ASSIGNMENT OF CSE101

(COMPUTER PROGRAMMING)

COMPUTER SCIENCE AND ENGINEERING

Transforming Education Transforming India

LOVELY PROFESSIONAL UNIVERSITY

Jalandhar, Punjab, India.

Book Management System

Submitted by: **Sumant Yadav A01**

Abhilash Prasad A02

Binit Pandey A03

Sahil A04

Submitted to: Mr. SAURABH PANDEY

Introduction

This code is an implementation of a Book Donor System in C. It defines a Book structure and uses an array of Book structures to store information about books. Users can perform operations like adding a book, listing all books, searching for a book by name, deleting a book by ID, and updating a book's information.

The program uses several macros to define constants like the maximum number of books, the maximum length of strings for book names, authors, and publishers. It uses the strcspn function to remove trailing spaces and newlines from strings read in from the user.

The add_book function allows users to add a new book to the system. It checks if the maximum number of books has been reached and if not, it prompts the user to enter the book's information and adds the book to the array.

The list_books function prints a formatted list of all books in the system.

The search_book function prompts the user to enter a book name and searches the array for a matching book. If found, it prints the book's information; otherwise, it reports that the book was not found.

The delete_book function prompts the user to enter a book ID and searches the array for a matching book. If found, it removes the book from the array by shifting all books after it back by one position.

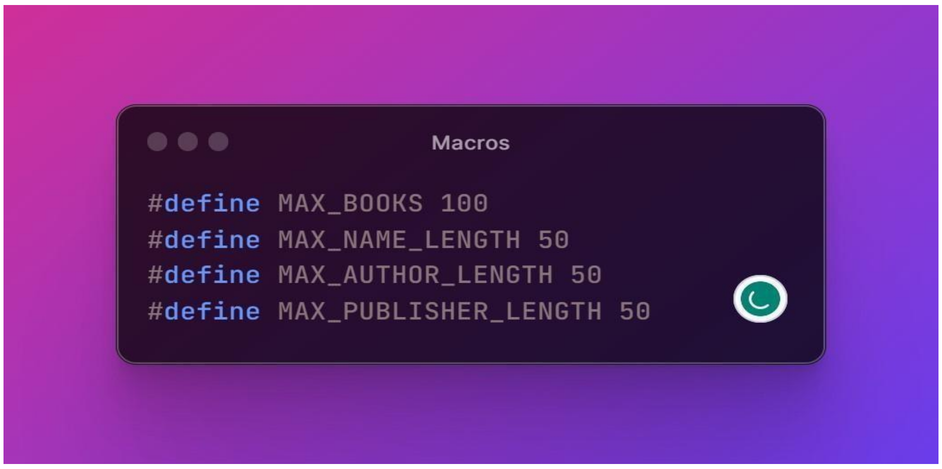
The update_book function prompts the user to enter a book ID and a field to update (name, author, publisher, or year). It then prompts the user for the new value of that field and updates the book's information in the array.

Overall, this code provides basic functionality for managing a collection of books. However, it lacks error handling for invalid user input and does not use dynamic memory allocation, which limits the maximum number of books that can be stored.

Macros

In C programming, macros are a way to define and use reusable code snippets or constants. A macro is essentially a preprocessor directive that tells the compiler to replace a certain sequence of code with another before the actual compilation process begins.

Macros can be defined using the #define directive and are typically used to define constants, functions, or code blocks that need to be reused multiple times throughout a program. Macros can also be used to improve code readability by giving meaningful names to code snippets.

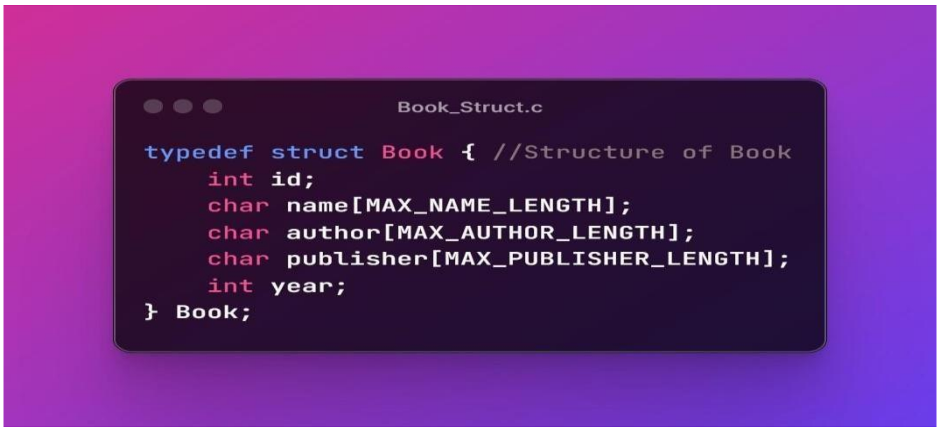


```c
#define MAX_BOOKS 100
#define MAX_NAME_LENGTH 50
#define MAX_AUTHOR_LENGTH 50
#define MAX_PUBLISHER_LENGTH 50
```

Struct and Array of Struct

In C programming, a struct is a composite data type that groups together variables of different data types under a single name. A struct can be used to define a collection of related data elements, such as the properties of a book.

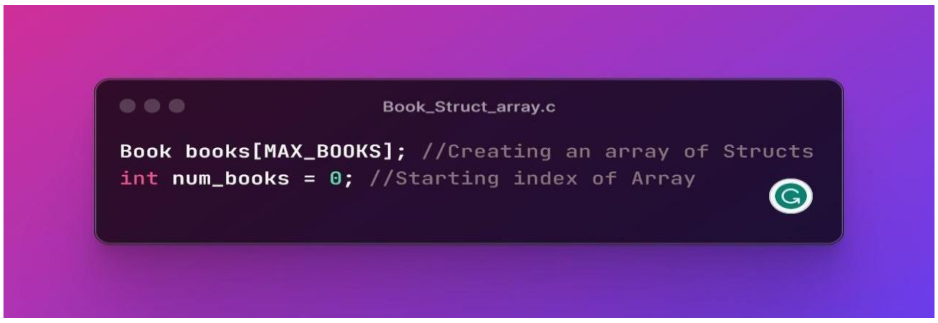
For example, we can define a struct called "book" as follows:



```c
typedef struct Book { //Structure of Book
    int id;
    char name[MAX_NAME_LENGTH];
    char author[MAX_AUTHOR_LENGTH];
    char publisher[MAX_PUBLISHER_LENGTH];
    int year;
} Book;
```

This struct defines a book with four properties: title, author, year_published, and price. Each property has a specific data type.

Now, we can create an array of structs to store multiple books. For example, we can define an array of 100 books as follows:

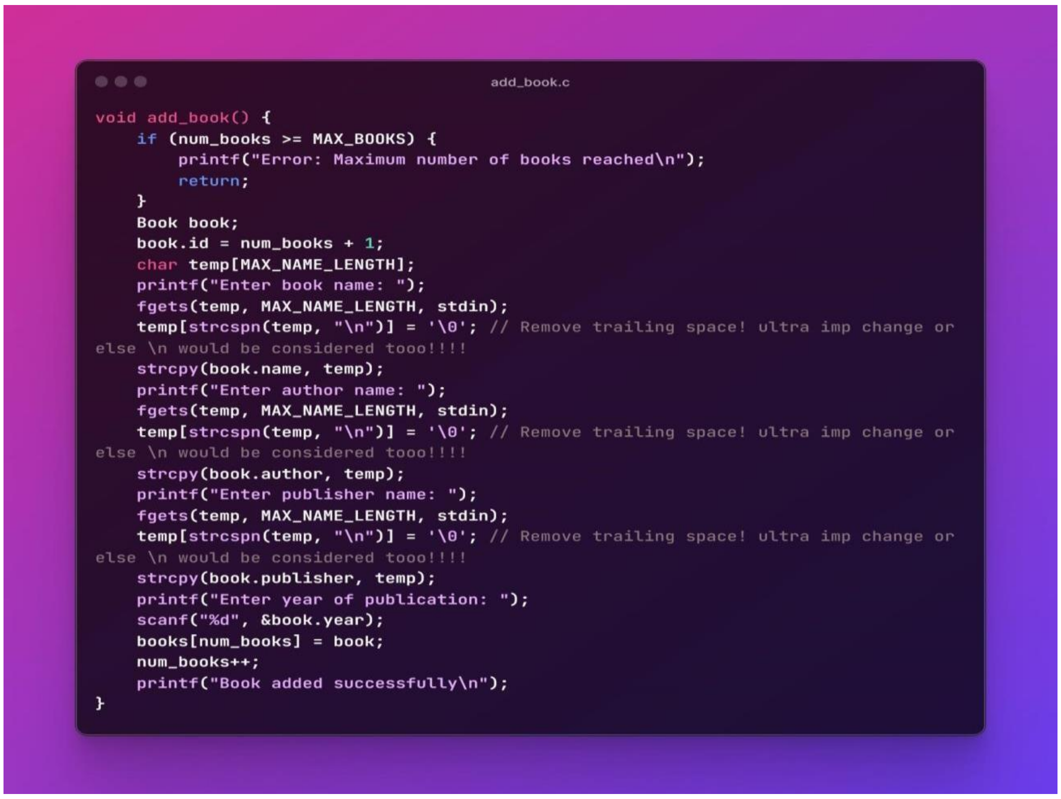


Book_Struct_array.c

```c
Book books[MAX_BOOKS]; //Creating an array of Structs
int num_books = 0; //Starting index of Array
```

Add Book

This module is a function in C programming that adds a new book to an array of book structures. It first checks if the maximum number of books has been reached, and if so, it prints an error message and returns without adding a book. If the maximum number of books has not been reached, it creates a
new book structure and prompts the user to input the book's name, author, publisher, and year of publication. It removes the trailing newline character from the input string and stores the information in the new book structure. Finally, it adds the new book to the array of book structures and increments the number of books in the array. The function then prints a message indicating that the



add_book.c

```c
void add_book() {
    if (num_books >= MAX_BOOKS) {
        printf("Error: Maximum number of books reached\n");
        return;
    }
    Book book;
    book.id = num_books + 1;
    char temp[MAX_NAME_LENGTH];
    printf("Enter book name: ");
    fgets(temp, MAX_NAME_LENGTH, stdin);
    temp[strcspn(temp, "\n")] = '\0'; // Remove trailing space! ultra imp change or
else \n would be considered tooo!!!!
    strcpy(book.name, temp);
    printf("Enter author name: ");
    fgets(temp, MAX_NAME_LENGTH, stdin);
    temp[strcspn(temp, "\n")] = '\0'; // Remove trailing space! ultra imp change or
else \n would be considered tooo!!!!
    strcpy(book.author, temp);
    printf("Enter publisher name: ");
    fgets(temp, MAX_NAME_LENGTH, stdin);
    temp[strcspn(temp, "\n")] = '\0'; // Remove trailing space! ultra imp change or
else \n would be considered tooo!!!!
    strcpy(book.publisher, temp);
    printf("Enter year of publication: ");
    scanf("%d", &book.year);
    books[num_books] = book;
    num_books++;
    printf("Book added successfully\n");
}
```

book was added successfully.

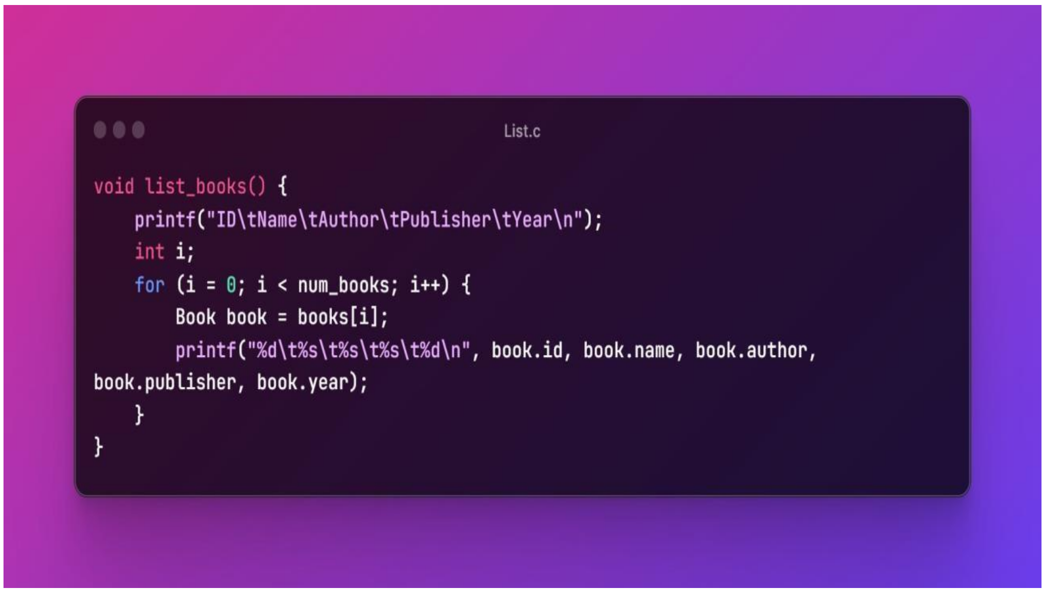
List Book

The function void list_books() is a module that lists all the books in the library. It begins by printing a header row indicating the different columns of information about each book (ID, Name, Author, Publisher, and Year).

It then enters a loop that iterates through each book in the books array, which stores the details of all the books in the library.

For each book, it assigns it to a temporary variable named book and then prints its details using printf(). The details of the book are printed in the same order as the header row: ID, Name, Author, Publisher, and Year.

The loop continues until all the books in the library have been listed.



```c
void list_books() {
    printf("ID\tName\tAuthor\tPublisher\tYear\n");
    int i;
    for (i = 0; i < num_books; i++) {
        Book book = books[i];
        printf("%d\t%s\t%s\t%s\t%d\n", book.id, book.name, book.author,
book.publisher, book.year);
    }
}
```

List Book

This is a module that allows the user to search for a book in the library by its name.

It first prompts the user to enter the name of the book they are looking for, and reads in the input using fgets(). It then removes any trailing spaces from the input string using strcspn().

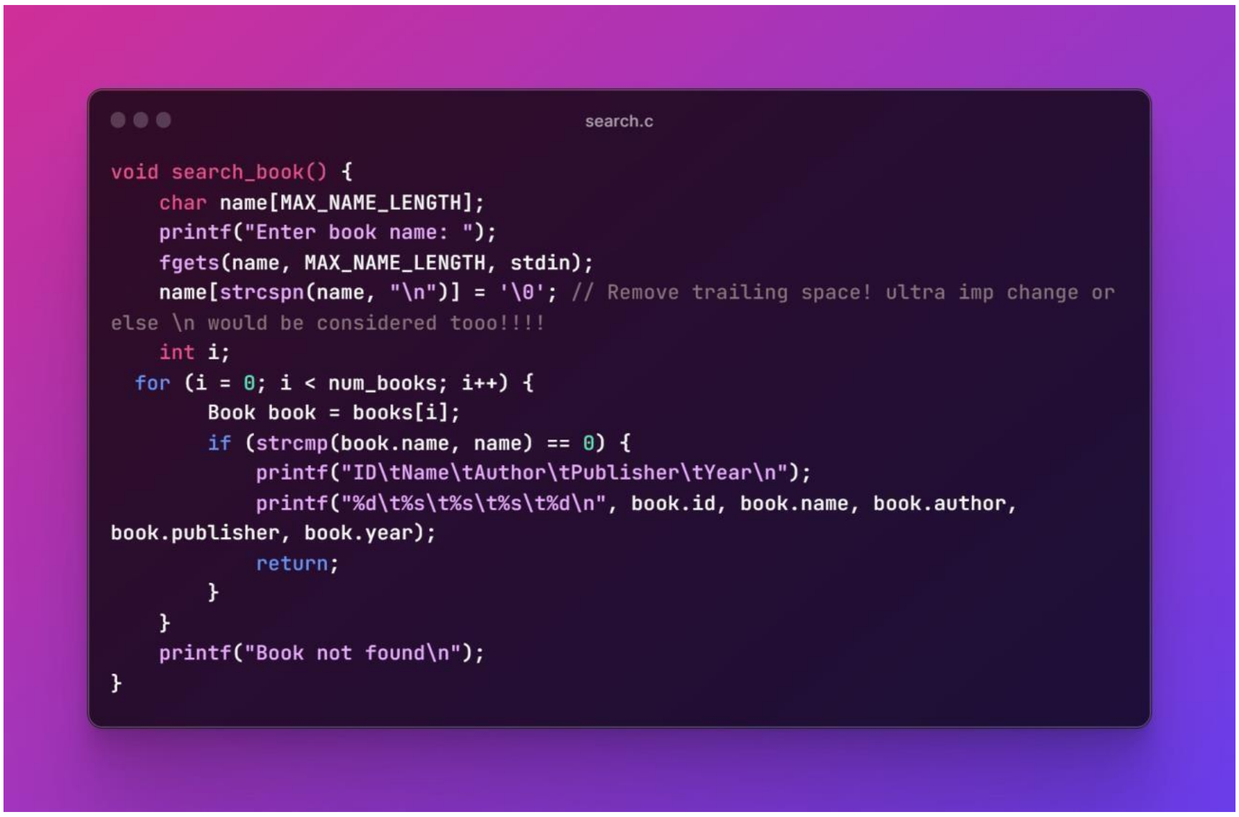
The function then loops through the array of books using a for loop and checks

if the name of each book matches the input string using strcmp(). If it finds a

match, it prints out the details of the book including its

ID, name, author, publisher, and year of publication, and returns from the function.

If the loop finishes and no match is found, it prints out a message indicating that the book was not found.


search.c

```
void search_book() {
    char name[MAX_NAME_LENGTH];
    printf("Enter book name: ");
    fgets(name, MAX_NAME_LENGTH, stdin);
    name[strcspn(name, "\n")] = '\0'; // Remove trailing space! ultra imp change or
else \n would be considered tooo!!!!
    int i;
  for (i = 0; i < num_books; i++) {
        Book book = books[i];
        if (strcmp(book.name, name) == 0) {
            printf("ID\tName\tAuthor\tPublisher\tYear\n");
            printf("%d\t%s\t%s\t%s\t%d\n", book.id, book.name, book.author,
book.publisher, book.year);
            return;
        }
    }
    printf("Book not found\n");
}
```

Delete Book

The delete_book() function in this C programming module allows the user to

delete a book from the library system based on its ID.

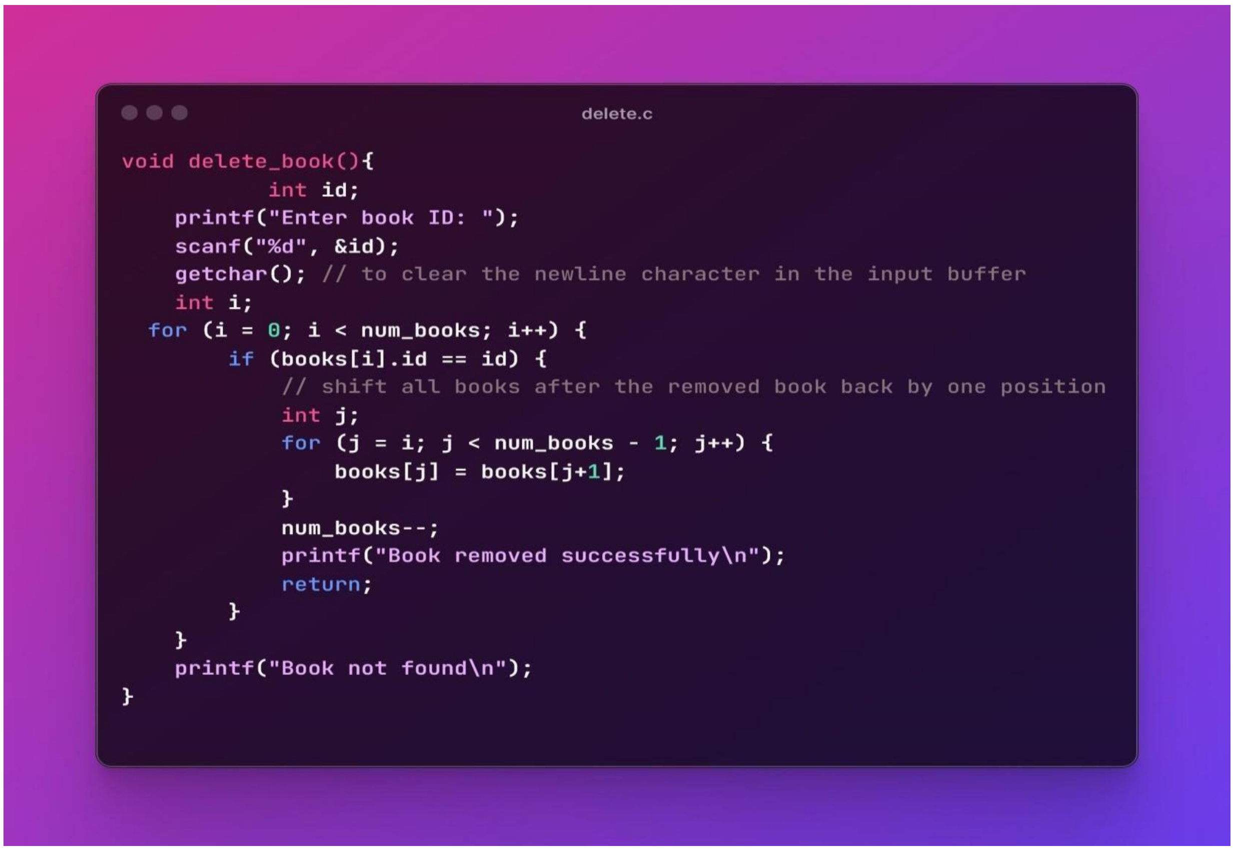
The module first prompts the user to enter the ID of the book they want to

remove. It then loops through the array of books to find the book with the

matching ID. If it is found, the function removes the book by shifting all the

books after it back by one position and decrementing num_books. Finally, the function prints a message indicating that the book has been removed and returns.

If no book is found with the given ID, the function prints a message indicating that the book was not found in the library system.


delete.c

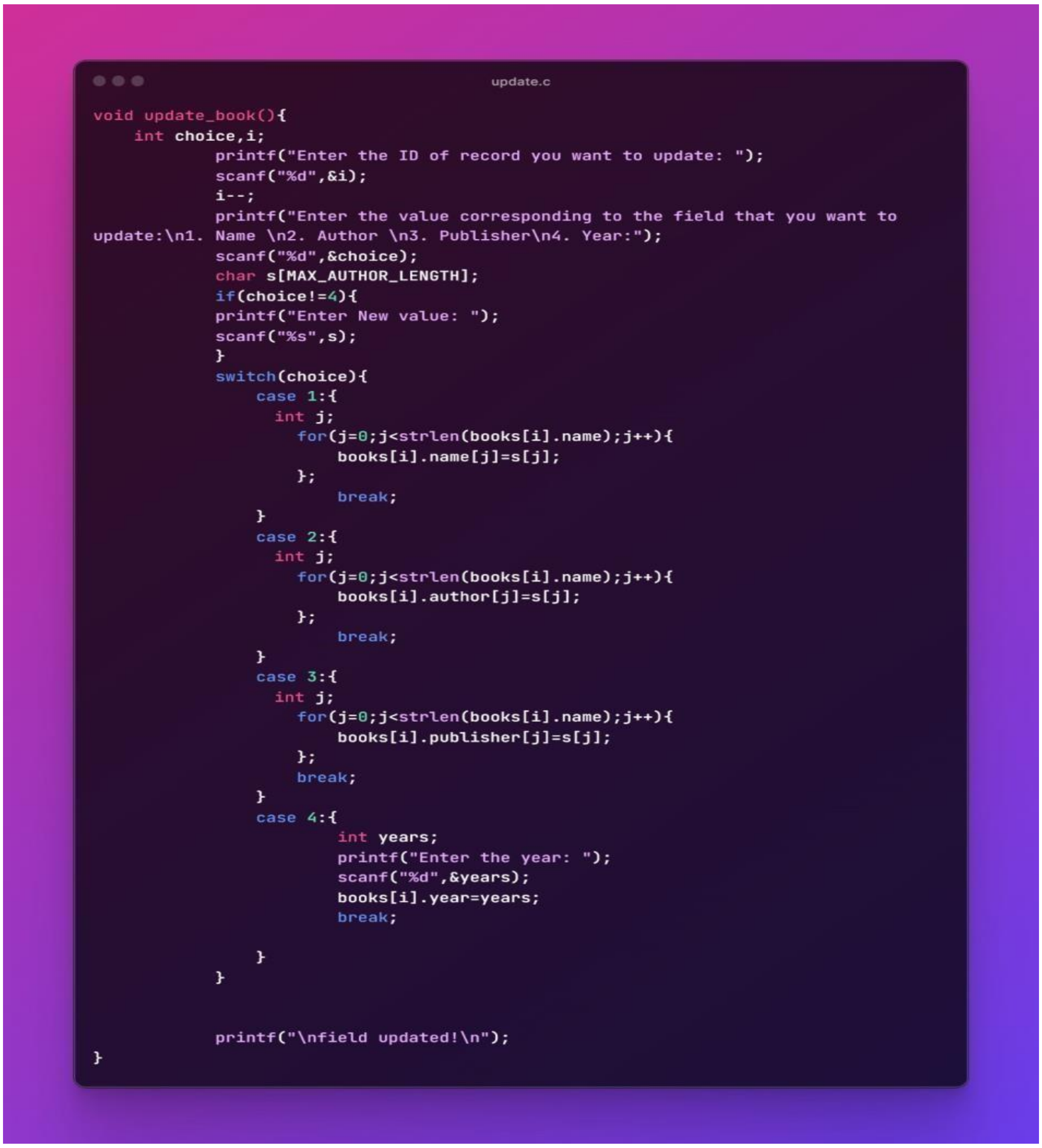
```c
void delete_book(){
            int id;
    printf("Enter book ID: ");
    scanf("%d", &id);
    getchar(); // to clear the newline character in the input buffer
    int i;
  for (i = 0; i < num_books; i++) {
        if (books[i].id == id) {
            // shift all books after the removed book back by one position
            int j;
            for (j = i; j < num_books - 1; j++) {
                books[j] = books[j+1];
            }
            num_books--;
            printf("Book removed successfully\n");
            return;
        }
    }
    printf("Book not found\n");
}
```

Update Book

The update_book() function is used to update an existing book record in the library system. It first prompts the user to enter the ID of the record they want to update and the field they want to update (either name, author, publisher or year). If the user chooses to update the name, author or publisher fields, they are prompted to enter the new value for that field. If the user chooses to update the year field, they are prompted to enter the new year value.

After getting the necessary input from the user, the function uses a switch-case statement to determine which field needs to be updated. It then uses a for loop to copy the new value into the corresponding field of

the book record.



```c
void update_book(){
    int choice,i;
            printf("Enter the ID of record you want to update: ");
            scanf("%d",&i);
            i--;
            printf("Enter the value corresponding to the field that you want to 
update:\n1. Name \n2. Author \n3. Publisher\n4. Year:");
            scanf("%d",&choice);
            char s[MAX_AUTHOR_LENGTH];
            if(choice!=4){
            printf("Enter New value: ");
            scanf("%s",s);
            }
            switch(choice){
                case 1:{
                  int j;
                    for(j=0;j<strlen(books[i].name);j++){
                        books[i].name[j]=s[j];
                    };
                        break;
                }
                case 2:{
                  int j;
                    for(j=0;j<strlen(books[i].name);j++){
                        books[i].author[j]=s[j];
                    };
                        break;
                }
                case 3:{
                  int j;
                    for(j=0;j<strlen(books[i].name);j++){
                        books[i].publisher[j]=s[j];
                    };
                    break;
                }
                case 4:{
                        int years;
                        printf("Enter the year: ");
                        scanf("%d",&years);
                        books[i].year=years;
                        break;

                }
            }


            printf("\nfield updated!\n");
}
```

Main Function

This is the main function of the Book Donor System program. It presents a menu to the user and allows them to perform various operations on the list of books.

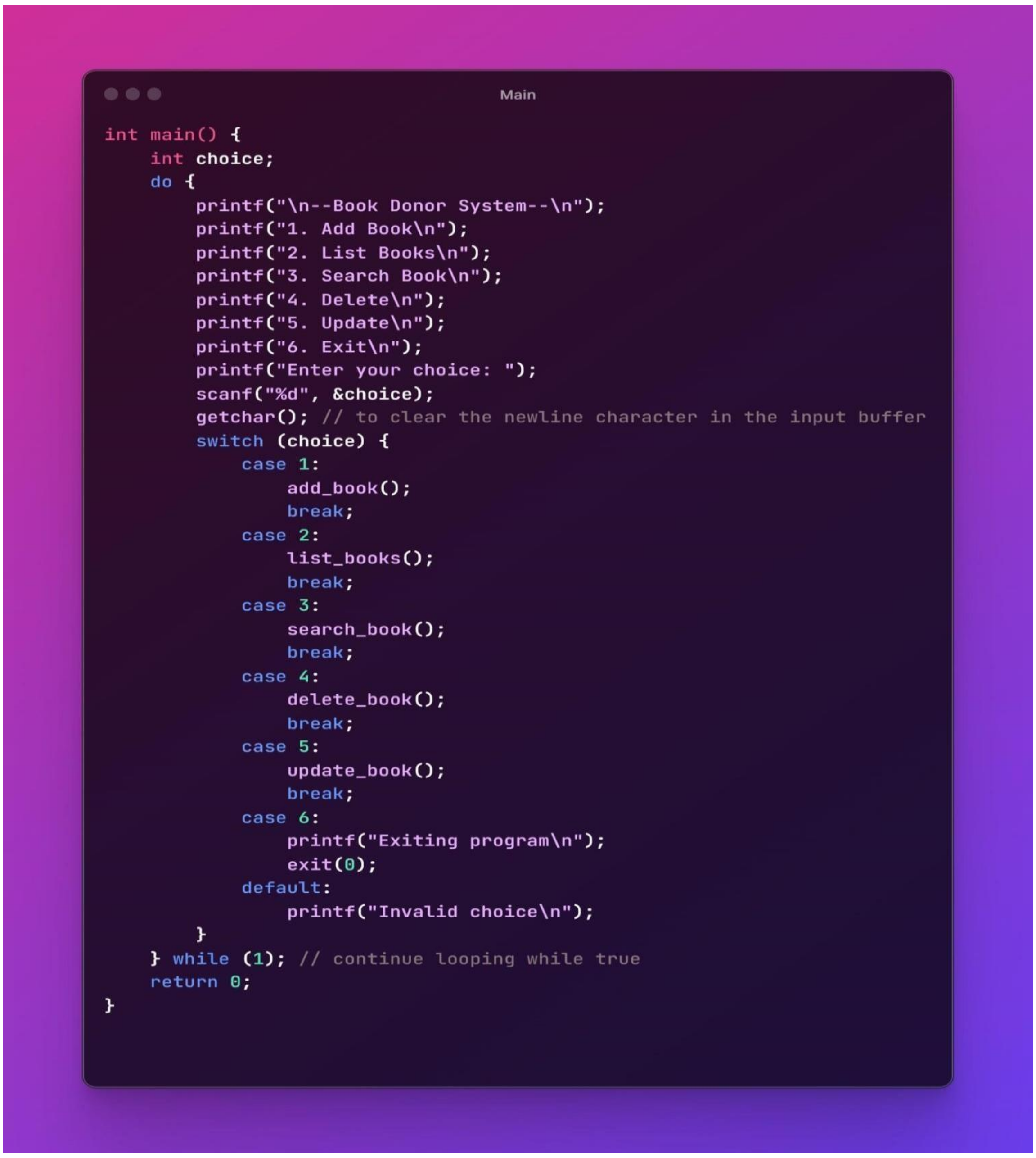
The menu has six options:

- Add Book: This option allows the user to add a new book to the list.
- List Books: This option displays the list of all books that have been added so far.
- Search Book: This option allows the user to search for a specific book by name.
- Delete: This option allows the user to delete a book from the list by ID.
- Update: This option allows the user to update the details of a book by ID.
- Exit: This option allows the user to exit the program.

The function uses a do-while loop to continue looping through the menu until the user chooses to exit the program. Each option is implemented using a separate function that performs the necessary actions. The program uses getchar() to clear the newline character in the input buffer after reading in the user's choice.

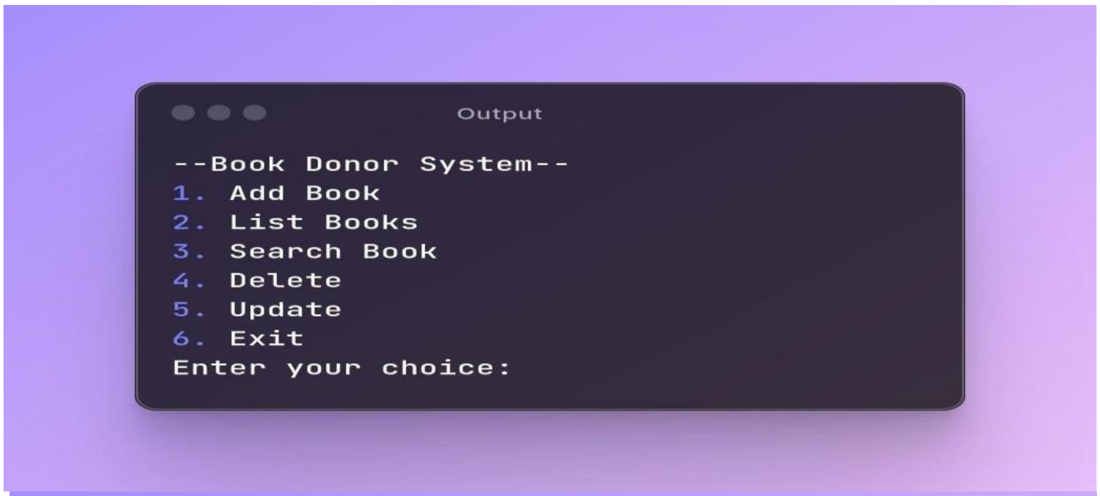
If the user enters an invalid choice, the program prints a message indicating that the choice is invalid. Finally, the function returns 0 to indicate successful program execution.

Input

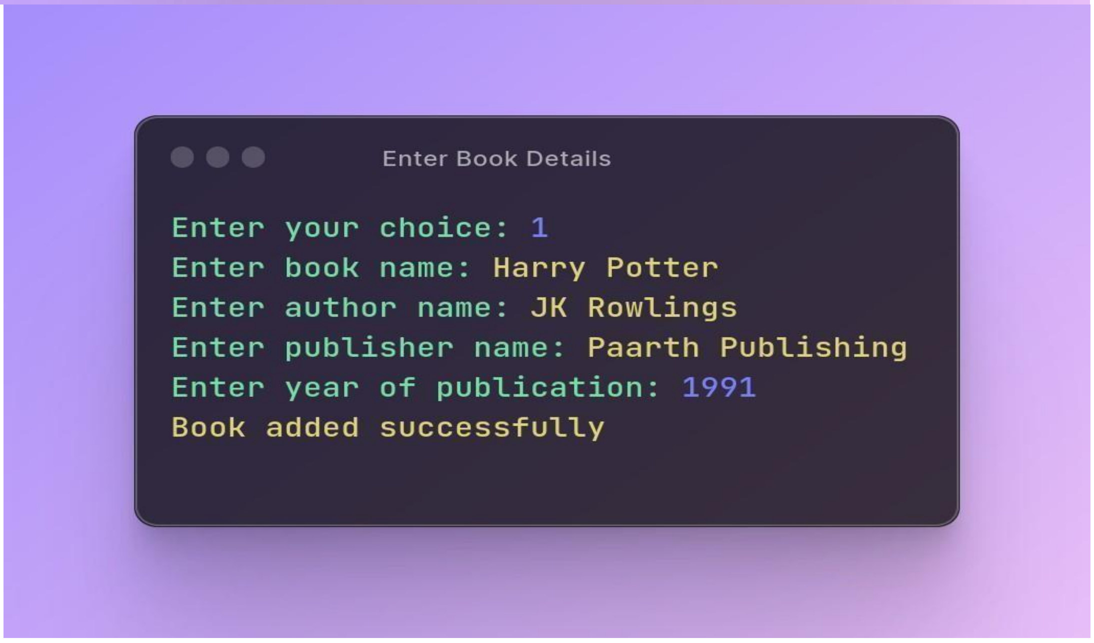


```c
int main() {
    int choice;
    do {
        printf("\n--Book Donor System--\n");
        printf("1. Add Book\n");
        printf("2. List Books\n");
        printf("3. Search Book\n");
        printf("4. Delete\n");
        printf("5. Update\n");
        printf("6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        getchar(); // to clear the newline character in the input buffer
        switch (choice) {
            case 1:
                add_book();
                break;
            case 2:
                list_books();
                break;
            case 3:
                search_book();
                break;
            case 4:
                delete_book();
                break;
            case 5:
                update_book();
                break;
            case 6:
                printf("Exiting program\n");
                exit(0);
            default:
                printf("Invalid choice\n");
        }
    } while (1); // continue looping while true
    return 0;
}
```

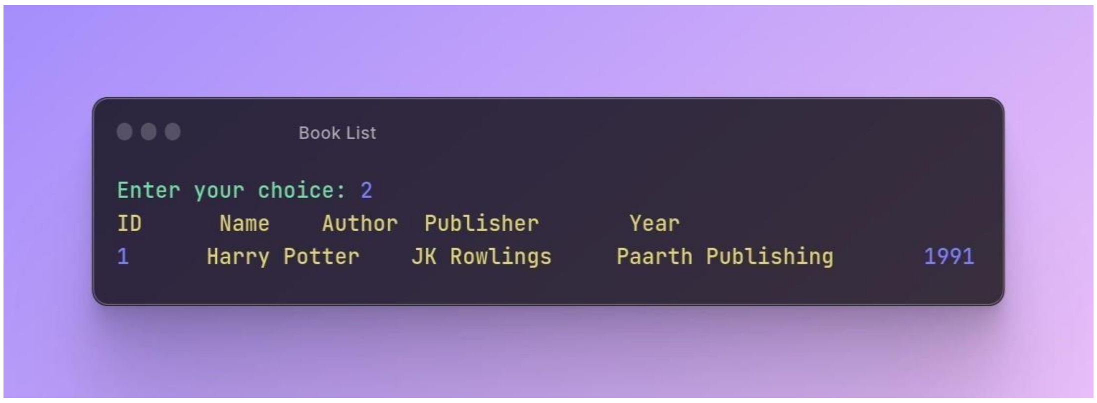
Output



```
--Book Donor System--
1. Add Book
2. List Books
3. Search Book
4. Delete
5. Update
6. Exit
Enter your choice:
```



```
Enter your choice: 1
Enter book name: Harry Potter
Enter author name: JK Rowlings
Enter publisher name: Paarth Publishing
Enter year of publication: 1991
Book added successfully
```



```
Enter your choice: 2
ID      Name     Author  Publisher        Year
1      Harry Potter     JK Rowlings     Paarth Publishing        1991
```

Conclusion

In conclusion, the Book Donor System program is a simple implementation of a book management system using C programming language. The program allows users to perform various operations on a collection of books, including adding new books, listing existing books, searching for a specific book, deleting a book, and updating book information. The program utilizes basic input/output operations, string manipulation functions, and control structures to achieve its functionality. While this program may not be suitable for large-scale book management systems, it provides a good starting point for beginners to understand the basics of programming and file handling.