BookBean.java

```
package beans;


/*
  This class is simply a bean/pojo class which has fields and a constructor to handle the class books in the liberaries
*/


public class BookBean {


   private String bookName;
   private ContributorBean contributor;
   private String authorName;
   private int publishYear;
   private boolean physicallyPresent;


   public void setBookName(String bookName) {
      this.bookName = bookName;
   }


   public String getContributorName() {
      return contributor.getContributorName();
   }


   public String getAuthorName() {
      return authorName;
   }


   public void setAuthorName(String authorName) {
```

```
        this.authorName = authorName;
    }

    public int getPublishYear() {
        return publishYear;
    }

    public void setPublishYear(int publishYear) {
        this.publishYear = publishYear;
    }

    /*
        This constructor with all the fields is used for the books to be put in the library and for the
simulation purpose, it
        also contains logging for the user to see on the console when there's a book created
    */
    public BookBean(String bookName, ContributorBean contributor, String authorName,
                int publishYear, boolean physicallyPresent) {
        this.bookName = bookName;
        this.contributor = contributor;
        this.physicallyPresent = physicallyPresent;
        this.authorName = authorName;
        this.publishYear = publishYear;
        System.out.println("A new book " +
                "[" + bookName + "] contributed by the [" + contributor.getContributorName() + "]");
    }

    public String getBookName() {
        return bookName;
```

```
    }

    public boolean isPhysicallyPresent() {
        return physicallyPresent;
    }

    public void setPhysicallyPresent(boolean physicallyPresent) {
        this.physicallyPresent = physicallyPresent;
    }
}
```

ClientBean.java

```
package beans;

import java.util.ArrayList;



/*
    This class serves the purpose of the client functionality
*/

public class ClientBean {

    private String clientName;

    private ArrayList<BookBean> booksRent = new ArrayList<>();

    /*
```

```
    This method takes care of the renting a book to the client also verifying if the book is physically
present and books rented are not more than 19
   */

   public void rentBook(BookBean book) {
      if (book.isPhysicallyPresent()) {
         if (booksRent.size() < 19) {
            booksRent.add(book);
            System.out.println("Book [" + book.getBookName() + "] is rented by the client [" + clientName +
"]");
         } else {
            System.out.println("Book rented limit is over, please return to continue the renting of books");
         }
      } else {
         System.out.println("Book [" + book.getBookName() + "] is physically not available, please check
later");
      }
   }

   /*
       This method takes care of the creating a new client
   */
   public ClientBean(String clientName) {
      this.clientName = clientName;
      System.out.println("A new client [" + clientName + "] has joined the library");
   }

   /*
       This method returns all the books rented by the client
   */
```

```
    public ArrayList<BookBean> getBooksRent() {
        System.out.println("all available books rented");
        return booksRent;
    }

}
```

ContributorBean.java

```
package beans;

import java.util.ArrayList;

//This class is used for the contributors, each contributor can have many books
public class ContributorBean {
    private String contributorName;
    private ArrayList<BookBean> booksContributed = new ArrayList<>();

    public ContributorBean(String contributorName) {
        this.contributorName = contributorName;
    }

    public String getContributorName() {
        return contributorName;
    }

    public ArrayList<BookBean> getBooksContributed() {
        return booksContributed;
    }
```

```
}
```

LibraryBean.java

```
package beans;

import java.util.ArrayList;
import java.util.Calendar;

/*
   This class takes care of the library functionality
*/

public class LibraryBean {

   private ArrayList<BookBean> allBooks = new ArrayList<>();

   /*
      This method is used get all the books in the library
   */
   public ArrayList<BookBean> getAllBooks() {
      System.out.println("Books available in library are : ");
      for (BookBean book : this.allBooks) {
         System.out.println("Book named --> " + "[" + book.getBookName() + "]");
      }
      return allBooks;
   }

    /*
      This method is used add a new book in the library
```

```
*/

public void addBook(BookBean book) {
   allBooks.add(book);
}

/*
   This method is used to remove a book from the library
*/
public void removeBook(BookBean book) {
   allBooks.add(book);
}

/*
   This method is used for searching books by the name from the library
*/
public ArrayList<BookBean> searchBooksByName(String name) {
   ArrayList<BookBean> result = new ArrayList<>();
   for (BookBean book : this.allBooks) {
      if (book.getBookName().equals(name)) {
         result.add(book);
      }
   }
   if (result.size() > 0) {
      System.out.println("Books found by name are : ");
      for (BookBean book : result) {
         System.out.println(book.getBookName());
      }
   } else {
```

```
        System.out.println("No books available with this name");
    }
    return result;
}

/*
    This method is used for searching books by the contributor from the library
*/

public ArrayList<BookBean> searchBooksByContributor(String contributor) {
    ArrayList<BookBean> result = new ArrayList<>();
    for (BookBean book : this.allBooks) {
        if (book.getContributorName().equals(contributor)) {
            result.add(book);
        }
    }

    if (result.size() > 0) {
        System.out.println("Books found by contributor name are : ");
        for (BookBean book : result) {
            System.out.println(book.getBookName());
        }
    } else {
        System.out.println("No books available with this contributor name");
    }
    return result;
}

/*
```

```
    This method is used for searching books by the author from the library
  */

  public ArrayList<BookBean> searchBooksByAuthor(String author) {
    ArrayList<BookBean> result = new ArrayList<>();
    for (BookBean book : this.allBooks) {
      if (book.getAuthorName().equals(author)) {
        result.add(book);
      }
    }
    if (result.size() > 0) {
      System.out.println("Books found by author name are : ");
      for (BookBean book : result) {
        System.out.println(book.getBookName());
      }
    } else {
      System.out.println("No books available with this author name");
    }
    return result;
  }

  /*
    This method is used for searching books by the age of the books calculated by the publication year
from the library
  */

  public ArrayList<BookBean> searchBooksByAge(int age) {
    ArrayList<BookBean> result = new ArrayList<>();
    for (BookBean book : this.allBooks) {
```

```
            if ((Calendar.getInstance().get(Calendar.YEAR) - book.getPublishYear()) == age) {
                result.add(book);
            }
        }
        if (result.size() > 0) {
            System.out.println("Books found by age are : ");
            for (BookBean book : result) {
                System.out.println(book.getBookName());
            }
        } else {
            System.out.println("No books available with this age");
        }
        return result;
    }



}
```

SimulationMethods.java

```
package simulator;

import beans.BookBean;
import beans.ClientBean;
import beans.ContributorBean;
import beans.LibraryBean;

import java.util.ArrayList;
import java.util.Iterator;

/*
```

```
 *
 * This class simulates the library behavior as per the requirements of the assignments
 *
 * */
public class SimulationMethods {

   LibraryBean library = new LibraryBean();

   /*
    *
    * This method adds 3 books to the library, those books are contributed by a member
    *
    * */

   public ArrayList<BookBean> addBooksInLiberay() {

      library.addBook(new BookBean("Book of Life",
            new ContributorBean("Adam"),
            "Peterson Jack", 1970, true));
      library.addBook(new BookBean("Book of Science", new ContributorBean("Edith"),
            "Peterson Adam", 1972, true));
      library.addBook(new BookBean("Book of Creativity", new ContributorBean("Stephen"),
            "Peterson John", 1974,
            true));
      return library.getAllBooks();

   }
```

```
    /*
     *
     * This method adds 2 clients to the system with different names
     *
     * */

    public ArrayList<ClientBean> clients() {

        ArrayList<ClientBean> clients = new ArrayList<>();
        clients.add(new ClientBean("Anna"));
        clients.add(new ClientBean("Katja"));
        return clients;

    }



    /*
     *
     * This method rents the books to the clients and each book that is rented
     * gets removed from the library and is no more physically present in the library
     * first we rent all books for a client and then when second client tries to rent a book
     * it is not given because there are no more physical books present in the library
     * so it throws an Exception, which I handled so that program behaves accordingly
     * */

    public void rentBooksToClients(ArrayList<ClientBean> clients, ArrayList<BookBean> books) throws
Exception {
        //client assignment creation
        ClientBean firstClient = clients.get(0);
```

```
        ClientBean secondClient = clients.get(1);

        //renting all books for one client
        for (Iterator<BookBean> iterator = books.iterator(); iterator.hasNext(); ) {
            BookBean book = iterator.next();
            firstClient.rentBook(book);
            iterator.remove(); //once book is rented, we remove it from library
            book.setPhysicallyPresent(false);
        }

        //renting book zero for second client, we can rent any book, since no books are available it must throw a Exception which is handled
        secondClient.rentBook(books.get(0));

    }
}
```

Main.java

```
import beans.BookBean;
import beans.LibraryBean;
import simulator.SimulationMethods;

/*
    This class performs the simulation task
*/

public class Main {

    public static void main(String[] args) {
        //Main Task, The Simulation
```

```
SimulationMethods simulation = new SimulationMethods();
try {
  /*
    Perform the simulations, clients method generates the clients, addBooksInLibrary method contributes books in the liberay,
    rentBooksToClients method checks the necessary conditions for the books to be rented to the clients and once conditions meet
    books are rented to the clients and their physical availability is changed to the false
  */
  simulation.rentBooksToClients(simulation.clients(), simulation.addBooksInLiberay());
} catch (Exception ex) {
  /*
    In case there are no books and we try to rent a book to clients than an exception is thrown which is handled here,
    showing the correct message for the user
  */
  System.out.println("Library has no physical book available, please come later");
}

//Subsidiary task to check the functionality of searchingFunctions
//This can be done in the main task too, since we are not asked to do so,
//that's why we are doing it in the end, to show the working of it
LibraryBean libraryBean = new LibraryBean();

//Adding books in the library
for (BookBean book : simulation.addBooksInLiberay()) {
  libraryBean.addBook(book);
}

//Searching Books by the names
```

```
        System.out.println("\n-----NOW BOOK SEARCH BY NAME-----");
        libraryBean.searchBooksByName("Book of Life");
        System.out.print("\n");

        //Searching Books by the Age (Today - Publication Year)
        System.out.println("-----NOW BOOK SEARCH BY AGE-----");
        libraryBean.searchBooksByAge(40);
        System.out.print("\n");

        //Searching Books by the Author
        System.out.println("-----NOW BOOK SEARCH BY AUTHOR-----");
        libraryBean.searchBooksByAuthor("Peterson John");
        System.out.print("\n");

        //Searching Books by the Contributor
        System.out.println("-----NOW BOOK SEARCH BY CONTRIBUTOR-----");
        libraryBean.searchBooksByContributor("Edith");
    }
}
```