CMM024 Object Oriented Programming

Practical Session: 7

This lab is dedicated to Object Orientated Programming. You are going to create your own classes and test them

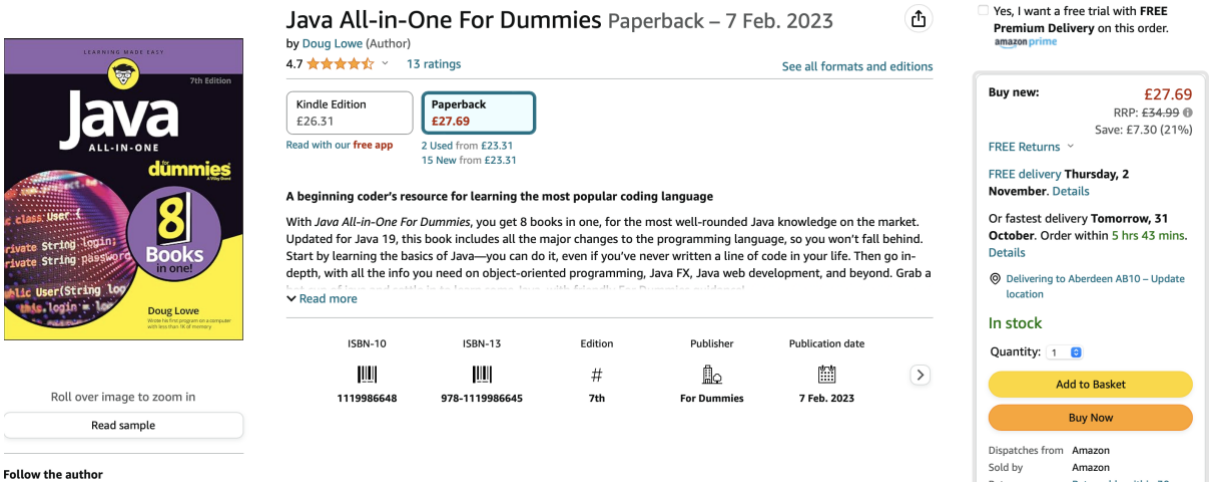
1. Book Class (worked Example)

Please read this example and implement the code. This is needed when coding the second part of this lab.

Brief:

This exercise is about modelling a book as a class.

Inspiration:



Attributes for Amazon Book Store

AS you can see from the Amazon Book store the following attributes are used. To keep it simple, we will not deal with the attributes in colour red.

- Title
- Author
- Price
- ISDN
- Edition number
- ~~Publisher~~
- ~~Publication Date~~
- ~~Edition Type~~
- ~~Abstract~~
- Discount

Attributes Datatype mapping

Since we are modelling a Book in Java, we need to map each attribute with a suitable data type.

Attribute	Data Type
Title	String
Author	String
Price	double
ISDN	String
Edition number	int
~~Publisher~~	~~String~~
~~Publication Date~~	~~String (or Date)~~
~~Edition Type~~	~~String~~
~~Abstract~~	~~String~~
Discount	double

UML modelling the Book class attributes (Word)

WE can use Word to create a UML model for the Book class.

+Book
-title: String -author: String -price: double -ISDN: String -editionNo:int -~~publisher: String~~ ~~-publicationDate: String~~ ~~-editionType: String~~ ~~-abstract: String~~ -discount: double

UML: modelling the Book class attributes (DrawIO)

Using DrawIO is much better though

+Book
- title: String - author: String - ISDN: String - price: double - discount: double - editionNo: int

UML: Creating a Constructor for the Book class

To create an Object instance from the Book class, we must create a constructor method that will pass enough information (in the parameters) to initialise all the attributes that must be set.

+Book
- title: String - author: String - ISDN: String - price: double - discount: double - editionNo: int
+ Book(title: String, author: String, ISDN: string, price: double, discount: double, editionNo: int)

Creating Setter Methods

Create setter method allow us to set or change the value of the attributes. Imagine that a mistake was made, how do you amend the existing information.

Attributes	Return type	Visibility	Method name	Method signature
-title: String	void	+	setTitle	+ void setTitle(String: Title)
-author: String	void	+	setAuthor	+ void setAuthor(String author)
-price: double	void	+	setPrice	+ void setPrice(double price)
-ISDN: String	void	+	setISDN	+ void setISDN(String ISDN)
-editionNo: int	void	+	setEditionNo	+ void setEditionNo(int editionNo)
-discount: double	void	+	setDiscount	+ void setDiscount(double discount)

Creating Getter Methods

All the attributes are private, which means that are invisible to other objects etc. However, we need to get access to these values, so we create public methods that return these values.

Attributes	Return type	Visibility	Method name	Method signature
-title: String	String	+	getTitle	+ String getTitle()
-author: String	String	+	getAuthor	+ String getAuthor()
-price: double	double	+	getPrice	+ double getPrice()
-ISDN: String	String	+	getISDN	+ String setISDN()
-editionNo: int	int	+	getEditionNo	+ int getEditionNo()
-discount: double	double	+	getDiscount	+ double getDiscount()

Creating Transformer Methods

There are times that you want to perform specific operations to some of the attributes. For instance, here, we may want to change the value of the book. If the percent rate is negative is negative, the price will therefore be less than original price; if positive the price will increase according to that rate. Ditto, for the Edition Number. It can increase or decrease depending on what is currently available for sale.

Attributes	Return type	Visibility	Method name	Method signature
-price: double	void	+	increasePrice	+ void changePrice(double PercentRate)
-editionNo: int	void	+	increaseEditionNo	+ void increaseEditionNo ()
-editionNo: int	void	+	decreaseEditionNo	+ void decreaseEditionNo ()

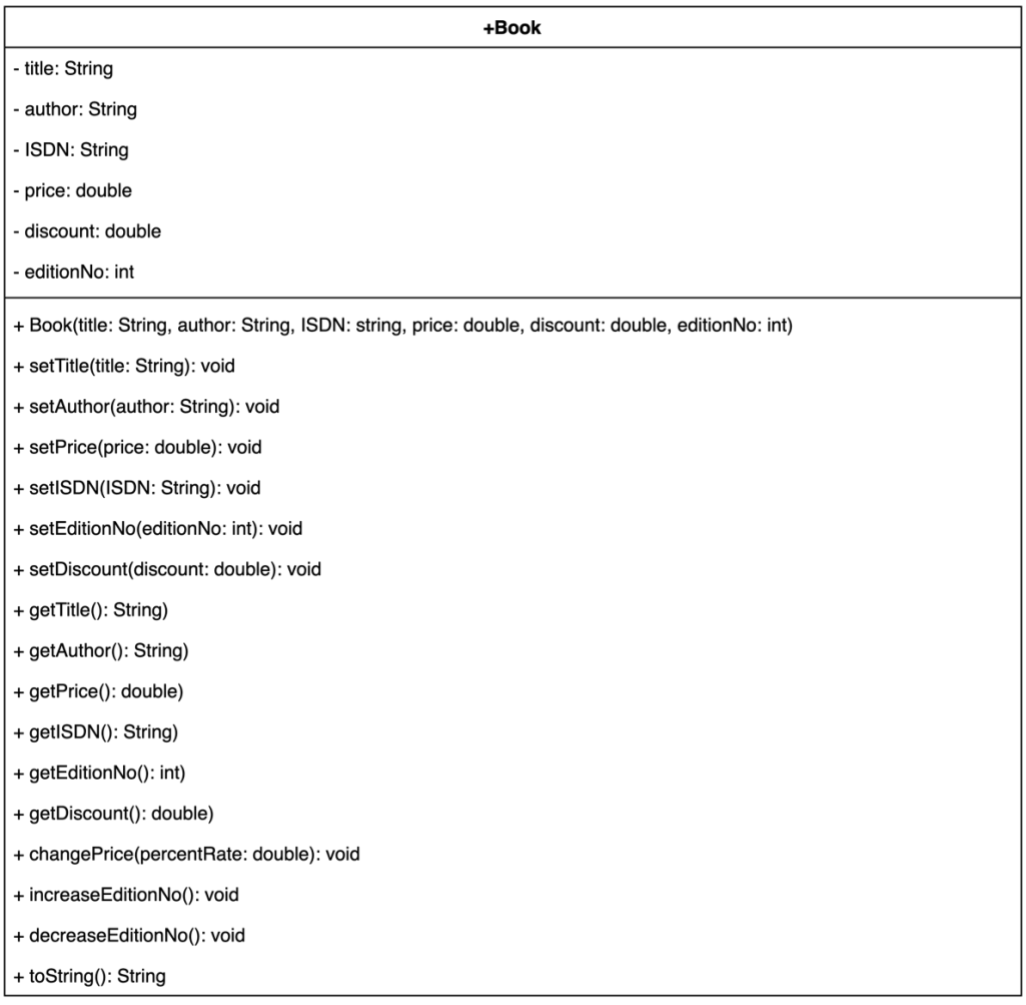
Creating toString() method

Often, we need to return the data information held in the attributes into a formatted string. This is also very useful for checking if the coding is right….when testing!

Attributes	Return type	Visibility	Method name	Method signature
NA	String	+	toString	+ String toString()

UML: modelling the Book class functionality (DrawIO)

In terms of UML, this is the full UML diagram for the Book class. By perusing it, you can code the skeleton of that class.



Coding the Book Class Attributes

We first create a Book class, and then we declared all the object attributes we need. We thought about what data type to use in an early process.

1) Create a **Book.java** java file

```
Book.java > ...
1   public class Book {
2
3       private String title;
4       private String author;
5       private String ISDN;
6       private double price;
7       private double discount;
8       private int editionNo;
9
10
11  }
```

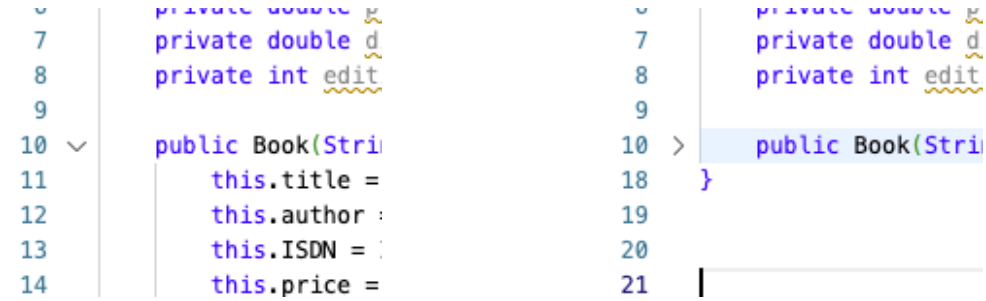
Coding the Book Class constructor

As we already said above, the constructor is a special method to create an object for a particular class, here it is the Book class. Since all the attributes are necessary and need to be set, we must use parameter that will hold the values to initialise them.
Note: to differentiate between the parameter name and the object attribute name we use THIS. Which refers to the class attribute rather the parameters. You can use different names for the parameters. For example: the parameter String title can be String theTitle, thus avoiding issues etc.

```
Book.java > ...
1   public class Book {
2
3       private String title;
4       private String author;
5       private String ISDN;
6       private double price;
7       private double discount;
8       private int editionNo;
9
10      public Book(String title, String author, String ISDN, double price, double discount, int editionNo){
11          this.title = title;
12          this.author = author;
13          this.ISDN = ISDN;
14          this.price = price;
15          this.discount = discount;
16          this.editionNo = editionNo;
17      }
18
19
20  }
```

A little trick with Visual Code

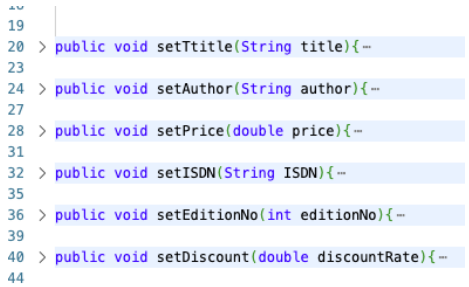
If you hover the method name, next to the line number, a small arrow will show up. If you click on it, all the code for this method will hide, only leaving the method signature. Very useful! Notice the line turns light blue when the code is hidden



Coding the Book Class Setter Methods

Setter methods have a parameter whose values is used to set the object attribute. i.e., the value in the parameter is used to set the value of the corresponding attribute. Of course the data type for the parameter must be the same as the one used for the object attribute!

```
19
20  public void setTtitle(String title){
21      this.title = title;
22  }
23
24  public void setAuthor(String author){
25      this.author = author;
26  }
27
28  public void setPrice(double price){
29      this.price = price;
30  }
31
32  public void setISDN(String ISDN){
33      this.ISDN = ISDN;
34  }
35
36  public void setEditionNo(int editionNo){
37      this.editionNo = editionNo;
38  }
39
40  public void setDiscount(double discountRate){
41      this.discount += discountRate;
42  }
```



Coding the Book Class Getter Methods

There will be no point of modelling something if there are no mechanisms to use the information and data in that object. Getter methods are returning the attributes' values.

```
47    public String getTitle(){
48        return this.title;
49    }
50
51    public String getAuthor(){
52        return this.author;
53    }
54
55    public double getPrice(){
56        return this.price;
57    }
58    public String getISDN(){
59        return this.ISDN;
60    }
61
62    public double getEditionNo(){
63        return this.editionNo;
64    }
65
66    public double getDiscount(){
67        return this.discount;
68    }
```

```
43
44  > public String getTitle(){…
47
48  > public String getAuthor(){…
51
52  > public double getPrice(){…
55
56  > public String getISDN(){…
59
60  > public double getEditionNo(){…
63
64  > public double getDiscount(){…
67
```

Coding the Book Class Transformer Methods

As we already mentioned, we can also have specific method to do specific tasks on the data.

```
71    public void changePrice(double discountRate){
72        double discountValue = this.discount * discountRate
73        this.discount += discountValue;
74    }
75
76    public void increaseEditionNo(){
77        this.editionNo++;
78    }
79
80    public void decreaseEditionNo(){
81        this.editionNo--;
82    }
```

```
68  > public void changePrice(double discountRate){
72
73  > public void increaseEditionNo(){…
76
77  > public void decreaseEditionNo(){…
80
```

Coding the Book Class toString() Method

The central piece is a String where the value of the attributes are appended to it. WE have used this before in other labs! Note the calculation done to calculate the retail value based on the initial price but taking into account the discount.

```java
public String toString() {

    String st = "\n";
    st += "Book title: ";
    st += this.title;

    st += " - Author: ";
    st += this.author;
    st += "\n";

    st += "ISDN: ";
    st += this.ISDN;
    st += " - Edition Number: ";
    st += this.editionNo;
    st += "\n";

    st += "Full Price: ";
    st += this.price;

    st += " - Discount: ";
    st += this.discount + "%";

    st += " - Retail Price: ";
    double retailPrice = this.price - (this.price * discount / 100);
    st += retailPrice;
    st += "\n";

    return st;
}
```

Testing the Book Class using its main(…) Method

We can add an extra static main method to test the class you have just coded. Note the keyword new. You have used it using the Scanner class in earlier labs!

```java
Run | Debug
public static void main(String [] args){

    Book myBook = new Book(title:"Java All-in-One For Dummies",
        author:"Doug Lowe", ISDN:"1119986648", price:34.99, discount:21,editionNo:7);
    System.out.println(myBook);

}
```

Creating a TestBook class to test the Book Class

The best way is to create a dedicated class to test classes.

1) Create a **TestBook.java** java file

```java
TestBook.java > ...
public class TestBook {

    Run | Debug
    public static void main(String[] args){

    }

}
```

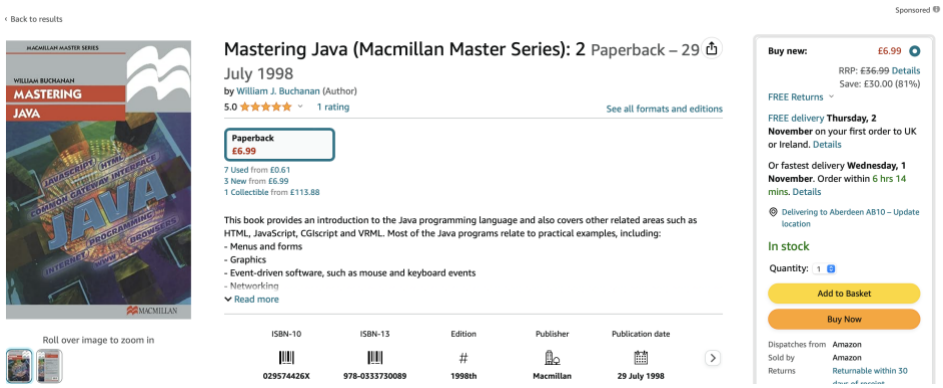
Book	myBook
Title	Java All-in-One For Dummies
Author	Doug Lowe
ISDN	1119986648
Price	34.99
Discount	21%
Edition Number	7

```java
public class TestBook {

    Run | Debug
    public static void main(String[] args) {
        Book myBook = new Book(title:"Java All-in-One For Dummies",
                author:"Doug Lowe", ISDN:"1119986648",
                    price:34.99, discount:21, editionNo:7);
        System.out.println(myBook);

    }

}
```

2) Run the **TestBook** class

```
<eStorage/88fe9f75fb7bccca48f8f3f3fd084b4d/redhat.java/jdt_ws/S

Book title: Java All-in-One For Dummies - Author: Doug Lowe
ISDN: 1119986648 - Edition Number: 7
Full Price: 34.99 - Discount: 21.0% - Retail Price: 27.6421

jean-claude@MacBook-Pro-2 SupportWeek %
```

3) Create another book in the code



Book	myBook1	myBook2
Title	Java All-in-One For Dummies	Mastering Java
Author	Doug Lowe	William Buchanan
ISDN	1119986648	029574426X

Price	34.99	6.99
Discount	21%	81%
Edition Number	7	1998

4) Create an ArrayList of Book and name it store

5) Add both books to the store

6) Display details of the books in the store

```
Method1.java  Loan1_v2.java  temp1.java  Person.java
TestBook.java > ...
1   import java.util.ArrayList;
2
3   public class TestBook {
4
        Run | Debug
5       public static void main(String[] args) {
6           Book myBook1 = new Book(title:"Java All-in-One For Dummies",
7                   author:"Doug Lowe", ISDN:"1119986648",
8                   price:34.99, discount:21, editionNo:7);
9
10          Book myBook2 = new Book(title:"Mastering Java",
11                  author:"William Buchanan", ISDN:"029574426X",
12                  price:36.99, discount:81, editionNo:1998);
13
14          ArrayList<Book> store = new ArrayList<>();
15          store.add(myBook1);
16          store.add(myBook2);
17
18          for (Book aBook: store){
19              System.out.println(aBook.toString());
20          }
21      }
22  }
23
```

```
e0e/bin TestBook

Book title: Java All-in-One For Dummies - Author: Doug Lowe
ISDN: 1119986648 - Edition Number: 7
Full Price: 34.99 - Discount: 21.0% - Retail Price: 27.6421


Book title: Mastering Java - Author: William Buchanan
ISDN: 029574426X - Edition Number: 1998
Full Price: 36.99 - Discount: 81.0% - Retail Price: 7.028100000000002

jean-claude@MacBook-Pro-2 SupportWeek %
```

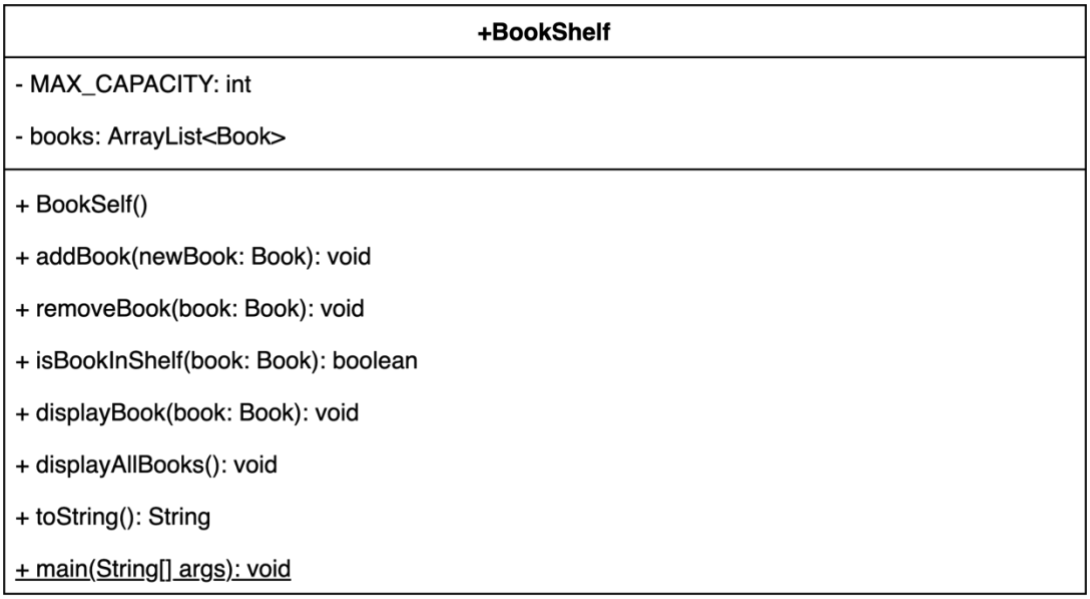
2. BookShelf Class

Brief:

This exercise is about modelling a bookshelf class to store our books.

Implementing the BookShelf class

Implement the BookShelf class following the UML Diagram below.



BookShelf Coding note:

- The ArrayList books is declared but not created. Create the books ArrayList in the constructor.
- The addBook method uses the ArrayList add method (books.add(newBook);)
- The removeBook method uses the ArrayList remove method
- The isBookInShelf method uses the ArrayList contains method. This method returns a boolean value which can be returned
- The displayBook method should first get the index of the book passed as a parameter, using the indexOf method of the ArrayList class. Then you should retrieve the book referred to by the index using the ArrayList get method. Finally use the println method to display the book details (using the toString method)
- The displayAllBooks method uses a for loop to iterate all the books stored in the books ArrayList and use the println method to display the details for the books

Implementing the BookShelf class main method

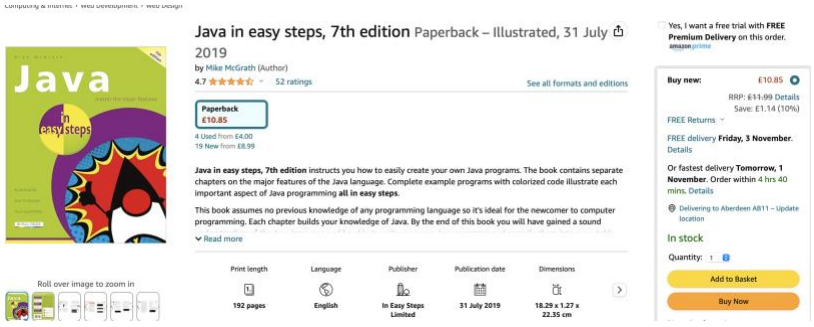
- Create a BookShelf object instance (using the new etc). Name it bookShelf
- Create 4 different books using the information below. You can use the one created in the Book class. Use myBook1, myBook2, myBook3, myBook4
- Add the 4 books to the bookshelf
- Invoke the displayAllBooks method to display the details for all the books

```
Book title: Java All-in-One For Dummies – Author: Doug Lowe
ISDN: 1119986648 – Edition Number: 7
Full Price: 34.99 – Discount: 21.0% – Retail Price: 27.6421


Book title: Mastering Java – Author: William Buchanan
ISDN: NA – Edition Number: 1998
Full Price: 36.99 – Discount: 81.0% – Retail Price: 7.028100000000002


Book title: Java in easy steps – Author: Mike McGrath
ISDN: 029574426X – Edition Number: 7
Full Price: 10.99 – Discount: 10.0% – Retail Price: 9.891


Book title: Effective Java – Author: Joshua Bloch
ISDN: 0134685997 – Edition Number: 3
Full Price: 40.49 – Discount: 6.0% – Retail Price: 38.0606
```

- Using the println method to display the bookShelf details

```
Full Price: 40.49 – Discount: 6.

Bookshelf max capacity: 12
Number books on the shelf: 4
```

Books information to use (copy and paste)



Book	myBook1	myBook2
Title	Java All-in-One For Dummies	Mastering Java
Author	Doug Lowe	William Buchanan
ISDN	1119986648	029574426X
Price	34.99	6.99
Discount	21%	81%
Edition Number	7	1998

Book	myBook3	myBook4
Title	Java in easy steps	Effective Java
Author	Mike McGrath	Joshua Bloch
ISDN	NA	0134685997
Price	10.99	40.49
Discount	10%	6%
Edition Number	7	3

Enhancing the BookShelf class functionality

As it stands, we can add the same book 12 times is one wished as we simply add them to the ArrayList without any checks. However, we should not be able to do this. Thus, we should check if the book is already stored in the list before adding it.
We use the indexOf method used in the displayBook method already implemented. Copy and paste this line of code i.e., the one using the indexOf
Note: the index method returns -1 if the object does not exist. Therefore, we can use an if statement to check if the value returned is -1 or not. If it is -1 then the book is not in the store so we can add it. Otherwise, we can display a message that this book is already in the store.
To test it you can add the same book in the main method twice in the bookshelf.

```
bookShelf.addBook(myBook1);
bookShelf.addBook(myBook2);
bookShelf.addBook(myBook3);
bookShelf.addBook(myBook4);
bookShelf.displayAllBooks();

bookShelf.addBook(myBook1);
```

```
Book title: Effective Java – Author: Joshua Bloch
ISDN: 0134685997 – Edition Number: 3
Full Price: 40.49 – Discount: 6.0% – Retail Price: 38.0606

The book: Java All-in-One For Dummies is aready in the bookshelf

Bookshelf max capacity: 12
Number books on the shelf: 4
```