```
/**
 * Represents an individual copy of a book
 */
public class Book {
  private String title;
  private String author;
  private double width;

  private int position;  // May change when another book is removed
  private int shelf;

  /**
   * Constructs a book objekt
   * @param author the author of the book
   * @param title the title of the  book
   * @param width the width of the book (how much space of the shelf it uses)
   */
  public Book(String author, String title, double width) {

      //                                                Uppgift 1 a

  }

  /**
   * Gives a string representation showing author and title.
   * If the position is greater than 0 the shelf and position
   * is included in the representation.
   * @return the string representation
   */
  public String toString() {

      //                                                Uppgift 1 b

  }

  /**
   * Tells the width of a book
   * @return the width of the copy i.e. how much space it uses in a shelf
   */
  public double getWidth() {
    return width;
  }

  /**
   * Tells the author of a book
   * @return the author of the book
   */
  public String getAuthor() {
    return author;
  }

  /**
   * Tells the title of a book
   * @return the title of the book
   */
  public String getTitle() {
    return title;
  }

  /**
   * Sets the position of this copy
   * @param position the position in the shelf
```



```
   */
  public void setPosition(int position) {
    this.position = position;
  }

  /**
   * Sets the shelf id for this copy
   * @param shelfId the shelf id
   */
  public void setShelf(int shelfId) {
    shelf = shelfId;
  }

  /**
   * Compares this copy with another copy to see if they are
   * copies of the same book i.e. if they have the same author
   * and the same title.
   * @param b the copy to be compared to
   * @return true if they are copies of the same book else false
   */
  public boolean equals(Book b) {

      //                                                Uppgift 1 c

  }

}
```

```
import java.util.ArrayList;

/**
 * Represents a book shelf
 */
public class Shelf {
  private ArrayList<Book> theBooks;
  private int shelfId;
  private int length;    // in mm
  private int occupied;  // in mm

  /**
   * Constructs a book shelf
   * @param length the length of the shelf (in mm)
   * @param shelfId the identity number of the shelf
   */
  public Shelf(int length, int shelfId) {
    this.length = length;
    this.shelfId = shelfId;
    occupied = 0;
    theBooks = new ArrayList<Book>();
  }

  /**
   * @return the lemngth of the shelf
   */
  public int getLength() {
    return length;
  }

  /**
   * Computes the amount of free space in the shelf
   */
  public int getFreeSpace() {

      //                                              Uppgift 2 a

  }

  /**
   * Puts a book in the shelf. Records it's position and
   * shelf id in the book object.
   * @param b the book to be placed in the shelf
   * @return <bf>true</bf> if the operation was successful else <bf>false</bf>
   */
  public boolean add(Book b) {

      //                                             Uppgift 2 b

  }

  /**
   * Computes a list of the copies af a specified book in the shelf
   * @param author author of the book
   * @param title title of the book
   * @return the list of the copies of the specified book
   */
  public ArrayList<Book> findBook(String author, String title) {
    Book b = new Book(author, title, 0);
    ArrayList<Book> theList = new ArrayList<Book>();
    for (int i=0; i<theBooks.size(); i++) {
      if (theBooks.get(i).equals(b)) {
```



```
        theList.add(theBooks.get(i));
      }
    }
    return theList;
  }

  /**
   * Computes a list of all copies of all books by a specified author
   * @param author the author to be searched for
   * @return a list of all copies in this shelf of all
   * books by the specified author
   */
  public ArrayList<Book> findByAuthor(String author)  {

      //                                              Uppgift 2 c

  }

  /**
   * Removes a copy at a specified position. Recalculates the position
   * of the remaining books in the shelf.
   * @param position the position to be removed from.
   * @return The book at the specified position
   */
  public Book remove(int position) {

      //                                              Uppgift 2 d

  }

  /**
   * Prints the contents of the shelf
   */
  public void print() {
    System.out.println("Shelf " + shelfId);
    for (int i= 0; i<theBooks.size(); i++) {
      System.out.println(theBooks.get(i).toString());
    }
  }

}
```

```
import java.util.ArrayList;

/**
 * Represents a library as a collection of shelfs
 *
 */
public class Library {

  private Shelf theShelfs[];
  private int shelfSize;      // Size in mm of individual shelfs

  /**
   * Constructs a  library with (space for) a specified number of shelfs
   * @param librarySize the maximum number of shelfs in the library
   * @param shelfSize (in mm) of the standard shelf
   */
  public Library(int librarySize, int shelfSize) {

      //                                          Uppgift 3 a

  }

  /**
   * Creates a book object and puts it into the library (if possible)
   */
  public boolean add(String author, String title, int width) {
    return add(new Book(author, title, width));
  }

  /**
   * Puts a book in the library.
   * Searches a shelf with enough space to store the book and
   * puts the book there.
   * If no shelf with enough space is found a new shelf (if possible)
   * is created.
   * @param book the book to be put into the library
   */
  public boolean add(Book book) {

      //                                         Uppgift 3 b

  }

  /**
   * Removes the book at a specified shelf and position
   * @param shelf the shelf id
   * @param position the position within the shelf
   * @return the book at the specified position
   */
  public Book remove(int shelf, int position) {
    if (shelf<1 || shelf>theShelfs.length || theShelfs[shelf]==null) {
      System.out.println("No such shelf: " + shelf);
      return null;
    } else {
      return theShelfs[shelf-1].remove(position);
    }
  }


  /**
   * Searches for copies of a specified book by a specified author.
   * @param author the author
```



```
   * @param title the title
   * @return A list of all copies of the specified book
   */
  public ArrayList<Book> findBook(String author, String title) {
    ArrayList<Book> theList = new ArrayList<Book>();
    for (int i=0; i<theShelfs.length; i++) {
      Shelf s = theShelfs[i];   // Check shelf number i
      if (s!=null) {
        theList.addAll(s.findBook(author, title));
      }
    }
    return theList;
  }

  /**
   * Searches for all books by a specified author.
   * @param author the author
   * @return a list of all copies of all books by the specified author
   */
  public ArrayList<Book> findByAuthor(String author) {
    ArrayList<Book> theList = new ArrayList<Book>();
    for (int i=0; i<theShelfs.length; i++) {
      if ( theShelfs[i]!=null )
        theList.addAll(theShelfs[i].findByAuthor(author));
    }
    return theList;
  }

  /**
   * Prints the contents of the library
   */
  public void print() {
    System.out.println("\nLibrary contents");
    System.out.println("================");
    for (int i= 0; i<theShelfs.length; i++) {
      if (theShelfs[i]!=null) {
        System.out.println();
        theShelfs[i].print();
      }
    }
  }

  /**
   * Prints a list of books
   * @param listTitle a title string for the list
   * @param aList the list to be pronted
   */
  public static void printList(String listTitle, ArrayList<Book> aList) {
    System.out.println();
    System.out.println(listTitle);
    for ( Book b : aList) {
      System.out.println("\t" + b);
    }
  }

  /**
   * Test program
   */
  public static void main(String[] args) {
    Library lib = new Library(10, 100);
    lib.add("Adams", "Liftarens guide till galaxen", 38);
    lib.add("Boye", "Kris", 47);
```

```
    lib.add("Dawkins", "The God Delusion", 50);
    lib.add("Adams", "Liftarens guide till galaxen", 38);
    lib.add("Boye", "Moln", 28);
    lib.add("Milne", "Nalle Puh", 22);
    lib.add("Fitzgerald", "The Great Gatsby", 36);
    lib.add("Golding", "Flugornas Herre", 47);
    lib.add("Joyce", "Ulysses", 38);
    lib.add("Heller", "Catch-22", 42);
    lib.add("Boye", "Kallocain", 37);
    lib.add("Adams", "Liftarens guide till galaxen", 38);
    lib.print();

    printList("By Boye", lib.findByAuthor("Boye"));
    printList("By Adams", lib.findByAuthor("Adams"));
    printList("Moln by Boye:", lib.findBook("Boye", "Moln"));
    System.out.println("\nRemoved: " + lib.remove(3,1));
    lib.print();
    System.out.println("\nIllegal remove: (3,8)");
    lib.remove(3,8);
    System.out.println("\nIllegal remove: (0,1)");
    lib.remove(0,1);
  }

}

/** Körresultat:

Library contents
================

Shelf 1
Adams      : Liftarens guide till galaxen <1, 1>
Boye       : Kris                         <1, 2>

Shelf 2
Dawkins    : The God Delusion             <2, 1>
Adams      : Liftarens guide till galaxen <2, 2>

Shelf 3
Boye       : Moln                         <3, 1>
Milne      : Nalle Puh                    <3, 2>
Fitzgerald : The Great Gatsby             <3, 3>

Shelf 4
Golding    : Flugornas Herre              <4, 1>
Joyce      : Ulysses                      <4, 2>

Shelf 5
Heller     : Catch-22                     <5, 1>
Boye       : Kallocain                    <5, 2>

Shelf 6
Adams      : Liftarens guide till galaxen <6, 1>

By Boye
        Boye       : Kris                         <1, 2>
        Boye       : Moln                         <3, 1>
        Boye       : Kallocain                    <5, 2>

By Adams
        Adams      : Liftarens guide till galaxen <1, 1>
        Adams      : Liftarens guide till galaxen <2, 2>
```



```
        Adams      : Liftarens guide till galaxen <6, 1>

Moln by Boye:
        Boye       : Moln                         <3, 1>

Removed: Boye       : Moln

Library contents
================

Shelf 1
Adams      : Liftarens guide till galaxen <1, 1>
Boye       : Kris                         <1, 2>

Shelf 2
Dawkins    : The God Delusion             <2, 1>
Adams      : Liftarens guide till galaxen <2, 2>

Shelf 3
Milne      : Nalle Puh                    <3, 1>
Fitzgerald : The Great Gatsby             <3, 2>

Shelf 4
Golding    : Flugornas Herre              <4, 1>
Joyce      : Ulysses                      <4, 2>

Shelf 5
Heller     : Catch-22                     <5, 1>
Boye       : Kallocain                    <5, 2>

Shelf 6
Adams      : Liftarens guide till galaxen <6, 1>

Illegal remove: (3,8)
Shelf 3 has no book at position 8

Illegal remove: (0,1)
No such shelf: 0


 */
```