**Assignment 2**

1.

Date: 27 May 2020.

File names: Book.java, Client.java, Crime.java, Drama.java

Purpose: Library application

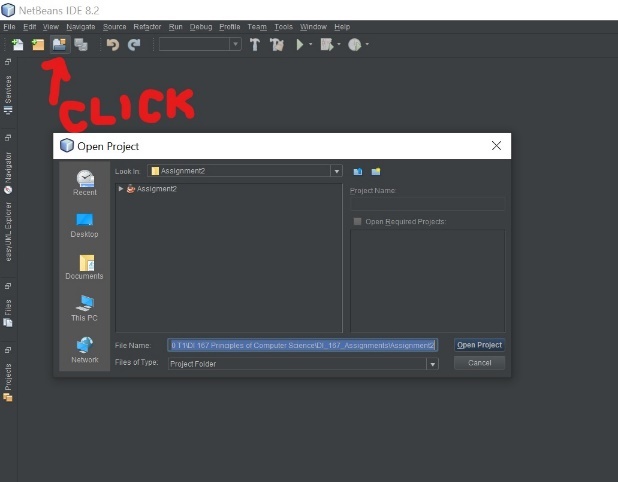
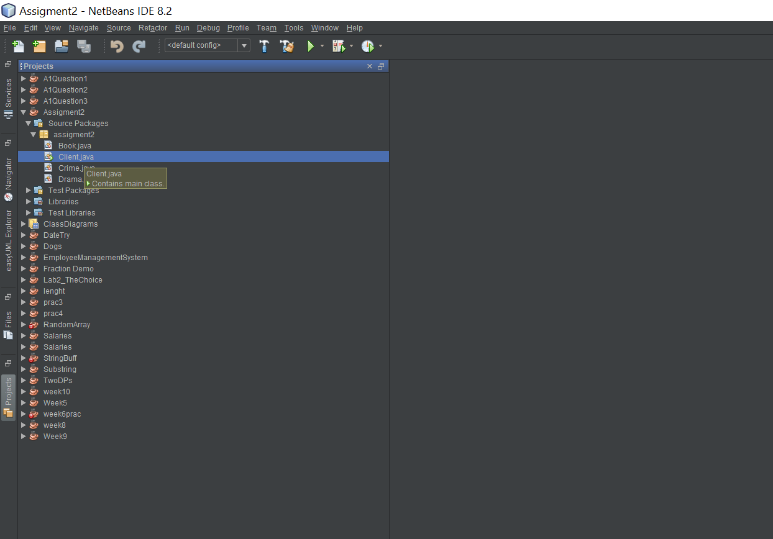
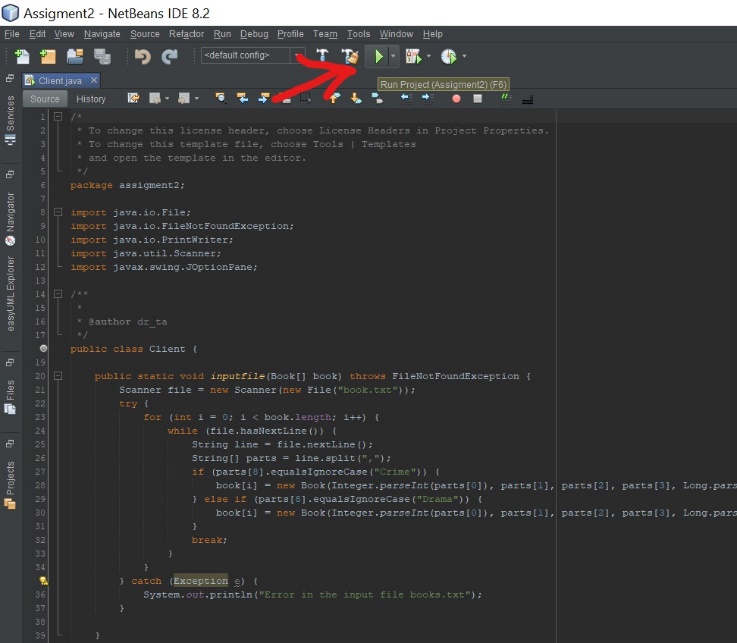
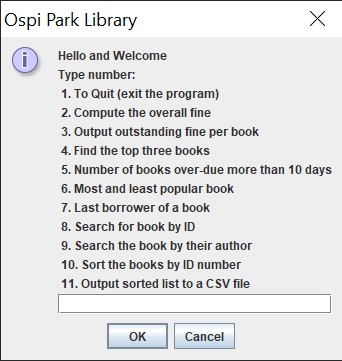
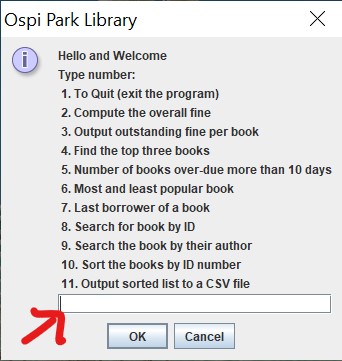
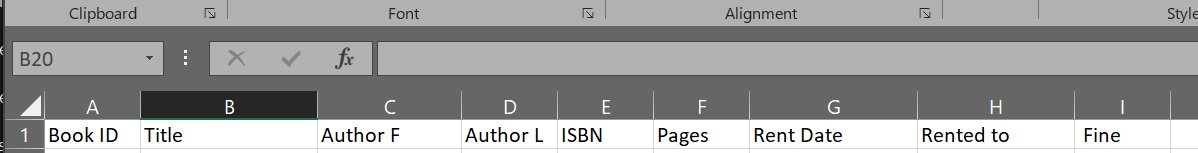
2.

Requirements/Specification:

This program takes a list of books with all their details from "book.txt" and saves them in a library application. Library application can be used for many things: searching for books by their IDs or Author names, also it can calculate fines for late return of the books, sort books by their ID and after that there is an option to print sorted list to "SortedList.csv" file. Menu of this program pops up at the very start and it contains 11 options. 1. To Quit (exit the program); 2. Compute the overall fine; 3. Output outstanding fine per book; 4. Find the top three books; 5. Number of books over-due more than 10 days; 6. Most and least popular book; 7. Last borrower of a book; 8. Search for book by ID; 9. Search the book by their author; 10. Sort the books by ID number; 11. Output sorted list to a CSV file.

3.

User Guide:

1. This program is in zip file. Unzip file and save it on your computer and open NetBeans.
2. Once NetBeans is open, go to top right corner and click “Open project” or press Ctrl+Shift+O
3. Navigate to the destination where this program is saved and select Assignment2 and click Open project
4. Find Assignment2 on the list of projects and double click on Client.java, this will show the source code
5. Click on a green arrow or press “F6” to run the application
6. Program will automatically import list of books from book.txt located inside Assignment2 File. To Choose any of the options simply type desirable number into the Line
7. Records in book.txt file **MUST** be in following order Book ID, Book Title, Authors First Name, Authors Last Name, ISBN number of the book, Number of book pages, Date when book is rented, Name of the person that book is rented, genre(Crime/Drama). Fine will be automatically calculated and displayed in .csv file at the end of all other book details.
8. List of books will be sorted by ID numbers and printed to .csv file **only if** the option “10” has been selected before option 11. Otherwise List will be identical to what it is in Book.txt file.

4.

Algorithm:

Book class

SET Variable boolean type

SET Variable string title

SET Variable intiger id

SET Variable long isbn

SET Variable string fauthor

SET Variable string lauthor

SET Variable intiger pages

SET Variable string date

SET Variable int day

SET Variable string borrow

SET Variable int fine

Default constructor

Initialize type

Initialize title

Initialize id

Initialize isbn

Initialize fauthor

Initialize lauthor

Initialize pages

Initialize day

Initialize date

Initialize borrow

Initialize fine

Initialize Start

Constructor with arguments (…)

Initialize type to ftype

Initialize title to temtitle

Initialize id to temid

Initialize isbn to temisbn

Initialize fauthor to temfauthor

Initialize lauthor to temlauthor

Initialize pages to tempages

Initialize day to 0

Initialize date temdate

Initialize borrow to temborrow

Initialize fine to 0

Method sort of books

New book Object

for each ID in books length

if books j-1 ID> books j ID

temp = books j - 1

books j – 1 = books j

books j = temp;

End sort method

Method get how many days diference

New current day object

String s1

for each Date in books length

declare date from records

New record date object

Compare current date and one from records and / by (1000 \* 60 \* 60 \* 24));

End of method

Method to calculateFines

IF type is true

New crime object

Receive fine from crime class

ELSE

New drama object

Receive fine from drama class

End of method

GETTERS

Get Title

return title

Get ID

return id

Get Isbn

return isbn

Get Fauthor

return fauthor;

Get Lauthor

return lauthor;

Get Pages

return pages;

Get Date

return date;

Get Borrow

return borrow;

Get Day

return day;

Get Fine

return fine;

Get Type

return type;

SETTERS

Set Title

Set Id

Set Isbn

Set Fauthor

Set Lauthor

Set Pages

Set Date

Set Borrow

Set Day

Crime Class

Method for calculating fines

IF less than 3 days

Fine = 0

Else IF 3-8 days

Fine=day\*5

Else IF 8-16 days

Fine = day \*10

Else “error”

Return fine

Drama Class

Method for calculating fines

IF less than 3 days

Fine = 0

Else IF 3-8 days

Fine=day\*7

Else IF 8-16 days

Fine = day \*12

Else “error”

Return fine

Client class

Method to input file throws FileNotFoundException

New Scanner Object(declare file)

Try

For book length

Read while file has next line

IF Crime

Book array(parts 0-7,true)

ELSE IF Drama

Book array(parts 0-7,false)

Break

Catch Exception

Print error message

End method

Main method

New book object

Call input file method

Call day method

Call fine method

Joptionpane

While (true)

Try

Create input dialog with options

Switch

case1

Message dialog Goodbye msg

Exit(0/false)

break

case2

Declare and initialize integer for total Crime

Declare and initialize integer for total Drama

For each book length

If Book type crime

Total = sum all crime fines

Else

Total =sum all drama fines

Message dialog Drama total, Crime total

break

case3

Declare string display

For each book length

Display books with all details + fines individually

Message dialog show display

break

case4

Message dialog not available

break

case5

Declare integer counter

For each book length

If day>10

Count

Message dialog how many books overdue more than 10 days

Break

case6

Message dialog not available

break

case7

Declare integer last

Initialize last from daylist

For each book length

IF day from list<last

Save last

Declare and initialize string to display

Display details of [last]

Message dialog for display

break

case8

Message dialog asking for input

Declare integer id from list of IDs

For each book length

IF ID matching record

Message dialog to display book details

break

ELSE

IF ID not matching record

Message dialog no result

break

case9

Message dialog asking for input

Declare string name

For each book length

IF first name OR last name matching record

Message dialog to display book details

break

ELSE

IF ID not matching record

Message dialog no result

Break

case10

Declare string sort

For each book length

Call sort method

For each book length

Display sort with details

Message dialog to display book details

break

case11

Declare and initialize string file name

Declare Print Writer

Try

New print writer object

Set heather for file

For each book length

Output ID

Separate with “,”

Output Title

Separate with “,”

Output First name

Separate with “,”

Output Last name

Separate with “,”

Output ISBN

Separate with “,”

Output Pages

Separate with “,”

Output Issue date

Separate with “,”

Output Borrower name

Separate with “,”

Output Fine

Separate with “,”

Close program

Message dialog that printing was successful

Catch exception

Message dialog about unexpected error

Break

Default for option menu

For number out of range 1-11

Message dialog about wrong input

Break

Catch exception for option menu

Message dialog about wrong input

End while

End Menu

End Client Class

5.

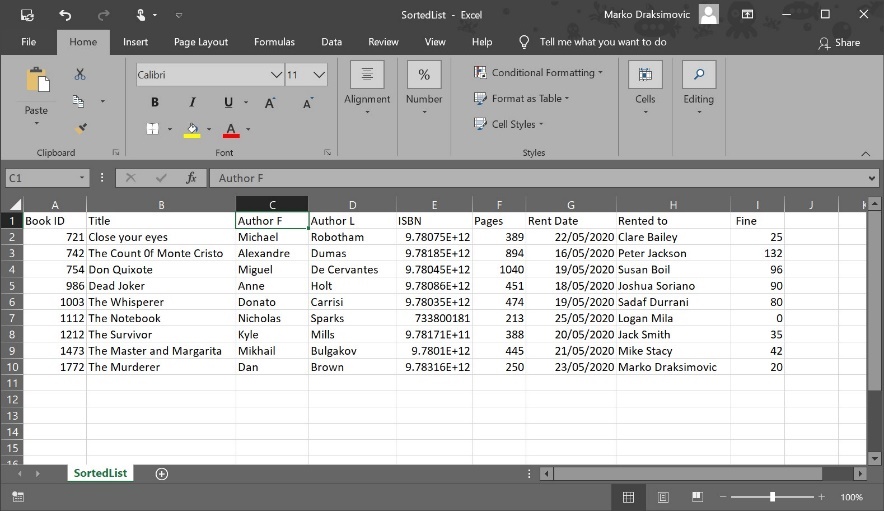
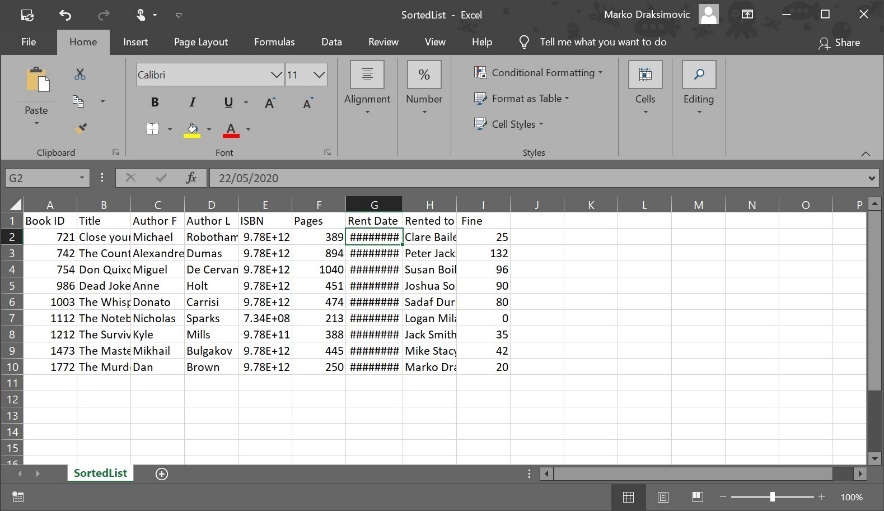
Limitations:

1) Does not perform option 4: Find the top three books; instead message "Sorry this option is not available!" will be displayed.

2) Does not perform option 6: Most and least popular book; instead message "Sorry this option is not available!" will be displayed.

3) I used poor solution, Boolean, to determine type of book such as crime and drama. If third genre is needed this section would have to be improved as new type such as comedy or biography could not be implemented.

4) all books in book.txt file must be borrowed, meaning they need to have date when they are borrowed and borrowers name. If this is not included program will not run.

5) in .csv file cells for Date and ISBN number are not showing correctly. If you spread out this cells date will show up but ISBN will stay shorten available to be reviewed on formula tab.

6) Pressing “x” button or cancel will not close application, this is possible only if user types “1”.

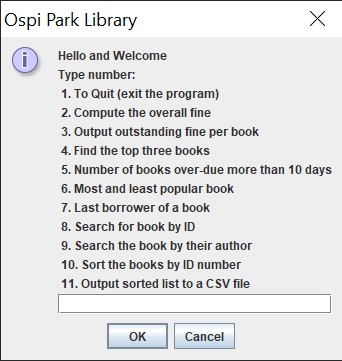
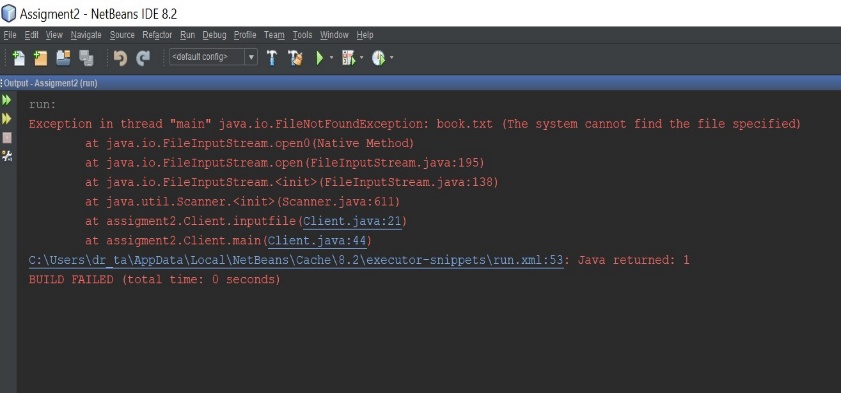
7) Borrow period can’t be more than 16 days, if by any chance is, fine for this will be “0”

8)Sorted books by their ID will be printed correctly in order only if option 10 was entered before 11.

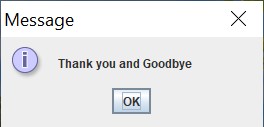
6.

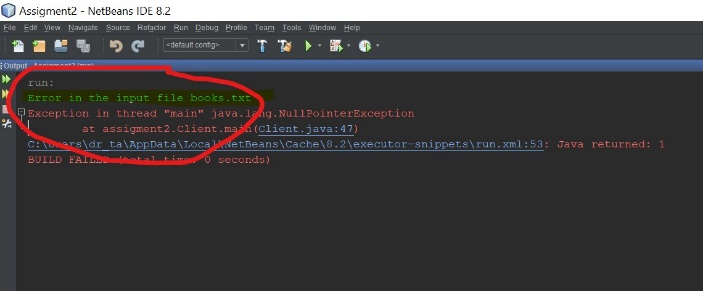
Testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test id:** | **Test description/justification – what is the test for and why this particular test** | **Actual data for this test** | **Expected output** | **Actual desk check; result when desk check is carried out** | **Test outcome : Pass/Fail** |
| 1 | Run the program | Press run or F6 | New window will be opened and menu fill be displayed | New window will be opened and menu fill be displayed | Pass |
| 2 | To test exception change name of the book.txt file | BookS.txt | The system can not find the file specified | The system can not find the file specified | Pass |
| 3 | Test error message for Wrong information in Book.txt file | No name | Error in the input file books.txt | Error in the input file books.txt | Pass |
| 4 | Check option 1 | 1 | Thank you and Goodbye | Thank you and Goodbye | Pass |
| 5 | Check option 2 | 2 | Overall fine  DRAMA:204 AUD  CRIME:180 AUD | Overall fine  DRAMA:204 AUD  CRIME:180 AUD | Pass |
| 6 | Check option 3 | 3 | List of books with calculated fine at the end of details | List of books with calculated fine at the end of details | Pass |
| 7 | Check option 4 | 4 | Sorry this option is not available! | Sorry this option is not available! | Pass |
| 8 | Check option 5 | 5 | There are 0 books that are over-due more than 10 days | There are 0 books that are over-due more than 10 days | Pass |
| 9 | Check option 6 | 6 | Sorry this option is not available! | Sorry this option is not available! | Pass |
| 10 | Check option 7 | 7 | LAST BORROWER 1112, The Notebook, Nicholas,  Sparks, 0733800181, 213, 2020/05/25, Logan Mila, 0 AUD | LAST BORROWER 1112, The Notebook, Nicholas,  Sparks, 0733800181, 213, 2020/05/25, Logan Mila, 0 AUD | Pass |
| 11 | Check option 8 | 8 | Please Enter the Book ID: | Please Enter the Book ID: | Pass |
| 12 | Under option 8 enter correct ID number of the book | 742 | Book Found!  742, The Count 0f Monte Cristo, Alexandre, Dumas, 9781853267338, 894, 2020/05/16, Peter Jackson, 120 AUD | Book Found!  742, The Count 0f Monte Cristo, Alexandre, Dumas, 9781853267338, 894, 2020/05/16, Peter Jackson, 120 AUD | Pass |
| 13 | Under option 8 enter incorrect ID number of the book | 743 | No results! | No results! | Pass |
| 14 | Under option 8 enter a word instead of a number | aaa | Error, wrong input! | Error, wrong input! | Pass |
| 15 | Check option 9 | 9 | Please Enter First or Last name of the Author | Please Enter First or Last name of the Author | Pass |
| 16 | Under option 9 enter correct first author name using randomly upper and lower cases | dONaTo | Book Found!  1003, The Whisperer, Donato, Carrisi , 9780349123448, 474, 2020/05/19, Sadaf Durrani, 35 AUD | Book Found!  1003, The Whisperer, Donato, Carrisi , 9780349123448, 474, 2020/05/19, Sadaf Durrani, 35 AUD | Pass |
| 17 | Under option 9 enter correct last author name using randomly upper and lower cases | dUmAs | Book Found!  742, The Count 0f Monte Cristo, Alexandre, Dumas, 9781853267338, 894, 2020/05/16, Peter Jackson, 120 AUD | Book Found!  742, The Count 0f Monte Cristo, Alexandre, Dumas, 9781853267338, 894, 2020/05/16, Peter Jackson, 120 AUD | Pass |
| 18 | Under option 9 enter incorrect author name | stefan | No results! | No results! | Pass |
| 19 | Under option 9 enter a number instead author name | 123 | Error, wrong input! | No results! | Fail |
| 20 | Check option 10 | 10 | Sort all books in order by their ID number from smallest to largest | Sort all books in order by their ID number from smallest to largest | Pass |
| 21 | Check option 11after running option 10 first | 11 | Sorted list printed Successfully  Books sorted in order by their ID number from smallest to largest and printed to SortedList.csv | Sorted list printed Successfully  Books sorted in order by their ID number from smallest to largest and printed to SortedList.csv | Pass |
| 22 | Check option 11before running option 10 first | 11 | Sorted list printed Successfully  Books NOT sorted in order by their ID number from smallest to largest and printed to SortedList.csv | Sorted list printed Successfully  Books NOT sorted in order by their ID number from smallest to largest and printed to SortedList.csv | Pass |
| 23 | In option menu enter option out of range 1-11 | 12 | Error, wrong input! | Error, wrong input! | Pass |
| 24 | In option menu enter word instead of a number | bla bla | Error, wrong input! | Error, wrong input! | Pass |
| 25 | In option menu press OK button without putting input | OK | Error, wrong input! | Error, wrong input! | Pass |
| 26 | In option menu press Cancel button without putting input. It is not set up in program so it should not exit program | Cancel | Error, wrong input! | Error, wrong input! | Pass |
| 27 | Press X in top right corner of option window. It is not set up in program so it should not exit program | X | Error, wrong input! | Error, wrong input! | Pass |

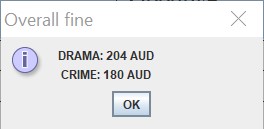


**Test 1 Test 2**



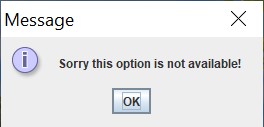
****

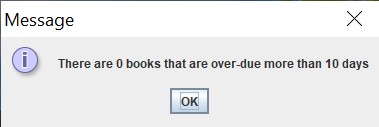
**Test 3**

 **Test 4**

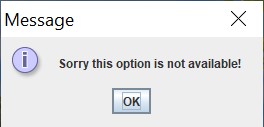
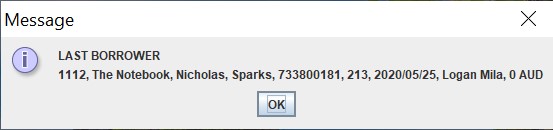


**Test 5 Test 6**

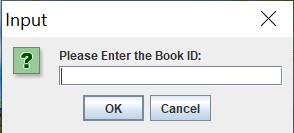
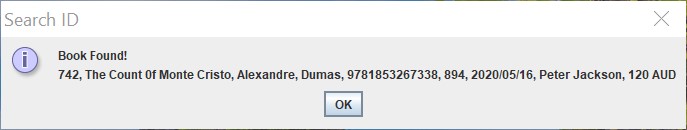
****



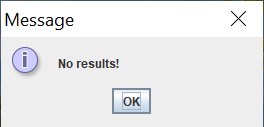
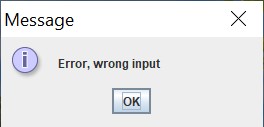
**Test 7 Test 8**

****

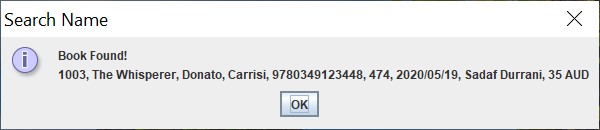
**Test 9 Test 10**

****

**Test 11 Test 12**

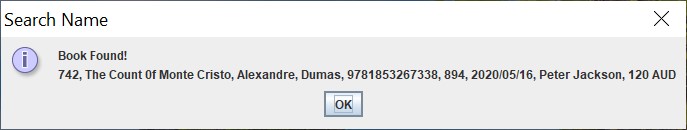
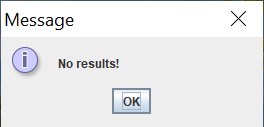
****

**Test 13 Test 14**

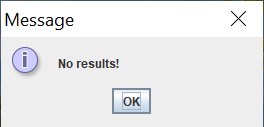


****

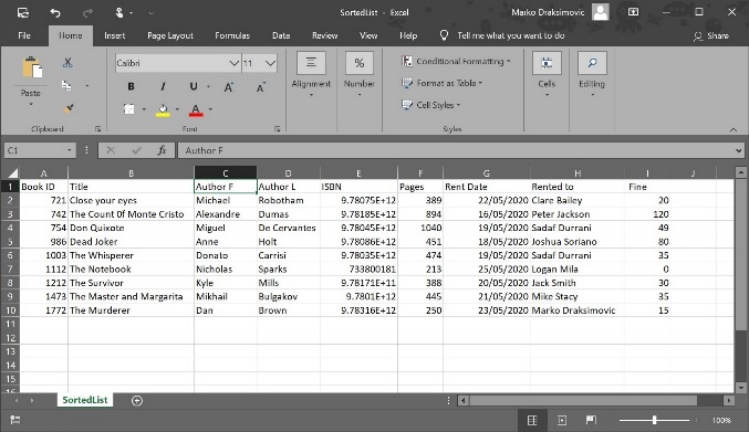
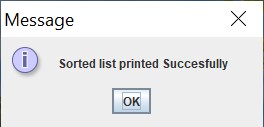
**Test 15 Test 16**



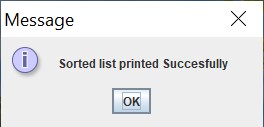
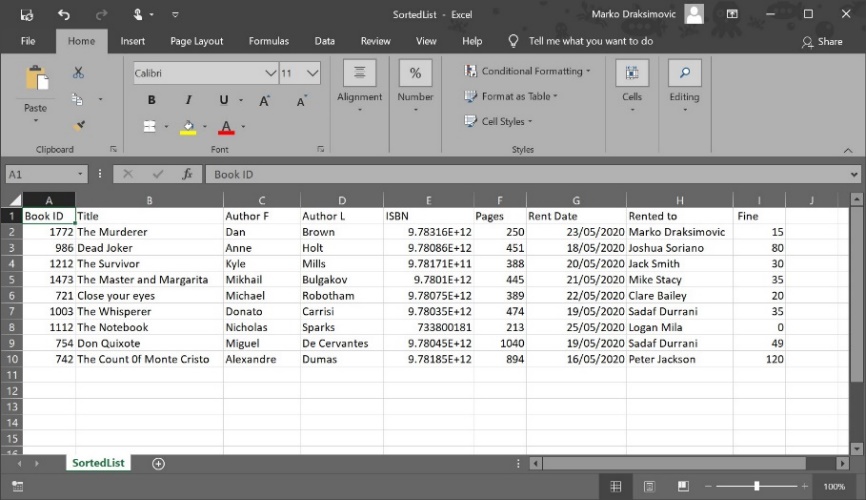
**Test 17 Test 18**



**Test 19 Test 20**



**Test 21**

 **Test 22**





**Test 23 Test 24**



**Test 25 Test 26**

**Test 27**

7.

Source program listings:

**CLIENT CLASS**

public class Book extends Client {

/\*\*

\* @param args the command line arguments

\*/

private boolean type;

private String title;

private int id;

private long isbn;

private String fauthor;

private String lauthor;

private int pages;

private String date;

public int day;

private String borrow;

public int fine;

// default constructor

public Book() {

type = false;

title = "No name yet.";

id = 0;

isbn = 0;

fauthor = "No First Name";

lauthor = "No Last Name";

pages = 0;

day = 0;

date = null;

borrow = null;

fine = 0;

}

//constructor with seven arg

public Book(int temid, String temtitle, String temfauthor, String temlauthor, long temisbn, int tempages, String temdate, String temborrow, boolean ftype) {

type = ftype;

title = temtitle;

id = temid;

isbn = temisbn;

fauthor = temfauthor;

lauthor = temlauthor;

pages = tempages;

day = 0;

date = temdate;

borrow = temborrow;

fine = 0;

}

public void sort(Book[] books) {

Book temp = new Book();

for (int i = 0; i < books.length; i++) {

for (int j = 1; j < (books.length - i); j++) {

if (books[j - 1].getId() > books[j].getId()) {

temp = books[j - 1];

books[j - 1] = books[j];

books[j] = temp;

}

}

}

}

public void day(Book[] books) {

Date current = new Date();

String s1;

for (int i = 0; i < books.length; i++) {

s1 = books[i].getDate();

Date record = new Date(s1);

books[i].setDay((int) (current.getTime() - record.getTime()) / (1000 \* 60 \* 60 \* 24));

}

}

public void calculateFine(Book[] books) {

if (type == true) {

Crime book = new Crime();

fine = book.CrimeFine(day, fine);

} else {

Drama book = new Drama();

fine = book.DramaFine(day, fine);

}

}

//GETTERS

public String getTitle() {

return title;

}

public int getId() {

return id;

}

public long getIsbn() {

return isbn;

}

public String getFauthor() {

return fauthor;

}

public String getLauthor() {

return lauthor;

}

public int getPages() {

return pages;

}

public String getDate() {

return date;

}

public String getBorrow() {

return borrow;

}

public int getDay() {

return day;

}

public int getFine() {

return fine;

}

public boolean isType() {

return type;

}

//SETTERS

public void setTitle(String title) {

this.title = title;

}

public void setId(int id) {

this.id = id;

}

public void setIsbn(int isbn) {

this.isbn = isbn;

}

public void setFauthor(String author) {

this.fauthor = author;

}

public void setLauthor(String lauthor) {

this.lauthor = lauthor;

}

public void setPages(int pages) {

this.pages = pages;

}

public void setDate(String date) {

this.date = date;

}

public void setBorrow(String borrow) {

this.borrow = borrow;

}

public void setDay(int day) {

this.day = day;

}

}

public class Client {

public static void inputfile(Book[] book) throws FileNotFoundException {

Scanner file = new Scanner(new File("book.txt"));

try {

for (int i = 0; i < book.length; i++) {

while (file.hasNextLine()) {

String line = file.nextLine();

String[] parts = line.split(",");

if (parts[8].equalsIgnoreCase("Crime")) {

book[i] = new Book(Integer.parseInt(parts[0]), parts[1], parts[2], parts[3], Long.parseLong(parts[4]), Integer.parseInt(parts[5]), parts[6], parts[7], true);

} else if (parts[8].equalsIgnoreCase("Drama")) {

book[i] = new Book(Integer.parseInt(parts[0]), parts[1], parts[2], parts[3], Long.parseLong(parts[4]), Integer.parseInt(parts[5]), parts[6], parts[7], false);

}

break;

}

}

} catch (Exception e) {

System.out.println("Error in the input file books.txt");

}

}

public static void main(String[] args) throws FileNotFoundException {

Book[] books = new Book[9];

inputfile(books);

Scanner kb = new Scanner(System.in);

//This line calculates the days

books[0].day(books);

//THIS LINE CALCULATE FINES

for (int i = 0; i < books.length; i++) {

books[i].calculateFine(books);

}

//JOPTIONPANE

while (true) {

//display Menu

try {

String OspiLibrary = JOptionPane.showInputDialog(null, "Hello and Welcome" + "\nType number: \n 1. To Quit (exit the program)"

+ "\n 2. Compute the overall fine" + "\n 3. Output outstanding fine per book" + "\n 4. Find the top three books"

+ "\n 5. Number of books over-due more than 10 days" + "\n 6.Most and least popular book" + "\n 7. Last borrower of a book"

+ "\n 8. Search for book by ID" + "\n 9. Search the book by their author" + "\n 10. Sort the books by ID number"

+ "\n 11. Output sorted list to a CSV file", "Ospi Park Library", JOptionPane.INFORMATION\_MESSAGE);

switch (Integer.parseInt(OspiLibrary)) {

case 1:

JOptionPane.showMessageDialog(null, "Thank you and Goodbye");

System.exit(0);

break;

case 2:

int totalcrime = 0;

int totaldrama = 0;

for (int i = 0; i < books.length; i++) {

if (books[i].isType()) {

totalcrime = totalcrime + books[i].getFine();

} else {

totaldrama = totaldrama + books[i].getFine();

}

}

//Display case2

JOptionPane.showMessageDialog(null, "DRAMA: " + totaldrama + " AUD" + "\n " + "CRIME: " + totalcrime + " AUD", "Overall fine", JOptionPane.INFORMATION\_MESSAGE);

break;

case 3:

//Books details with fines

String display = "";

for (int i = 0; i < books.length; i++) {

display = display + books[i].getId() + ", " + books[i].getTitle() + ", " + books[i].getFauthor() + ", " + books[i].getLauthor() + ", "

+ books[i].getIsbn() + ", " + books[i].getPages() + ", "

+ books[i].getDate() + ", " + books[i].getBorrow() + ", " + books[i].getFine() + " AUD" + "\n";

}

//Display case3

JOptionPane.showMessageDialog(null, display, "Outstanding fine per book", JOptionPane.INFORMATION\_MESSAGE);

break;

case 4://not available

JOptionPane.showMessageDialog(null, "Sorry this option is not available!");

break;

case 5:

int counter = 0;

for (int i = 0; i < books.length; i++) {

if (books[i].getDay() > 10) {

counter++;

}

}

//Display message

JOptionPane.showMessageDialog(null, "There are " + counter + " books that are over-due more than 10 days");

break;

case 6://not available

JOptionPane.showMessageDialog(null, "Sorry this option is not available!");

break;

case 7:

int last;

last = books[0].getDay();

for (int i = 0; i < books.length; i++) {

if (books[i].getDay() < last) {

last = i;

}

}

//Display

String disp = "LAST BORROWER \n";

disp = disp + books[last].getId() + ", " + books[last].getTitle() + ", " + books[last].getFauthor() + ", " + books[last].getLauthor() + ", "

+ books[last].getIsbn() + ", " + books[last].getPages() + ", "

+ books[last].getDate() + ", " + books[last].getBorrow() + ", " + books[last].getFine() + " AUD" + "\n";

JOptionPane.showMessageDialog(null, disp);

break;

case 8:

String option = JOptionPane.showInputDialog("Please Enter the Book ID: ");

int id = Integer.parseInt(option);

for (int i = 0; i < books.length; i++) {

if (books[i].getId() == id) {

JOptionPane.showMessageDialog(null, "Book Found!" + "\n" + books[i].getId() + ", " + books[i].getTitle() + ", " + books[i].getFauthor() + ", " + books[i].getLauthor() + ", "

+ books[i].getIsbn() + ", " + books[i].getPages() + ", "

+ books[i].getDate() + ", " + books[i].getBorrow() + ", " + books[i].getFine() + " AUD" + "\n", "Search ID", JOptionPane.INFORMATION\_MESSAGE);

//printing(books[i]);

break;

} else {

if (i == books.length - 1 && books[i].getId() != id) {

JOptionPane.showMessageDialog(null, "No results!");

}

}

}

break;

case 9:

String namesearch = JOptionPane.showInputDialog("Please Enter First or Last name of the Author ");

String fauthor = namesearch;

for (int i = 0; i < books.length; i++) {

if (books[i].getFauthor().equalsIgnoreCase(fauthor) || books[i].getLauthor().equalsIgnoreCase(fauthor)) {

JOptionPane.showMessageDialog(null, "Book Found!" + "\n" + books[i].getId() + ", " + books[i].getTitle() + ", " + books[i].getFauthor() + ", " + books[i].getLauthor() + ", "

+ books[i].getIsbn() + ", " + books[i].getPages() + ", "

+ books[i].getDate() + ", " + books[i].getBorrow() + ", " + books[i].getFine() + " AUD" + "\n", "Search Name", JOptionPane.INFORMATION\_MESSAGE);

break;

} else {

if (i == books.length - 1 && (!books[i].getFauthor().equalsIgnoreCase(fauthor) || !books[i].getLauthor().equalsIgnoreCase(fauthor))) {

JOptionPane.showMessageDialog(null, "No results!");

}

}

}

break;

case 10:

String sort = "";

for (int i = 0; i < books.length; i++) {

books[i].sort(books);

}

for (int i = 0; i < books.length; i++) {

//printing(books[i]);

sort = sort + books[i].getId() + ", " + books[i].getTitle() + ", " + books[i].getFauthor() + ", " + books[i].getLauthor() + ", "

+ books[i].getIsbn() + ", " + books[i].getPages() + ", "

+ books[i].getDate() + ", " + books[i].getBorrow() + ", " + books[i].getFine() + " AUD" + "\n";

}

JOptionPane.showMessageDialog(null, sort, "Sorted by ID number", JOptionPane.INFORMATION\_MESSAGE);

break;

case 11:

String fileName = "SortedList.csv";

PrintWriter outputStream = null;

try {

outputStream = new PrintWriter(fileName);

outputStream.println("Book ID,Title,Author F,Author L,ISBN,Pages,Rent Date,Rented to, Fine");

for (int i = 0; i < books.length; i++) {

outputStream.print(books[i].getId());

outputStream.append(',');

outputStream.print(books[i].getTitle());

outputStream.append(',');

outputStream.print(books[i].getFauthor());

outputStream.append(',');

outputStream.print(books[i].getLauthor());

outputStream.append(',');

outputStream.print(books[i].getIsbn());

outputStream.append(',');

outputStream.print(books[i].getPages());

outputStream.append(',');

outputStream.print(books[i].getDate());

outputStream.append(',');

outputStream.print(books[i].getBorrow());

outputStream.append(',');

outputStream.print(books[i].getFine());

outputStream.append('\n');

}

outputStream.close();

JOptionPane.showMessageDialog(null, "Sorted list printed Succesfully"); //confirm printing to .csv file

} catch (Exception e) {

JOptionPane.showMessageDialog(null, "Unexspected ERROR!");

}

break;

default:

JOptionPane.showMessageDialog(null, "ERROR!");

break;

}//end of switch

} catch (Exception e) {

JOptionPane.showMessageDialog(null, "Error, wrong input"); //pop out message if input for menu is wrong

}

}//end of while loop set up to repeat until type: 1

}//end of Manu

}//end of Client class

public class Crime extends Book {

public int CrimeFine(int day, int fine) {

if (day < 3) {

fine = 0;

} else if (day >= 3 && day < 8) {

fine = day \* 5;

} else if (day >= 8 && day <= 16) {

fine = day \* 10;

} else {

System.out.println("Error");

}

return fine;

}

}

public class Drama extends Book {

public int DramaFine(int day, int fine) {

if (day < 3) {

fine = 0;

} else if (day >= 3 && day < 8) {

fine = day \* 7;

} else if (day >= 8 && day <= 16) {

fine = day \* 12;

} else {

System.out.println("Error");

}

return fine;

}

}