[redacted]

[redacted]

task Book Collection

Think of a use case for **list management** (e.g
Book collection, phone book, inventory list, task management, ...) with **at least four properties.** Users should be able to edit the entries in the list with the usual operations (insert, edit, delete, sort, search, etc.). See the evaluation criteria and the example on the following pages.

The application should be written in the **Java** programming language and be controlled via the **command line .** A menu structure should be used (see example below). Also, be sure to intercept and handle erroneous user input appropriately. The program must not crash if incorrect entries are made! Implement an appropriate **data structure** (without using the Java standard library) for the list and also implement the necessary **algorithms** for the common operations (without using the Java standard library).

Write a short documentation **as a PDF file**

1. Description of the use case 2. Installation instructions, 3. User instructions, 4. Description of the implemented data structure, 5. Analysis of the runtime complexity of the implemented sorting and search algorithms and
6. Attribution of all sources of documentation and source code

[redacted]

evaluation criteria

1. Functionality of the features (20 points)

List operations for adding and editing entries List operations for deleting individual	0 - 6 points
entries and the entire list Function for displaying the complete list and number of entries	0 - 2 points
Operation for sorting the list (at least three different properties are supported)	0 - 2 points
	0 - 4 points
List operation to search for entries (at least three different properties are supported)	0 - 4 points
min. an additional operation suitable for the application	0 - 2 points

2. Implementation (16 points)

Choosing the appropriate data structure and data types for the list and their Characteristics	0 - 4 points
Choosing the appropriate algorithms for the list operations (see underlined operations above)	0 - 6 points
Complexity of the sorting algorithm used (heap/radix sort > Quick/Merge Sort > Elementary Sorting Algorithms)	0 - 6 points

3. Code quality (6 points)

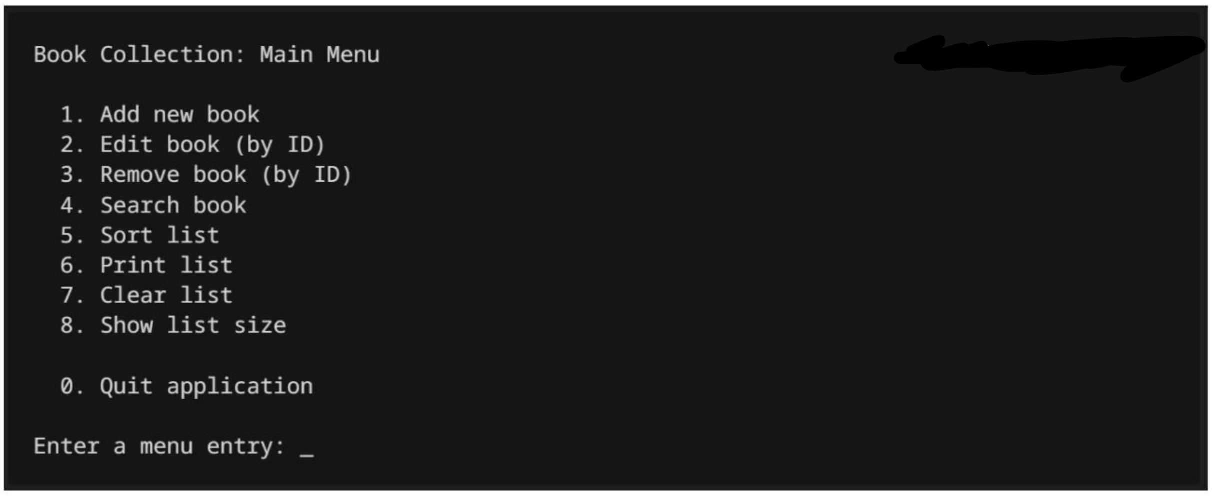
Meaningful folder structure, method and class naming, more consistent code style	0 - 4 points
Comments, parameter description	0 - 2 points

4. Quality of documentation (8 points)

Description of the use case, installation instructions, User guide, naming of sources	0 - 4 points
Description of the data structure and data types for the list	0 - 2 points
Analysis of the runtime complexity of the sorting and search algorithms	0 - 2 points

[redacted]

Example: list management for a book collection

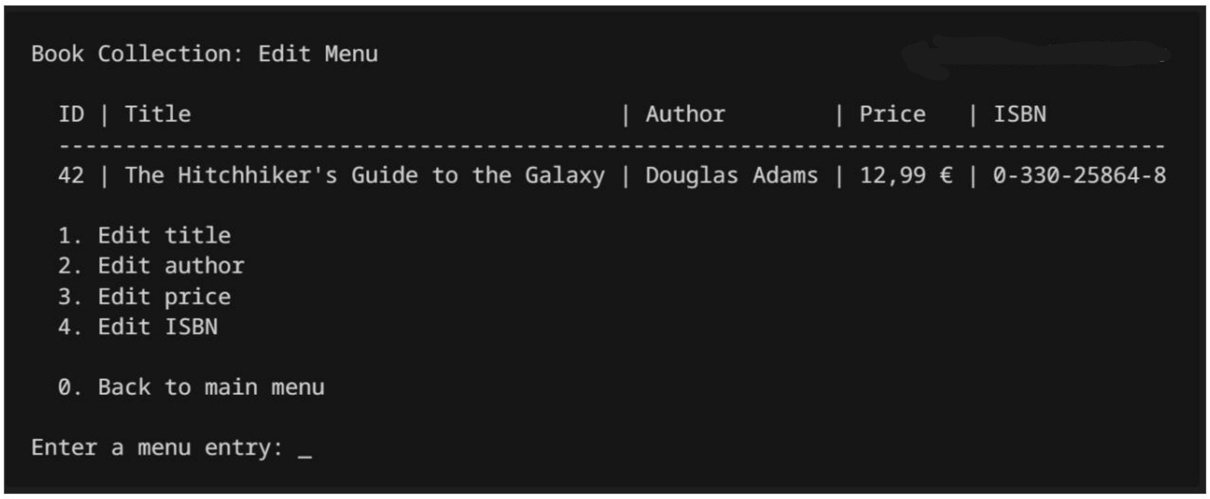


```
Book Collection: Main Menu                                  [redacted]

  1. Add new book
  2. Edit book (by ID)
  3. Remove book (by ID)
  4. Search book
  5. Sort list
  6. Print list
  7. Clear list
  8. Show list size

  0. Quit application

Enter a menu entry: _
```



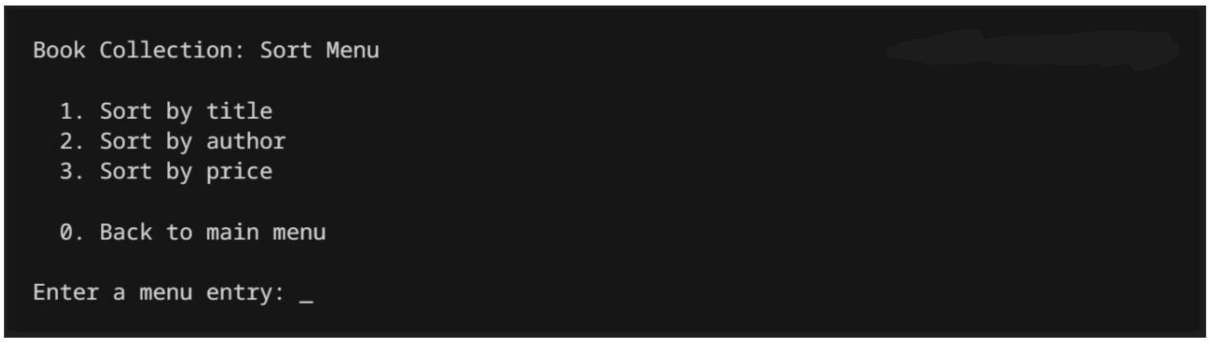
```
Book Collection: Edit Menu                                  [redacted]

  ID | Title                           | Author        | Price   | ISBN
  ----------------------------------------------------------------------------------
  42 | The Hitchhiker's Guide to the Galaxy | Douglas Adams | 12,99 € | 0-330-25864-8

  1. Edit title
  2. Edit author
  3. Edit price
  4. Edit ISBN

  0. Back to main menu

Enter a menu entry: _
```



```
Book Collection: Sort Menu                                  [redacted]

  1. Sort by title
  2. Sort by author
  3. Sort by price

  0. Back to main menu

Enter a menu entry: _
```