intero che restituirà la dimensione dell'array di libri trovati per riferimento modificando direttamente il valore della variabile corrispondente che sarà passata come argomento.
- **int nLibri** , il numero di libri presenti nella libreria.

Scopo: Cerca i libri con un prezzo compreso tra prezzoMinimo e prezzoMassimo.

Valore ritornato: Un puntatore a un array dinamico di Libro contenente i libri che rientrano nell'intervallo di prezzo specificato. Aggiorna per riferiemnto *size con il numero di libri trovati.

Libro* cercaAnnoPubb(int anno, int* size, int nLibri)

Parametri:

- **int anno**, l'anno di pubblicazione dei libri da cercare.
- **int* size**, un puntatore a intero che restituirà la dimensione dell'array di libri trovati.
- **int nLibri**, il numero di libri presenti nella libreria.

Scopo: Cerca tutti i libri pubblicati in un anno specifico.

Valore ritornato: Un puntatore a un array dinamico di Libro contenente i libri pubblicati nell'anno specificato. Aggiorna *size con il numero di libri trovati.

int leggiCSV(const char* nomeFile)

Parametri: **const char* nomeFile**, il nome del file CSV da leggere.

Scopo: Legge i dati dal file CSV e popola l'array globale libreria.

Valore ritornato: Un intero che rappresenta il numero di libri letti dal file.

CHIAMATE DELLE FUNZIONI

La funzione main chiama:

- **leggiCSV(nomeFile):** Carica i dati dei libri da un file CSV.
- **cercaConAutore(autoreCercato, &size, nLibri)**: Cerca libri di un autore specificato.
- **cercaConTitolo(titoloCercato)**: Cerca un libro per titolo.
- **cercaAnnoPubb(anno, &size, nLibri)**: Cerca libri pubblicati in un anno specificato.
- **compresiNeiPrezzi(prezzoMinimo, prezzoMassimo, &size, nLibri):** Cerca libri in un intervallo di prezzo specificato.
- **stampaCategoria(categoriaCercata, nLibri):** Stampa i libri appartenenti a una categoria specifica.

La funzione stampaCategoria(char categoria[], int nLibri) chiama:

- **strcasecmp(const char *s1, const char *s2)** per verificare l'uguaglianza tra due stringhe in maniera case insensitive
- **categoriaStringa(char categoria[]),** per convertire il campo categoria del libro in una stringa
- **StampaLibro(Libro libro),** per stampare i dettagli del libro

CONCLUSIONI

Il progetto della libreria digitale è stato realizzato con successo, rispettando le specifiche e le tempistiche inizialmente stabilite. Il software sviluppato consente all'utente di gestire una libreria virtuale in modo intuitivo, permettendo di cercare, visualizzare e filtrare i libri in base a diversi criteri come titolo, autore, anno di pubblicazione e prezzo. La struttura del codice e l'utilizzo di strutture dati appropriate, come le strutture Libro e l'enumeratore Categoria, garantiscono un'organizzazione chiara e modulare del software, facilitando così eventuali futuri miglioramenti.
Le funzioni implementate sono state testate e ottimizzate per gestire i dati in modo efficiente, utilizzando strutture dinamiche e garantendo la corretta liberazione della memoria per evitare sprechi di risorse. La lettura dei dati da un file CSV consente un facile aggiornamento e gestione della libreria, rendendo il software flessibile e adattabile a diverse esigenze.