LIBRARY MANAGEMENT SYSTEM

Submitted by,
Adithya Ashok
Roll No:2
Computer Science Dept

17-07-2024

INTRODUCTION

The efficient management of library resources is crucial for educational institutions, public libraries, and various other organizations. A Library Management System (LMS) provides a comprehensive solution for handling the cataloging, issuing, and returning of books. This report presents the development of a Library Management System using C programming language, focusing on essential features such as adding books, issuing books to users, and calculating fines for overdue returns.

PROBLEM STATEMENT

Libraries play a crucial role in the dissemination of knowledge and information. However, managing a library's resources manually can be time-consuming, error-prone, and inefficient. To address these challenges, there is a need for an automated Library Management System that can streamline the processes of adding books, issuing books to users, returning books, and calculating fines for overdue returns. This system should be developed using the C programming language to leverage its efficiency and control over hardware resources.

OBJECTIVES

1.Adding Books: Create functionality to add new books to the library's inventory. This includes storing details such as the book ID, title, author, and availability status.
2.Issuing Books: Implement a feature to issue books to registered users. This involves recording the user's information, the book's details, and the date of issue.
3.Returning Books: Develop a system to handle the return of issued books. This includes updating the book's availability status and recording the return date.
4.Calculating Fines: Establish a mechanism to calculate fines for overdue books based on a predefined set of rules. The system should be able to compute the number of days a book is overdue and apply the corresponding fine.

SYSTEM SPECIFICATIONS

Hardware Requirements

1.User's Laptop Specifications:

1. Operating System: Windows 11
2. Processor: 13th Generation Intel i5 Core-13450HX
3. RAM: 16 GB
4. Storage: 512 GB SSD

Software Requirements

1.Operating System:

1. Windows 11

2.Development Environment:

1. C Compiler: GCC (GNU Compiler Collection)
2. IDE: Code: Visual Studio Code

DESIGN AND DEVELOPMENT

Description of Program Logic

The Library Management System (LMS) is designed to handle the key operations of a library, including adding books, issuing books, returning books, and calculating fines. The program logic is structured into several modules, each responsible for a specific functionality. The main modules are:

1. **Main Menu**: Provides options for different operations (add book, issue book, return book, calculate fine, view books, view users).
2. **Add Book**: Allows the librarian to add new books to the inventory.
3. **Issue Book**: Facilitates the issuing of books to registered users.
4. **Return Book**: Manages the return of issued books and updates the inventory.
5. **Calculate Fine**: Computes fines for overdue books.
6. **View Books**: Displays the list of all books in the library.

PSUEDOCODE

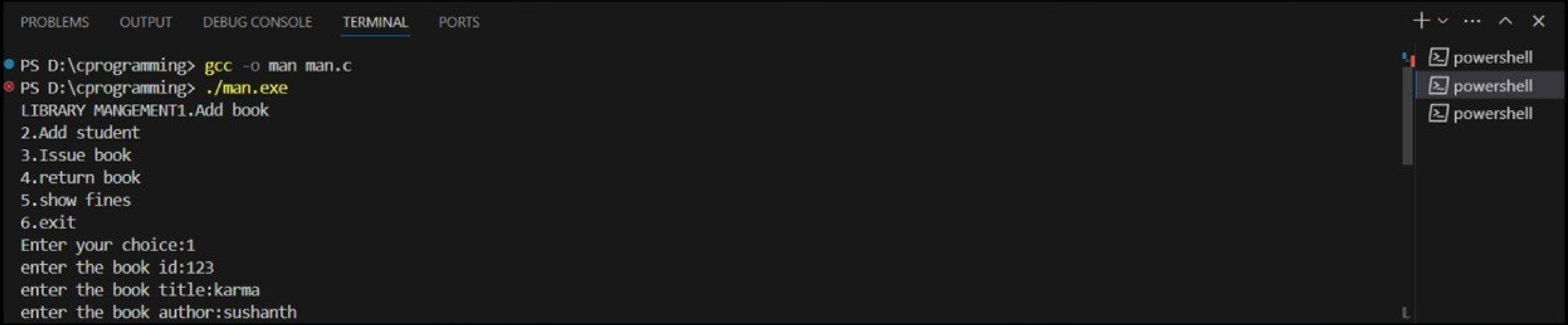
```
START
  DISPLAY "LIBRARY MANAGEMENT"
  DISPLAY "1. Add book"
  DISPLAY "2. Add student"
  DISPLAY "3. Issue book"
  DISPLAY "4. Return book"
  DISPLAY "5. Show fines"
  DISPLAY "6. Exit"

  REPEAT
    DISPLAY "Enter your choice:"
    INPUT choice

    SWITCH choice
      CASE 1:
        CALL add_book()
      CASE 2:
        CALL add_student()
      CASE 3:
        CALL issue_book()
      CASE 4:
        CALL return_book()
      CASE 5:
        CALL show_fine()
      CASE 6:
        EXIT
      DEFAULT:
        DISPLAY "Invalid choice!"
    END SWITCH
  UNTIL choice == 6
END
```

INPUT STUDENT DATA

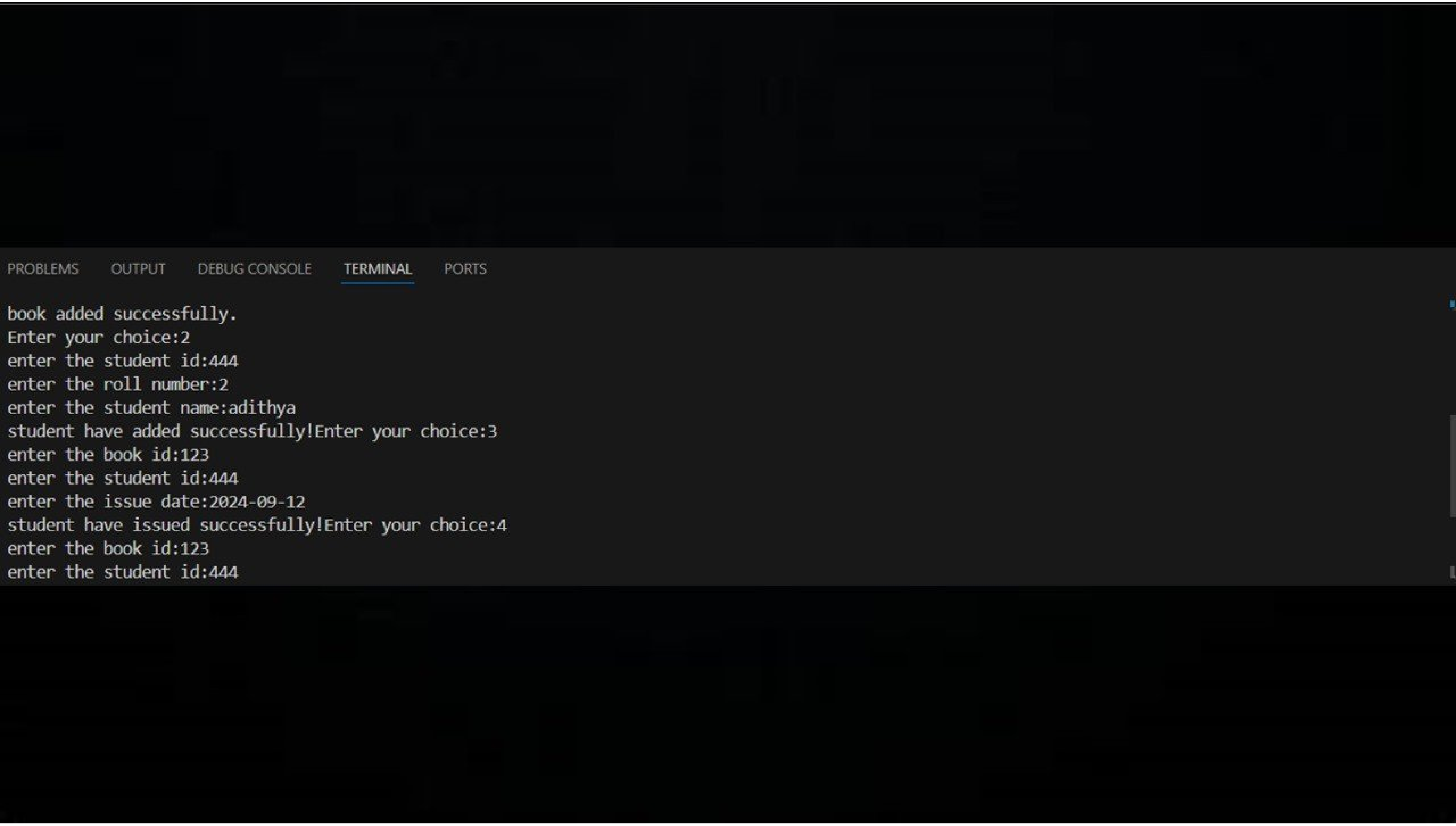
```
PROCEDURE inputStudentData()
  IF num_students < MAX_STUDENTS THEN
    INPUT "Enter student ID:" INTO students[num_students].id
    INPUT "Enter student name:" INTO students[num_students].name
    FOR day FROM 0 TO MAX_DAYS - 1
      INPUT "Enter attendance for day", day + 1, "(0 for absent, 1 for present):" INTO students[num_students].attendance[day]
    END FOR
    INCREMENT num_students
  ELSE
    DISPLAY "Maximum number of students reached."
  END IF
END PROCEDURE
```

GENERATE ATTENDANCE REPORT

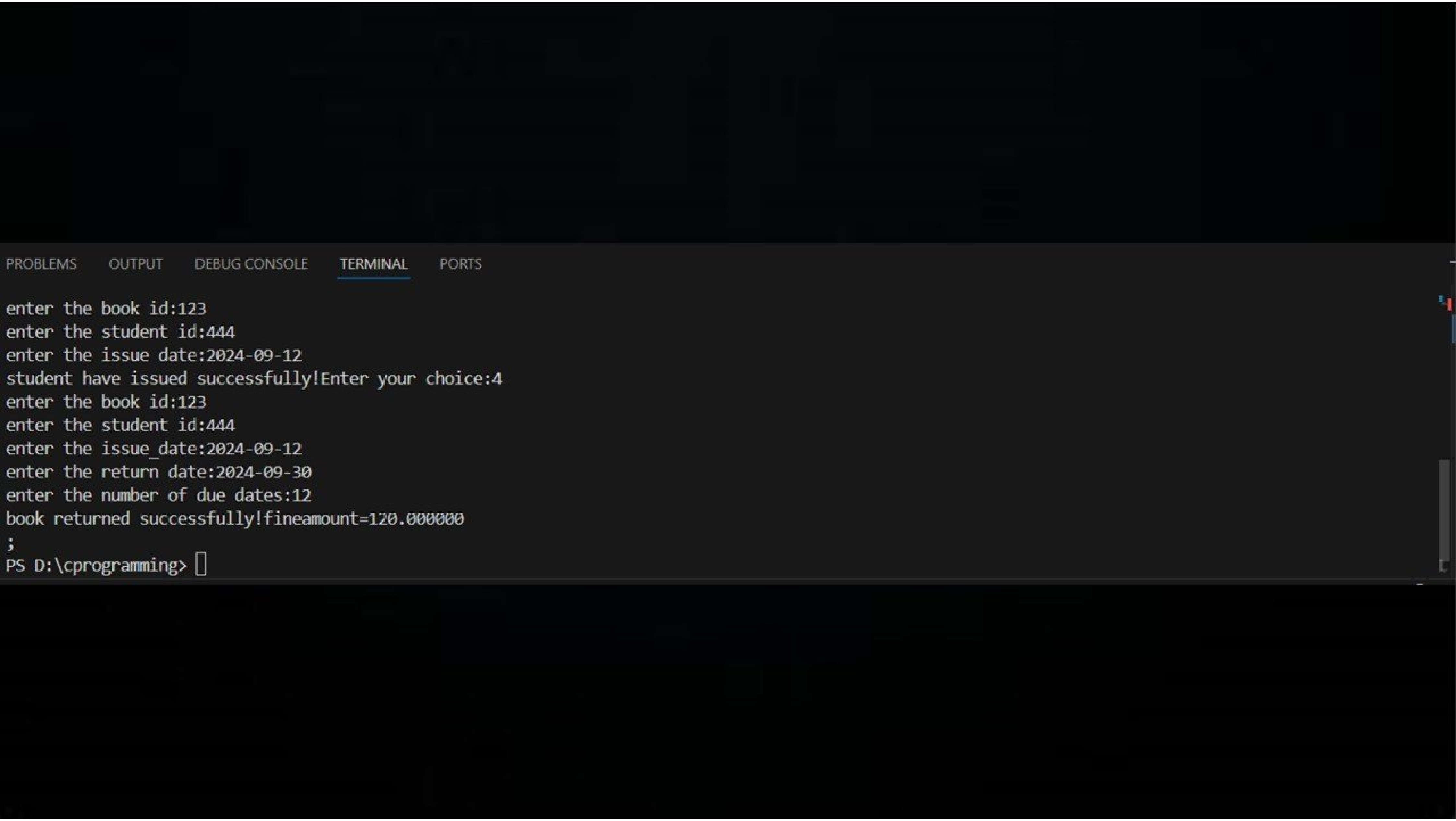
```
PROCEDURE generateAttendanceReport()
  FOR EACH student IN students UP TO num_students - 1
    SET present_count TO 0
    FOR day FROM 0 TO MAX_DAYS - 1
      IF student.attendance[day] == 1 THEN
        INCREMENT present_count
      END IF
    END FOR
    SET attendance_percentage TO (present_count / MAX_DAYS) * 100
    DISPLAY "Student ID:", student.id, ", Name:", student.name, ",
Attendance:", attendance_percentage, "%"
  END FOR
END PROCEDURE
```



```
PS D:\cprogramming> gcc -o man man.c
PS D:\cprogramming> ./man.exe
LIBRARY MANGEMENT1.Add book
2.Add student
3.Issue book
4.return book
5.show fines
6.exit
Enter your choice:1
enter the book id:123
enter the book title:karma
enter the book author:sushanth
```



```
book added successfully.
Enter your choice:2
enter the student id:444
enter the roll number:2
enter the student name:adithya
student have added successfully!Enter your choice:3
enter the book id:123
enter the student id:444
enter the issue date:2024-09-12
student have issued successfully!Enter your choice:4
enter the book id:123
enter the student id:444
```



```
enter the book id:123
enter the student id:444
enter the issue date:2024-09-12
student have issued successfully!Enter your choice:4
enter the book id:123
enter the student id:444
enter the issue_date:2024-09-12
enter the return date:2024-09-30
enter the number of due dates:12
book returned successfully!fineamount=120.000000
;
PS D:\cprogramming> 
```

SUMMARY

This Library Management System project in C provides a basic yet functional platform to manage books, students, book issuance, and returns, along with tracking fines for overdue books. The system uses text files for persistent data storage. Users can add books by entering the book's ID, title, and author, which are then saved in books.txt. Similarly, students can be added with their ID, roll number, and name, stored in students.txt. Books can be issued to students by providing the book ID, student ID, and issue date, and these records are saved in issues.txt. When books are returned, the system records the return details along with the number of overdue days and calculates fines if applicable. These details are stored in returned.txt, and any fines are recorded in fines.txt. The system includes functions to add books and students, issue and return books, and display all fines. The main function provides a menu-driven interface for user interaction, making it straightforward to manage library operations.

Future enhancements for this Library Management System project could include:

1.Graphical User Interface (GUI): Implement a user-friendly graphical interface using libraries like GTK or QT for better user experience and easier interaction compared to a command-line interface.
2.Database Integration: Replace text file storage with a relational database management system (RDBMS) like MySQL or SQLite to improve data handling, query capabilities, and overall system performance.
3.Advanced Search and Filtering: Add functionality to search for books and students using various criteria such as book title, author, student name, or ID, and filter results accordingly.
4.User Authentication: Implement user authentication and authorization to ensure that only authorized personnel can add, issue, or return books, enhancing security.
5.Fine Calculation Automation: Automate fine calculation by integrating a real-time clock to automatically compute overdue days and fines without manual input.
6.Notification System: Implement a notification system to alert students and librarians about due dates, overdue books, and fines via email or SMS.
7.Book Reservation System: Allow students to reserve books that are currently checked out, ensuring they can borrow the book once it is returned.
8.Reporting and Analytics: Develop detailed reporting and analytics features to provide insights into library usage, popular books, frequent borrowers, and more.
9.Mobile Application: Create a companion mobile application to allow students and librarians to interact with the library system on the go.
10.Barcode/QR Code Integration: Integrate barcode or QR code scanning for quicker and more accurate data entry of books and student IDs.
11.Multilingual Support: Add support for multiple languages to make the system accessible to a broader audience.
These enhancements would significantly improve the functionality, usability, and efficiency of the Library Management System.