**British University College**

### 2023-2024 Academic Year



Assignment

**COS212 – Object-Oriented Programming with C++**

## by

## Naing Min Khant (HDIT Class - D)

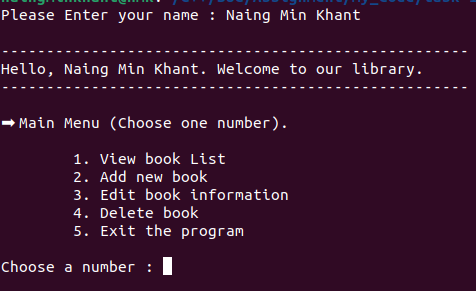
Presented On – June 30 2023

**Table of Content**

* [Task 1 - I/O File Manipulation 2](#_TOC_250002)
* [Task 2 - Inheritance 20](#_TOC_250001)
* [Task 3 - Encapsulation 23](#_TOC_250000)

# Task 1 - I/O File Manipulation

A simple library management system with file manipulation including read, write, edit, and delete features. Once we run a program, it will ask for the guest's name and print the greeting message as shown in Fig (1.1). Then, Main Menu will display the options. We can see the code behind this output in Fig (1.2), Fig (1.2.1), and Fig (1.3).



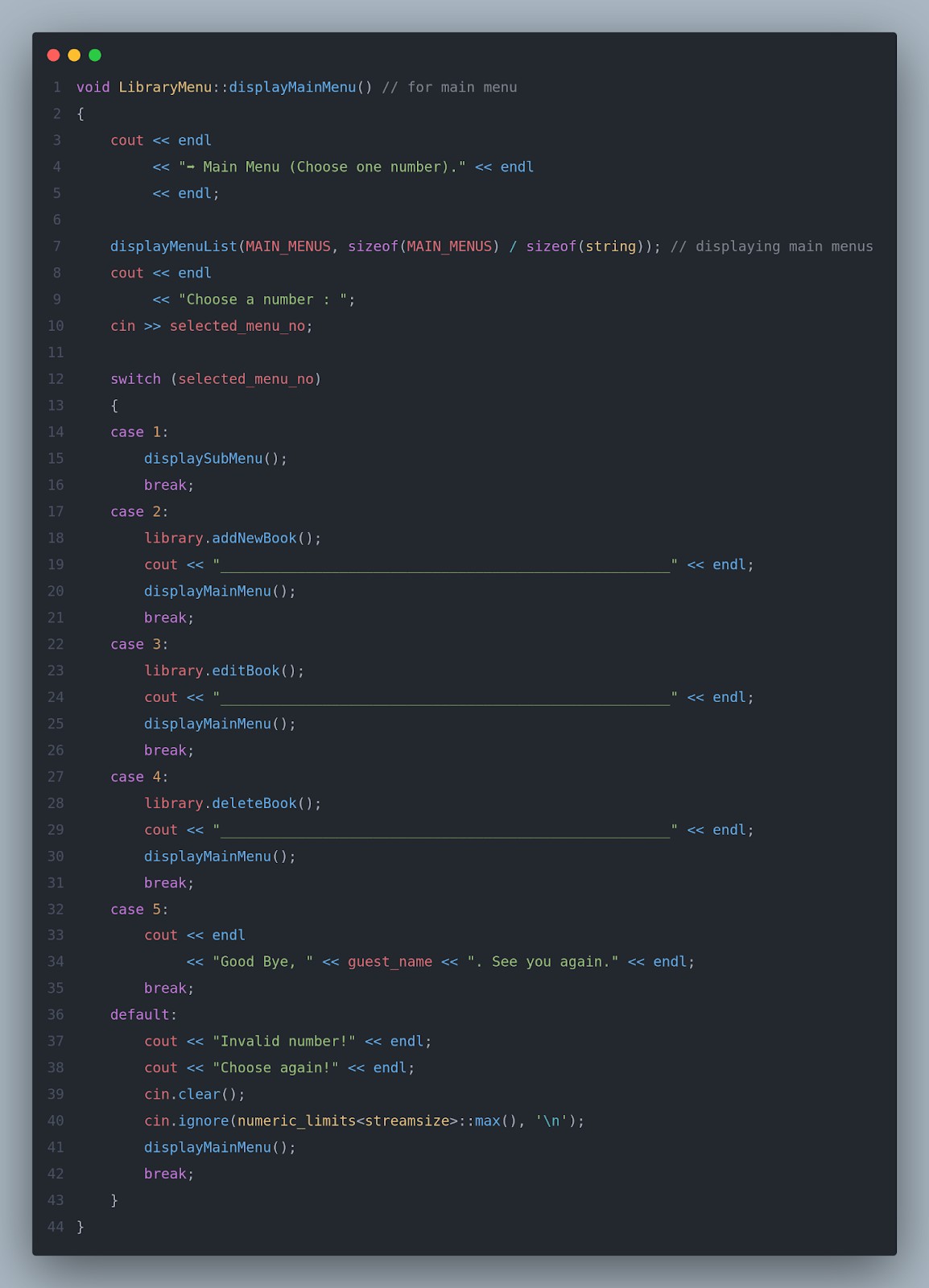
Fig(1.1)



Fig(1.2)

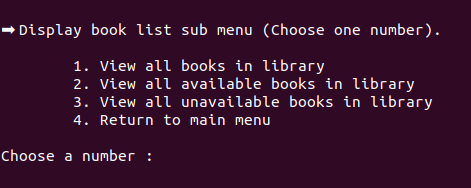


Fig(1.2.1)



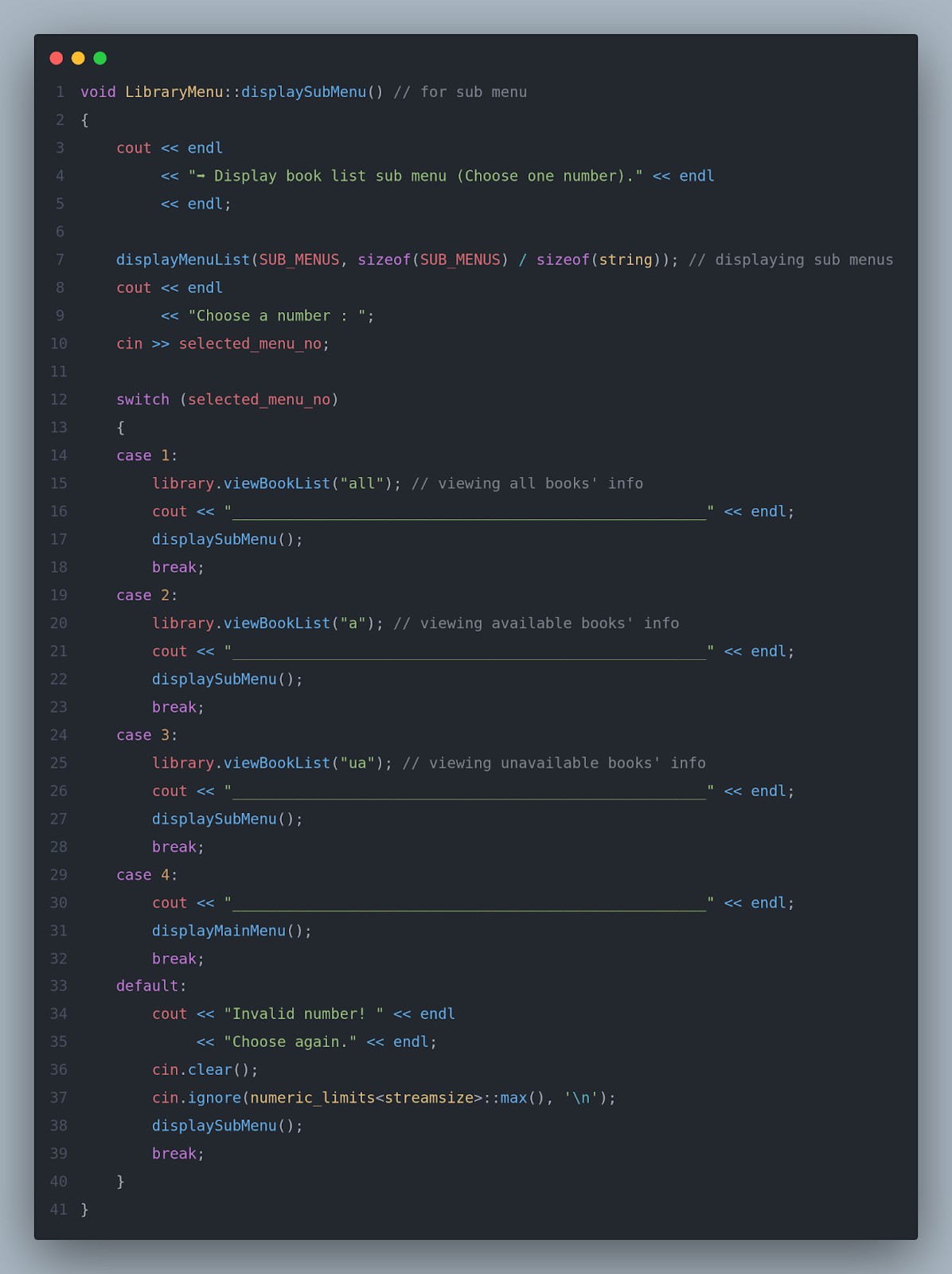
Fig(1.3)

If we choose the number 1, a sub-menu will display. There are more options for viewing books. The sub-menu will be shown so that the user can choose the option to view the book list as shown in Fig (1.4).



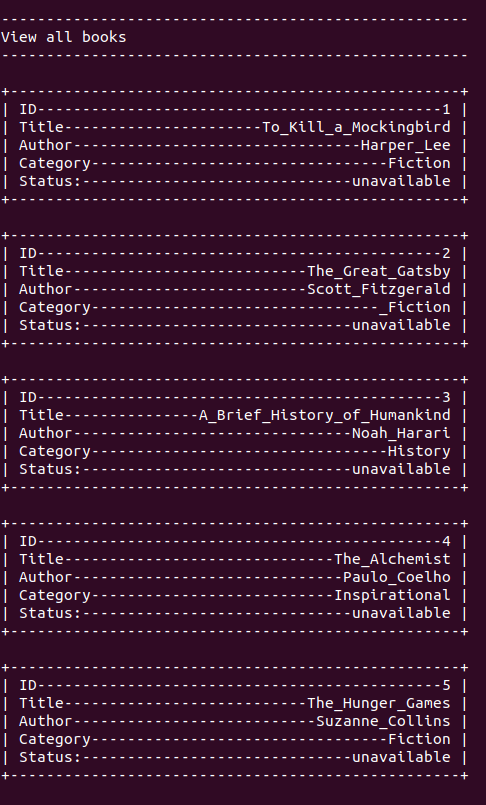
Fig(1.4)

The code working behind this output can be seen in Fig (1.5).



Fig(1.5)

If we choose the number 1, we will see all of the books in the library like in Fig(1.6). Numbers 2 and 3 will work the same as number 1. Number 2 will only show the available books in the library while number 3 does the opposite. It will show the unavailable books. The code behind this feature can be seen in Fig(1.7) and Fig(1.8).



Fig(1.6)

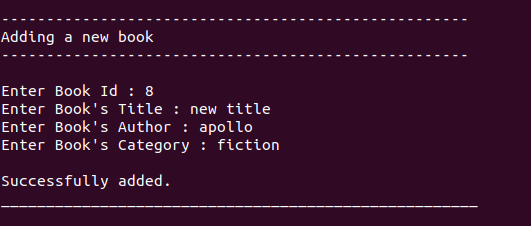


Fig(1.7)



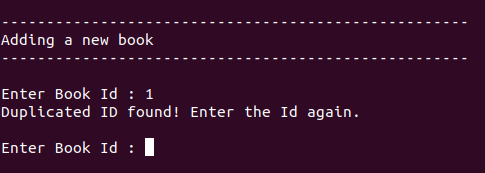
Fig(1.8)

If we click number 2 of the main menu, the program will ask for the data for a new book. As we can see in Fig(1.9).

.

Fig(1.9)

If we add the id that already exists, the program will show the warning message and ask for the id again as shown in Fig (1.10). We can see the code for this feature in Fig (1.11) and Fig(1.12).



Fig(1.10)

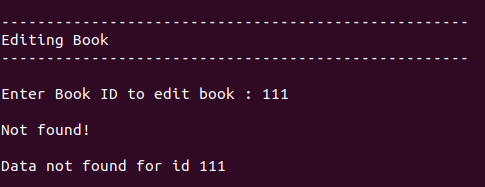


Fig(1.11)



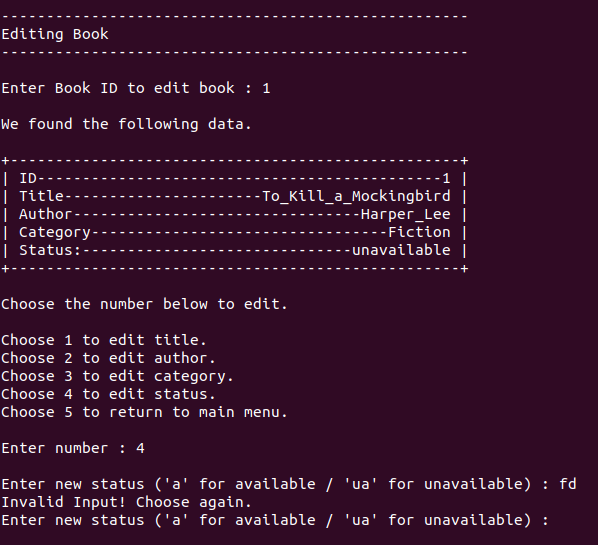
Fig(1.12)

Number 3 is for editing the data of the book using book id. Firstly, we need to input the id of the book that we want to edit. So, the program will find the book using the id that we provided. If we enter the id which does not exist in the library, the program will say a ‘Not Found’ message as we can see in Fig (1.13).



Fig(1.13)

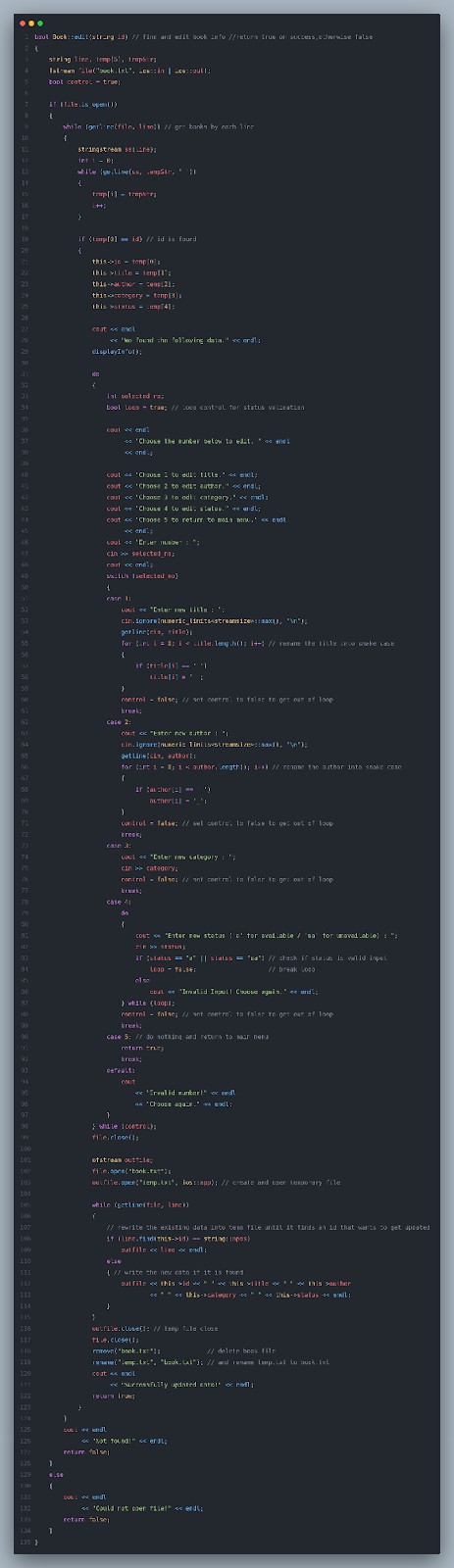
If we enter the invalid input for the book’s status in editing, the program will print a message and ask for the input again. That feature is shown in Fig (1.14). The code that works behind this feature is shown in Fig (1.15) and Fig (1.16).



Fig(1.14)

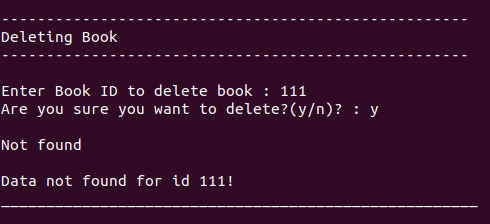


Fig(1.15)

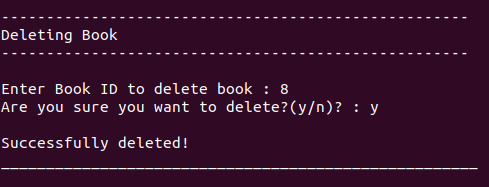


Fig(1.16)

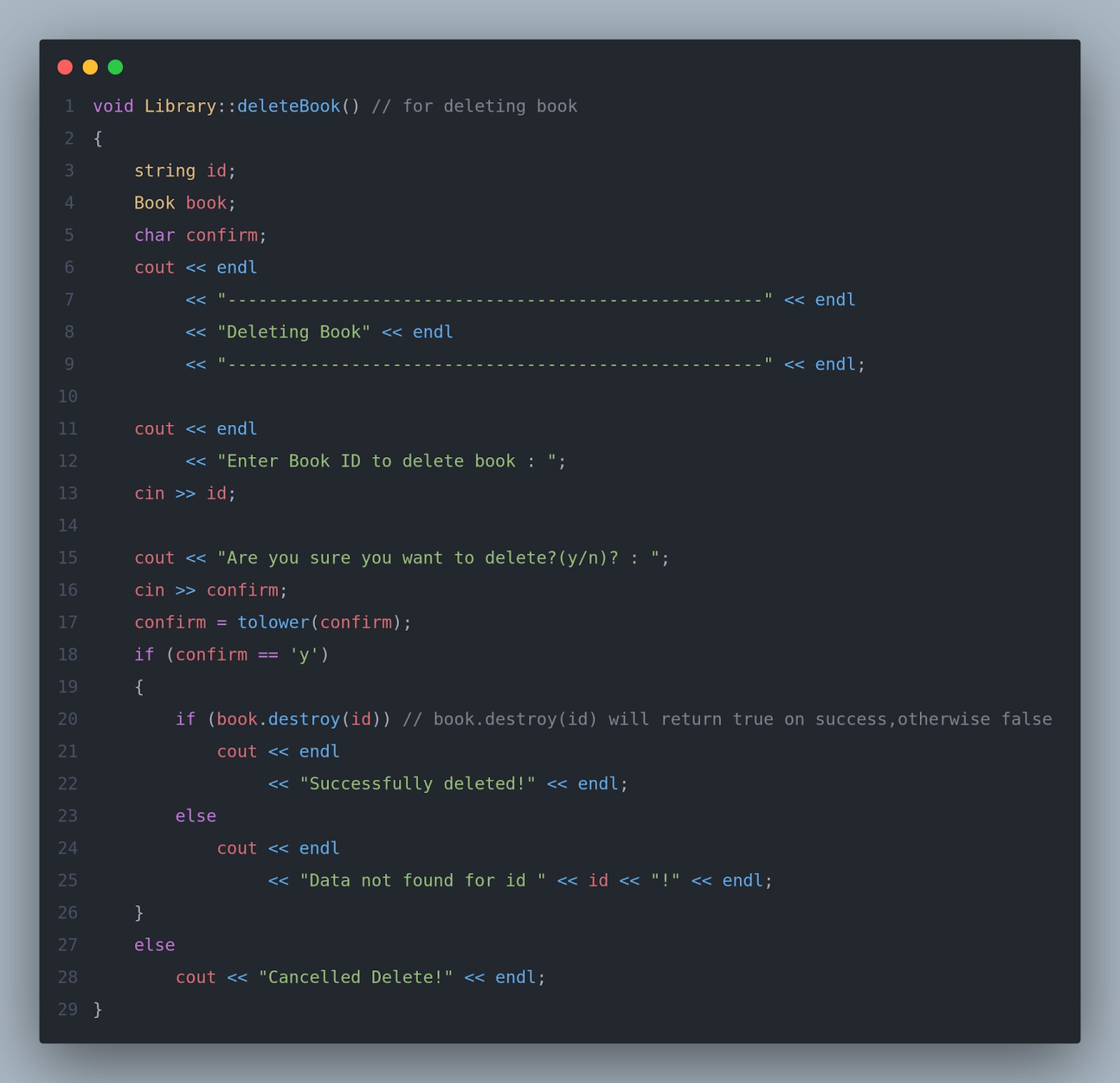
Number 4 is for deleting a book using its id. Firstly, we need to provide the id of the book that we want to delete. So, the program will find the book using the id that we provided. The program will also print the confirmation message which asks the user whether he really wants to delete. If we enter the id that does not exist in the library, the program will say a ‘Not Found’ message. Otherwise, it will be successfully deleted. For this feature, Fig (1.17), Fig (1.18), Fig(1.19), and Fig (1.20) are shown below.



Fig(1.17)



Fig(1.18)



Fig(1.19)

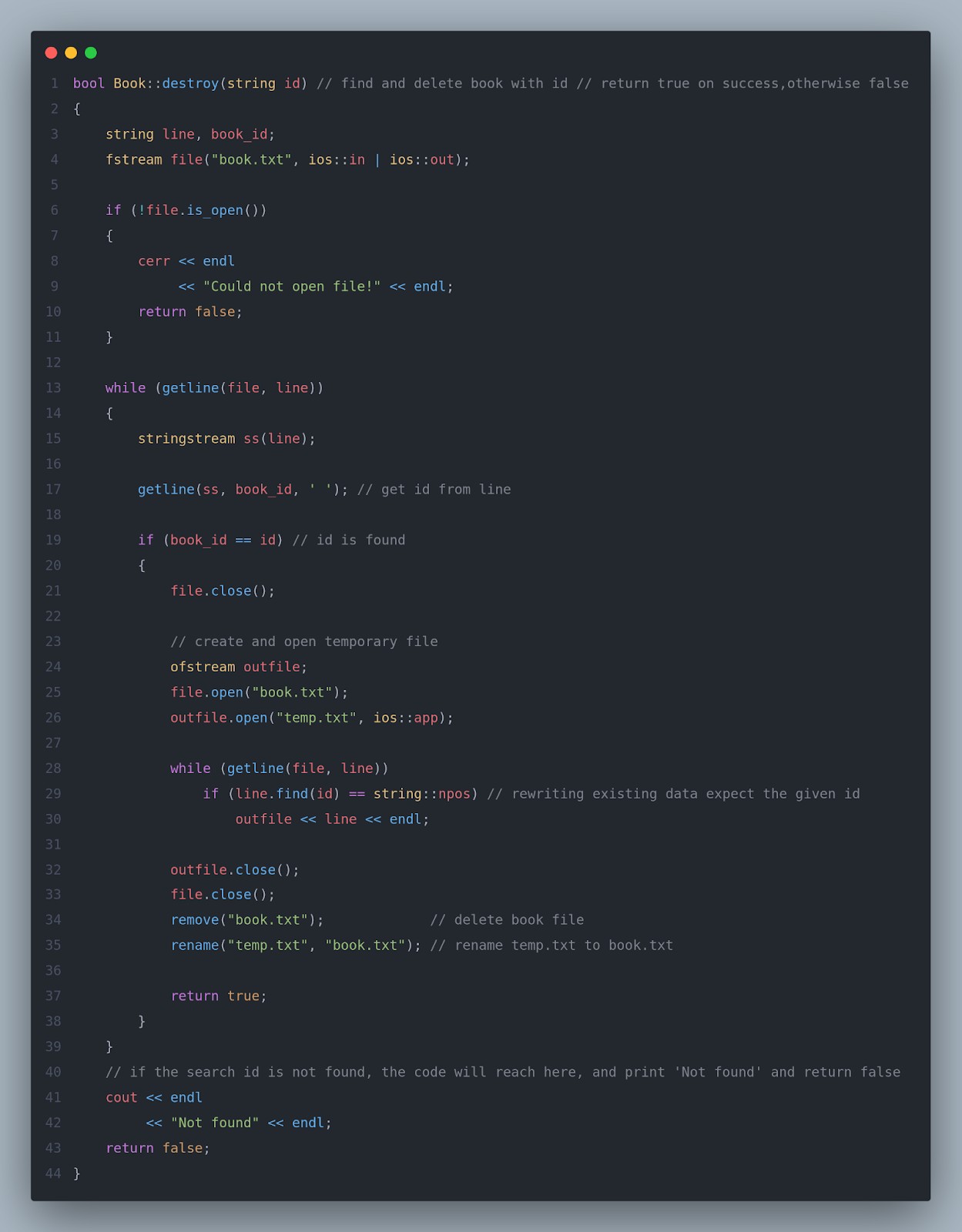


Fig (1.20)

To exit the program, choose number 5. The program will print the goodbye message to the user with his name which was provided at the start of the program as shown in Fig (1.21).

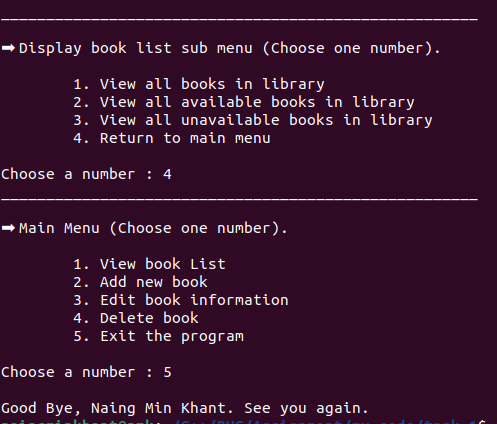


Fig (1.21)

The code behind this feature is shown in Fig (1.22).

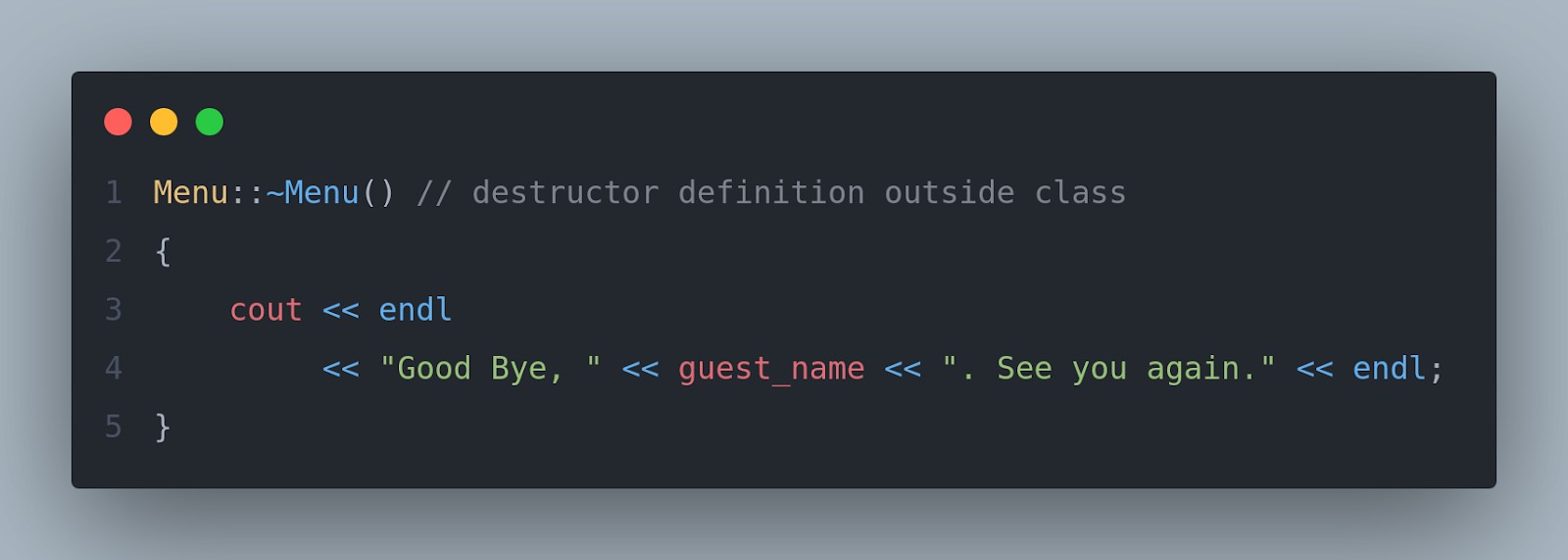


Fig (1.22)

# Task 2 - Inheritance

One class cannot access the properties and methods of another class without a relationship, although it can be done by instantiating an object of another. In this case, Inheritance can be used as a relationship between classes. Inheritance is one of the most important features of Object-Oriented Programming. It is just like a Parent-Child relationship which means Child has the ability to derive the parent's possession. By implementing Inheritance, the derived class(child class) will have the ability to access the properties of the base class (parent class). However, the members of the base class can be controlled by the access specifiers and will not be affected by the derived class. The derived class now is said to be inherited from the base class.

There are basically three types of Inheritance in C++, which are Single Inheritance, Multi-Level Inheritance, and Multiple Inheritance. If a derived class has only one base class(parent class), it can be called Single Inheritance. One derived class can inherit from more than one class, and it is known as Multiple Inheritance. If a derived class is created from another derived class, it is said to be Multi-Level Inheritance.

I implement the Single Inheritance in the program. Implementation of Single Inheritance in the program can be seen in Fig (2.1) and Fig (2.2).

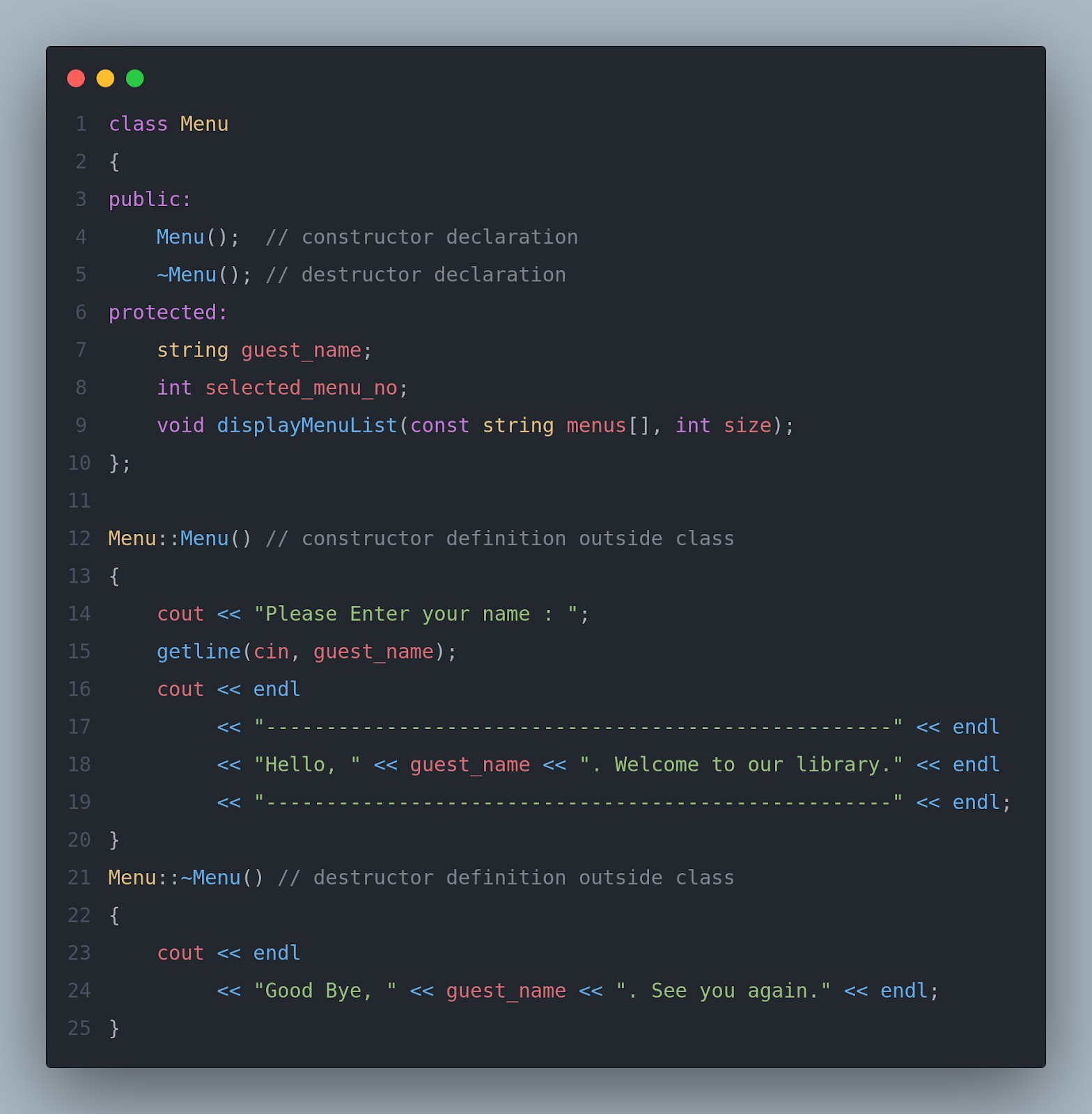


Fig (2.1)

In this figure, the Menu class, which has a constructor, destructor, and three protected members, is the base class (parent class).



Fig (2.2)

In Fig (2.2), the Library class is said to be inherited publically from the Menu class (the base class). The Library class can, now, have access to manipulate the protected members of the Menu class.

# Task 3 - Encapsulation

Encapsulation is one of the features of Object-Oriented Programming. Encapsulation can be defined as setting the data of a class as private properties and manipulating them by using class functions. Those class functions can only be accessed by an object of its class and must use only member variables. If there is no function inside the class that is using the member variable of the class, it cannot be called encapsulation. By using encapsulation, the security of class data can be increased and it also helps to control the modification of class data members.

Implementation of Encapsulation, which is used in the program, is shown in Fig (3.1) and Fig (3.2).

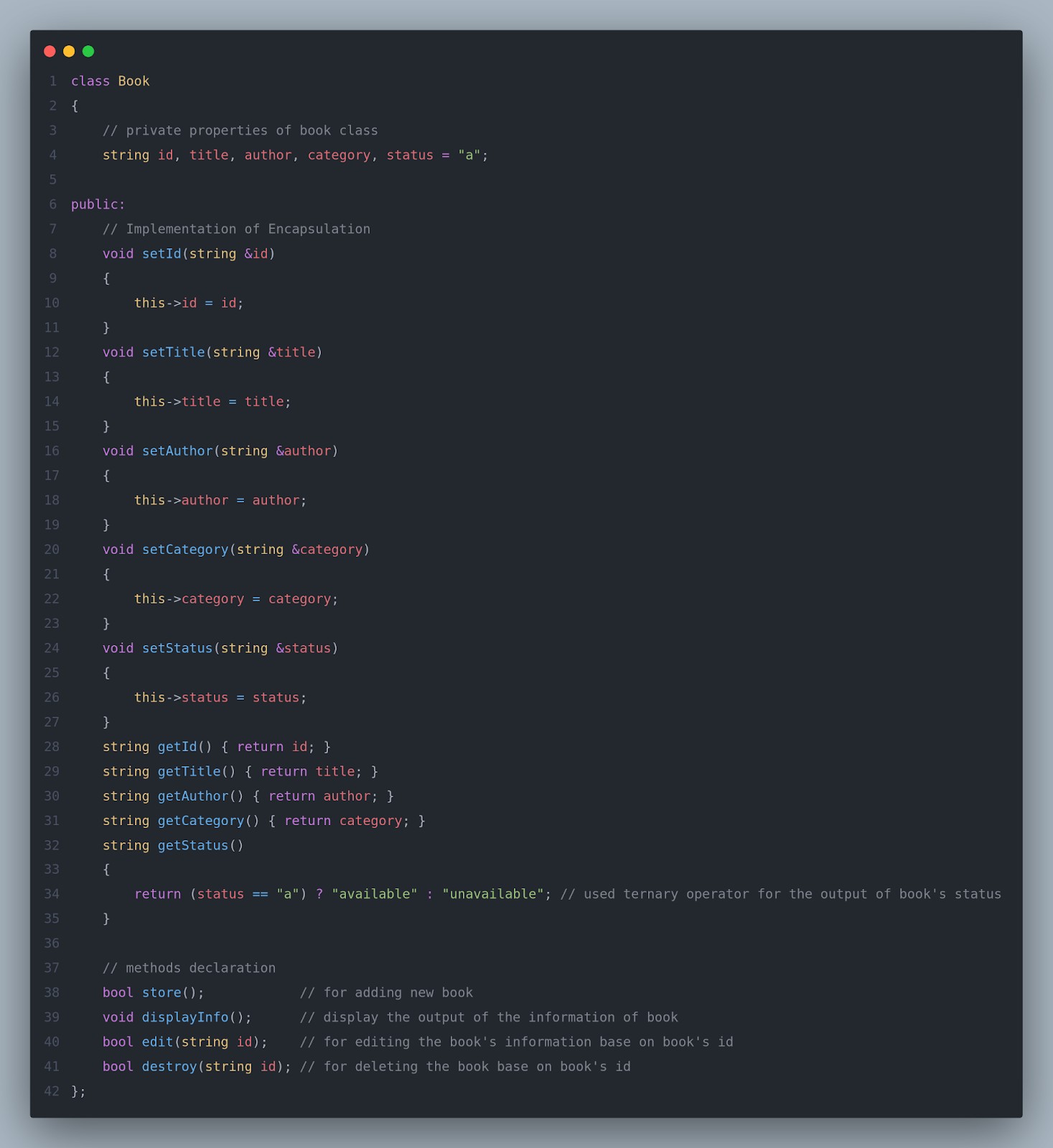


Fig (3.1)

The Book class has the private properties and those can only be manipulated by the getters and setters functions of the class.

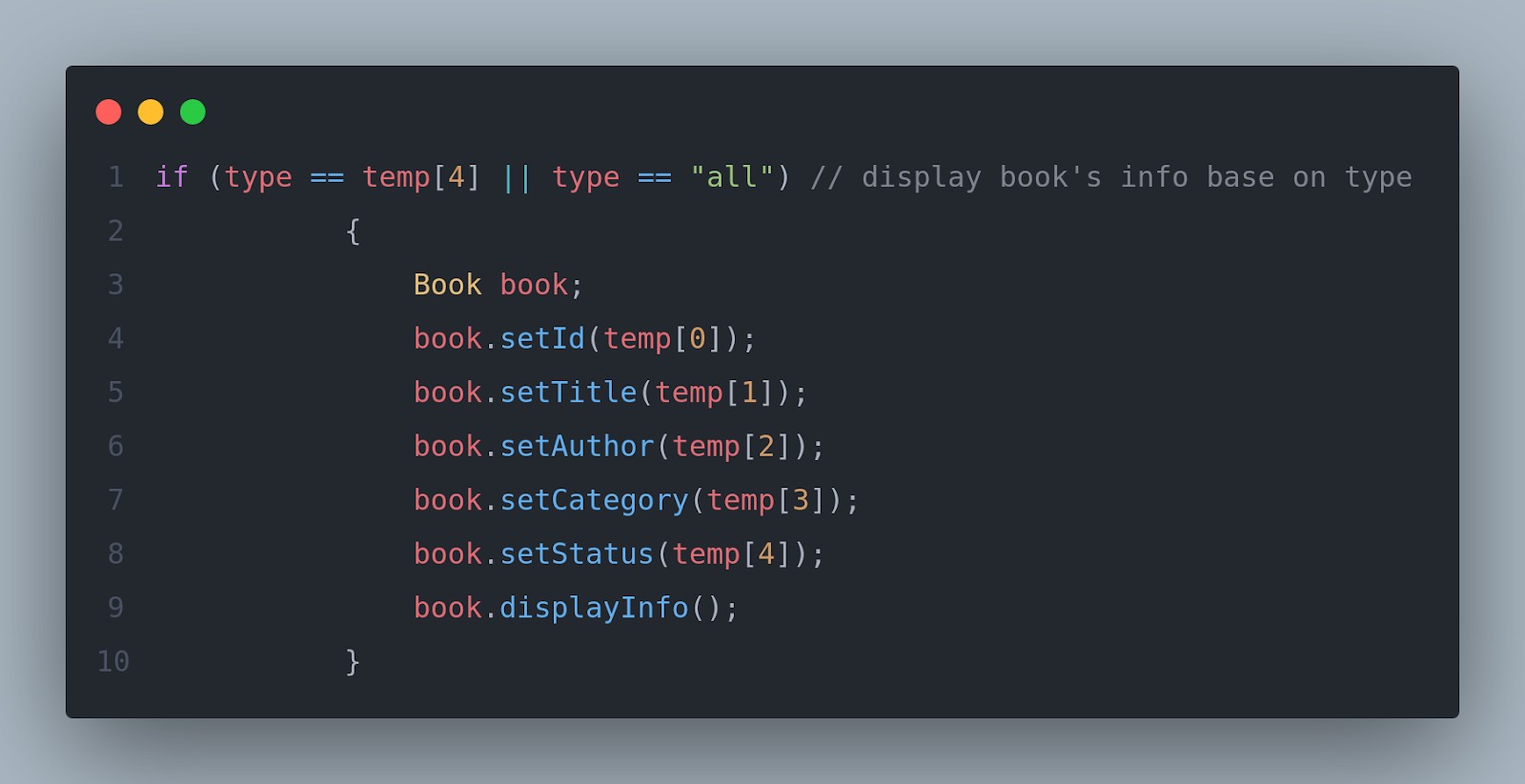


Fig (3.2)

Fig (3.2) shows the private members of the Book class are being set by the setters methods of its class.