

Assignment: OOP

IMPORTANT: This assignment must be done individually.

Read Section A to understand the programming requirements, Section B to understand the programming tasks that you need to carry out. Section C provides a full screenshot of outputs and Section D describes what you need to submit as the result.

A. Description

In this assignment, you will create a menu-driven console program to manage books in a library named “**BookMan**”.

```
-----  
1. list all books  
2. add a new book  
3. edit book  
4. delete a book  
5. search books by name  
6. sort books descending by price  
  
0. save & exit  
-----  
Your option:
```

1. Feature: Load books from file

In a fixed-length format file, you are provided with Book details in “**books.txt**”. Whenever starting the program, it reads the file then parses the data for Book objects and add to the list of books. The length and position of attributes are as follows.

Attribute	Length	Position
Id	5	1
Name	45	6
Price	10	51

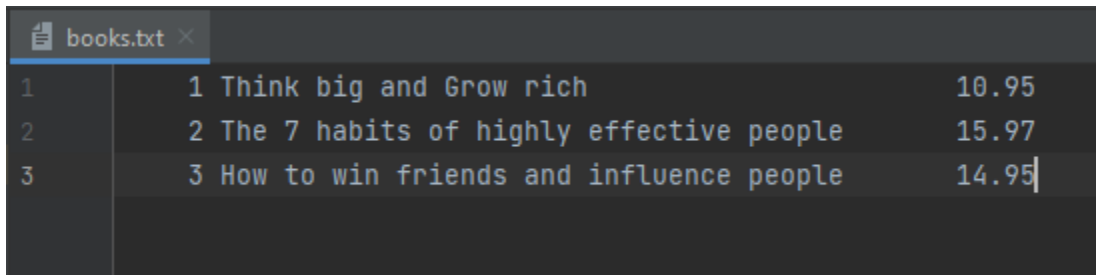
Input format:

Read the details from the input file “**books.txt**”

Output format:

Print out message

Sample input:



A screenshot of a text editor window titled 'books.txt'. The window contains three lines of text, each representing a book entry. The first line is '1 Think big and Grow rich 10.95', the second is '2 The 7 habits of highly effective people 15.97', and the third is '3 How to win friends and influence people 14.95'. The text is displayed in a monospaced font on a dark background.

1	1 Think big and Grow rich	10.95
2	2 The 7 habits of highly effective people	15.97
3	3 How to win friends and influence people	14.95

Sample output:

Loading books...

2. Feature: List all books

Display a list of all books in a **tabular form** in the console. If no books, print out “(*empty*)”.

Output format:

- Use "%-5s %-45s %-10s" for the heading
- Use "%5d %-45s %10.2f" for displaying Book details
- Print 2 digits after the decimal for the double datatype
- Refer to the sample output for the formatting specification

Sample output:

- Case: not empty

ID	Name	Price
1	Think big and Grow rich	10.95
2	The 7 habits of highly effective people	15.97
3	How to win friends and influence people	14.95

- Case: empty

(empty)

3. Feature: Add a new book

Get details from user to create a new Book object, then add it into the list of books.

Input format:

Get book details from user

Output format:

Notify user whether success or not.

Sample:

- Case: success

```
Enter book id: 5
Enter book name: Harry Potter
Enter book price: 15.4
Added successfully.
```

- Case: error

```
Enter book id: 1
Enter book name: Think big and Grow rich
Enter book price: 10.95
Duplicated ID!
```

4. Feature: Edit book

Ask user to enter book id to edit. If the Book object with entered id does not exist in the list of books, notify user “**Invalid ID!**”, otherwise get other details from user then update the Book object.

Input format:

Get book details from user

Output format:

Notify user whether success or not.

Sample:

- Case: success

```
Enter book id: 5
Enter book name: Harry Potter and the Deathly Hallows
Enter book price: 23
Updated successfully.
```

- Case: error

```
Enter book id: 6
Invalid ID!
```

5. Feature: Delete a book

Ask user to enter book id to edit. If the Book object with entered id does not exist in the list of books, notify user “**Invalid ID!**”, otherwise remove the Book object from the list of books.

Input format:

Get book id from user

Output format:

Notify user whether success or not.

Sample:

- Case: success

```
Enter book id: 5
Deleted successfully.
```

- Case: error

```
Enter book id: 6
Invalid ID!
```

6. Feature: Search books by name

Get keyword from user, filter Book objects in the list of books with name contains keyword (case insensitive), then print out in a **tabular form** in the console. If no books, print out “(empty)”.

(Similar to *Feature: List all books*)

Input format:

Get keyword from user

Output format:

- Use "%-5s %-45s %-10s" for the heading
- Use "%5d %-45s %10.2f" for displaying Book details
- Print 2 digits after the decimal for the double datatype
- Refer to the sample output for the formatting specification

Sample:

- Case: matches

Enter keyword: people		
ID	Name	Price
2	The 7 habits of highly effective people	15.97
3	How to win friends and influence people	14.95

- Case: no result

Enter keyword: love story (empty)		
---	--	--

7. Feature: Sort books by price

Sort books descending by price then print out in a **tabular form** in the console. If no books, print out “(empty)”.

(Similar to *Feature: List all books*)

Output format:

- Use "%-5s %-45s %-10s" for the heading
- Use "%5d %-45s %10.2f" for displaying Book details

- Print 2 digits after the decimal for the double datatype
- Refer to the sample output for the formatting specification

Sample output:

- Case: not empty

After sorting:		
ID	Name	Price
2	The 7 habits of highly effective people	15.97
3	How to win friends and influence people	14.95
1	Think big and Grow rich	10.95

- Case: empty

Enter keyword: love story (empty)

8. Feature: Save & exit

Save all books into file **books.txt** in required format, then end program. If no books, write nothing to the file.

The **books.txt** is a fixed-length format file. The length and position of attributes are as follows.

Attribute	Length	Position
Id	5	1
Name	45	6
Price	10	51

(See *Feature: Load books from file*)

Output format:

- Use "%5d %-45s %10.2f" for displaying Book details
- Print 2 digits after the decimal for the double datatype
- Refer to the sample output for the formatting specification

Sample output:

Saving to file... Bye!

```
books.txt x
1      2 The 7 habits of highly effective people      15.97
2      3 How to win friends and influence people      14.95
3      1 Think big and Grow rich                      10.95
4      |
```

B. Programing tasks

IMPORTANT: Strictly follow to the Object-Oriented specifications given in the problem statement. All class names, attribute names, and method names should be the same as specified in the problem statement.

Consider a **Book** class with the following attributes & methods

Attribute	Datatype
id	Integer
name	String
price	Double

Method	Description
public Book(int id, String name, double price)	This constructor is to create a new Book object from provided book details
public void setName(String name)	This setter updates this.name with provided value
public void setPrice(double price)	This setter updates this.price with provided value
public String toString()	This method returns string representation of this Book in the required format. Refer to the sample outputs in Section A for the formatting specification.

Consider the class **BookManager** and define the following attribute & methods

Attribute	Datatype
books	ArrayList<Book>

Method	Description
<code>public BookManager()</code>	This constructor initializes <code>this.books</code> as an empty list
<code>public ArrayList<Book> getBooks()</code>	This getter returns all books managed by this
<code>public void loadFromFile()</code>	This method reads the file books.txt , parses the data in the file to <code>Book</code> objects and adds them to <code>this.books</code> .
<code>public void printBooks(ArrayList<Book> books)</code>	This method accepts a list of <code>Book</code> objects as the argument and prints out the <code>Book</code> details in required format by iterating the list. If the input list <code>books</code> is empty, print “(empty)”
<code>public boolean add(Book book)</code>	This method accepts a <code>Book</code> object as an input argument and adds it to <code>this.books</code> if <code>book.id</code> is not duplicated, then returns whether added or not
<code>public Book getBookById(int id)</code>	This method accepts an id of <code>Book</code> as input and returns the <code>Book</code> object from <code>this.books</code> with the corresponding id.
<code>public void remove(Book book)</code>	This method accepts a <code>Book</code> object as the argument and remove it from <code>this.books</code> .
<code>public void sortDescByPrice()</code>	This method modifies <code>this.books</code> to be sorted in the descending order of price.
<code>public List<Book> searchByName(String keyword)</code>	This method accepts a keyword string as argument and returns the list of <code>Book</code> objects whose name contains the keyword (IMPORTANT : case-insensitive)
<code>public void saveToFile()</code>	This method writes <code>Book</code> objects managed by this into the file books.txt in the required format.

Consider a program class called **Main** to show the menu, get user inputs, invoke appropriate methods from other classes to carry out functionalities & print out suitable messages in the main method. This method also prints out “**Bye!**” when user exit the program.

IMPORTANT: Use `System.out` for errors

C. Screenshots

Loading books... ----- 1. list all books 2. add a new book 3. edit book 4. delete a book 5. search books by name
--

6. sort books descending by price

0. save & exit

Your option: 8

Invalid option!

1. list all books

2. add a new book

3. edit book

4. delete a book

5. search books by name

6. sort books descending by price

0. save & exit

Your option: 1

ID	Name	Price
2	The 7 habits of highly effective people	15.97
3	How to win friends and influence people	14.95
1	Think big and Grow rich	10.95

1. list all books

2. add a new book

3. edit book

4. delete a book

5. search books by name

6. sort books descending by price

0. save & exit

Your option: 2

Enter book id: 5

Enter book name: Harry Potter

Enter book price: 15.4

Added successfully.

1. list all books

2. add a new book

3. edit book

4. delete a book

- 5. search books by name
- 6. sort books descending by price

0. save & exit

Your option: 1

ID	Name	Price
2	The 7 habits of highly effective people	15.97
3	How to win friends and influence people	14.95
1	Think big and Grow rich	10.95
5	Harry Potter	15.40

- 1. list all books
- 2. add a new book
- 3. edit book
- 4. delete a book
- 5. search books by name
- 6. sort books descending by price

0. save & exit

Your option: 3

Enter book id: 5

Enter book name: Harry Potter and the Death Hallows

Enter book price: 23

Updated successfully.

- 1. list all books
- 2. add a new book
- 3. edit book
- 4. delete a book
- 5. search books by name
- 6. sort books descending by price

0. save & exit

Your option: 1

ID	Name	Price
2	The 7 habits of highly effective people	15.97
3	How to win friends and influence people	14.95
1	Think big and Grow rich	10.95

5 Harry Potter and the Death Hallows	23.00

1. list all books	
2. add a new book	
3. edit book	
4. delete a book	
5. search books by name	
6. sort books descending by price	
0. save & exit	

Your option: 4	
Enter book id: 5	
Deleted successfully!	

1. list all books	
2. add a new book	
3. edit book	
4. delete a book	
5. search books by name	
6. sort books descending by price	
0. save & exit	

Your option: 4	
Enter book id: 5	
Invalid ID!	

1. list all books	
2. add a new book	
3. edit book	
4. delete a book	
5. search books by name	
6. sort books descending by price	
0. save & exit	

Your option: 5	
Enter keyword: people	
ID	Price
2 The 7 habits of highly effective people	15.97

```

3 How to win friends and influence people 14.95
-----
1. list all books
2. add a new book
3. edit book
4. delete a book
5. search books by name
6. sort books descending by price

0. save & exit
-----
Your option: 5
Enter keyword: love story
(empty)
-----
1. list all books
2. add a new book
3. edit book
4. delete a book
5. search books by name
6. sort books descending by price

0. save & exit
-----
Your option: 6
After sorting:
ID      Name                                     Price
  2 The 7 habits of highly effective people    15.97
  3 How to win friends and influence people    14.95
  1 Think big and Grow rich                    10.95
-----
1. list all books
2. add a new book
3. edit book
4. delete a book
5. search books by name
6. sort books descending by price

0. save & exit
-----
Your option: 0

```

Saving to file...
Bye!

D. Submission

You must submit a single zip file containing the application to the portal by the due date. The zip file name must be of the form `a1_Sid.zip`, where *Sid* is your student identifier (the remaining bits of the file name must not be changed!). For example, if your student id is 2001040001 then your zip file must be named `a1_2001040001.zip`.

IMPORTANT: failure to name the file as shown will result in no marks being given!

NO PLAGIARISM: if plagiarism is detected, 0 mark will be given!