Lab 2 – Stack/Queue Data Structure (2%)

Instructions: Must submit its executions before 11:00 pm, otherwise no grade will be awarded. Submit a link to the video demonstrating the execution of the program.

Problem: Create a Book class with the following features:

- Has fields/properties title, author, and number of pages
- Two books will be equal if the title and author are the same.
- Able to create book object with given title and author
- Able to create book objects with a given title, author, and number of pages.
- ToString () for string representation of Object

Develop a menu-driven application to maintain a collection of books using a **Stack** data structure:

- The user should be able to add a new book.
- The user should be able to display all the stored books.
- The user should be able to determine how many books are stored.
- Should be able to remove a book.
- Should be able to remove a given book i.e. given title of an author.

Guidelines:

We may have two files: 1) Program.cs or Main.cs file, 2) Class.cs file; Where inside the main file you will have a class, a main method, and inside this main method you will write the other methods to solve the lab problem.

At the very beginning, you have to create the stack datatype.

```
static Stack<Book> bookStack = new Stack<Book>();
```

We can have 6 methods in total inside program.cs:

1. **AddNewBook():**
   a. It will take the book title, author name, and page number as input; use Console.ReadLine().
   b. You have to set a logic for what will happen if the page number is less than 0
   c. Finally, you can create an instance of the Stack datatype using a new keyword. Then you can push the book name to the stack using the push method.
2. **DisPlayAllBooks():** Use the foreach loop to display all books, as I showed in the class.
3. **DisplayNumberOfBooks():** Use Count methods to display all books.
4. **RemoveBooks():** Use Remove methods to display all books

5. **RemoveBooksByTitleAndAuthor:** It will take input for the book name or author name, find the name in the Stack, and will use Pop methods to remove the Book.

Inside the class.cs file you will keep all the get/set methods and constructors.

For example:

```
public string Title { get; set; }
```

```
  public Book(string title, string author, int numberOfPages)
  {
      Title = title;
      Author = author;
      NumberOfPages = numberOfPages;
  }
```