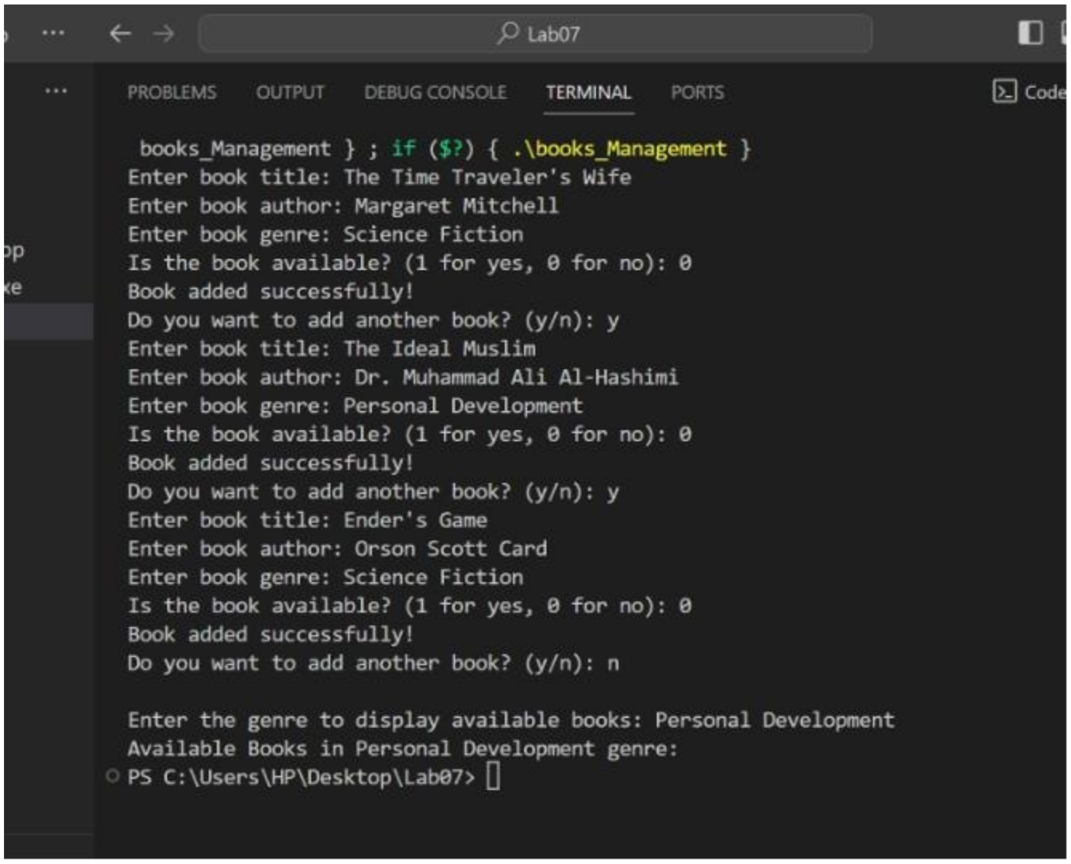
COURSE: (CL-1004) OBJECT ORIENTED PROGRAMMING LAB

LAB Assignment

Q No.1:

Scenario 1: Library Management System

1. Design Book Class:
   Create a class named `Book` with the following properties:
      `title`: Title of the book.
      `author`: Author of the book.
      `genre`: Genre of the book.
      `availability_status`: Indicates whether the book is available or not.
   Implement appropriate getter and setter methods for each property to ensure encapsulation.
2. Implement Library Class:
   Create a class named `Library` to manage the collection of books.
   Define a dynamic array of `Book` pointers to store the books.
   Include an integer variable to keep track of the current number of books in the library.
   Implement a constructor to initialize the array and the count of books.
   Implement a destructor to deallocate memory when the library object is destroyed.
3. Add Book Functionality:
   Implement a member function in the `Library` class to add a book to the library's collection.
   This function should take input parameters for the title, author, genre, and availability status of the book.
   Allocate memory for a new `Book` object and add it to the array of books.
4. Display Available Books by Genre:
   Implement a member function in the `Library` class to display available books of a specific genre.
   Prompt the user to enter a genre.
   Iterate through the array of books, check the availability, and match the genre.
   Print details of the available books that match the specified genre.
5. User Interaction:
   In the `main()` function, create an instance of the `Library` class.
   Use a loop to allow users to add books to the library one by one.
   After adding books, prompt the user to enter a genre to display available books of that genre.
   Terminate the program when the user finishes interacting with the library system.
6. Memory Management:
   Ensure proper memory allocation when adding books to the library.
   Deallocate memory for book objects when they are removed or when the library system shuts down to prevent memory leaks.



```
 books_Management } ; if ($?) { .\books_Management }
Enter book title: The Time Traveler's Wife
Enter book author: Margaret Mitchell
Enter book genre: Science Fiction
Is the book available? (1 for yes, 0 for no): 0
Book added successfully!
Do you want to add another book? (y/n): y
Enter book title: The Ideal Muslim
Enter book author: Dr. Muhammad Ali Al-Hashimi
Enter book genre: Personal Development
Is the book available? (1 for yes, 0 for no): 0
Book added successfully!
Do you want to add another book? (y/n): y
Enter book title: Ender's Game
Enter book author: Orson Scott Card
Enter book genre: Science Fiction
Is the book available? (1 for yes, 0 for no): 0
Book added successfully!
Do you want to add another book? (y/n): n

Enter the genre to display available books: Personal Development
Available Books in Personal Development genre:
PS C:\Users\HP\Desktop\Lab07> 
```

Q No.2:

Statement: Food Item Calorie Calculator

Features:

1. Thae program should define a class named **FoodItem** that have attributes for name, calories, grams of fat, grams of carbohydrates, and grams of protein.
2. The class should provide different constructors to create FoodItem objects with different levels of detail (name only, name and calories, or all details).
3. Implement getter and setter methods for each attribute to control access and validate input (e.g., calories cannot be negative).
4. Include a **friend function** named calculateCalories that calculates the total calories based on the provided protein and carb content using the formula: calories = (4 * carbs) + (4 * proteins).
5. In main function that interacts with the user to:

a. Get user input for food item name, protein content, and carbohydrate content.

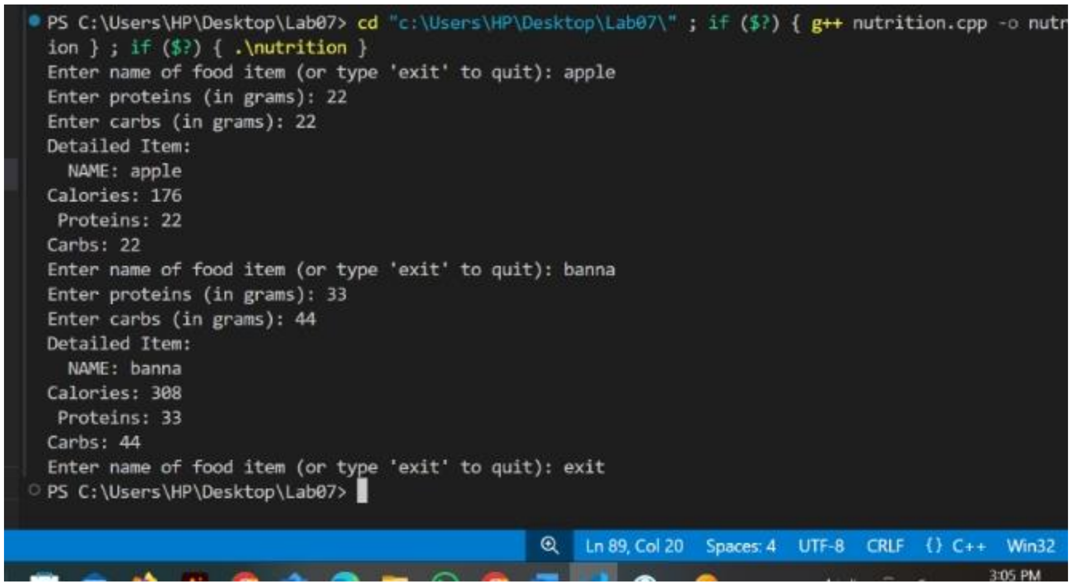
b. Display the detailed information of the food item including name, protein content, and carbohydrate content and calculated calories.

c. The program should continue prompting for input until the user enters "exit" to quit.

Bonus:

- Implement error handling for invalid user input (e.g., nonnumeric input for protein or carbs).
- Allow the user to set fat content as well and modify the calorie calculation formula to include fat (1 gram of fat = 9 calories).

Sample Run: You can add more statements for display on console.



```
PS C:\Users\HP\Desktop\Lab07> cd "c:\Users\HP\Desktop\Lab07\" ; if ($?) { g++ nutrition.cpp -o nutr
ion } ; if ($?) { .\nutrition }
Enter name of food item (or type 'exit' to quit): apple
Enter proteins (in grams): 22
Enter carbs (in grams): 22
Detailed Item:
  NAME: apple
Calories: 176
 Proteins: 22
Carbs: 22
Enter name of food item (or type 'exit' to quit): banna
Enter proteins (in grams): 33
Enter carbs (in grams): 44
Detailed Item:
  NAME: banna
Calories: 308
 Proteins: 33
Carbs: 44
Enter name of food item (or type 'exit' to quit): exit
PS C:\Users\HP\Desktop\Lab07>
```