# Lab: Iterators and Comparators

## Library

Create a class **Book** which should have three public properties:

* **string Title**
* **int Year**
* **List<string> Authors**

Authors can be **anonymous, one or many**. A Book should have only **one** **constructor**.

Create a class **Library** which should store a collection of books and implement the **IEnumerable<Book>** interface.

* **List<Book> books**

A Library should be could be intilized without books or with any number of books and should have only **one** **constructor**.

### Examples

|  |
| --- |
| Program.cs |
| public static void Main()  {  Book bookOne = new Book("Animal Farm", 2003, "George Orwell");  Book bookTwo = new Book("The Documents in the Case", 2002, "Dorothy Sayers", "Robert Eustace");  Book bookThree = new Book("The Documents in the Case", 1930);  Library libraryOne = new Library();  Library libraryTwo = new Library(bookOne, bookTwo, bookThree);  } |

### Solution



## Library Iterator

Extend your solution from the prevoius task. Inside the Library class create a **nested class** **LibraryIterator** which should implement the **IEnumerator<Book>** interface. Try to implement the bodies of the inherited methods by yourself. You will need two more members:

* **List<Book> books**
* **int currentIndex**

Now you should be able to iterate through a Library in the Main method.

### Examples

|  |
| --- |
| Program.cs |
| public static void Main()  {  Book bookOne = new Book("Animal Farm", 2003, "George Orwell");  Book bookTwo = new Book("The Documents in the Case", 2002, "Dorothy Sayers", "Robert Eustace");  Book bookThree = new Book("The Documents in the Case", 1930);  Library libraryOne = new Library();  Library libraryTwo = new Library(bookOne, bookTwo, bookThree);  foreach (var book in libraryTwo)  {  Console.WriteLine(book.Title);  }  } |

|  |
| --- |
| **Output** |
| Animal Farm  The Documents in the Case  The Documents in the Case |

### Solution

thisI