import java.io.File;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.RandomAccessFile;

import java.util.Arrays;

import java.util.Scanner;

/\*

1. Да се състави клас Book, имплементиращ Comparable

Частни полета: заглавие, име на автор, година на издаване, страници;

1.1 Подразбиращ се конструктор;

1.2 Експлицитен конструктор – по всички променливи;

1.3 Методи за четене/запис на член-променливите;

1.4 Метод за стрингова интерпретация;

1.5 Метод за еквивалентност;

1.6 Имплементация на интерфейса.

2. Да се състави клас Library

Частен масив от книги;

2.1 Експлицитен конструктор с име на файл - public Library(String filename)

Формат на файла:

<заглавие>SP<име на автор>SP<година на издаване>SP<страници>CR

2.2 Метод за извеждане на масива на конзола;

2.3 Метод за сортиране на масива по година на издаване;

2.4 Метод, който изчислява и връща общия брой страници на книгите от масива;

2.5 Метод, който изчислява и връща броя книги, чиито страници са повече от зададен параметър;

2.6 Метод, който изчислява и върща колко книги от даден автор има в масива;

2.7 Метод, който изчислява и връща средната стойност на страниците на книгите от масива;

2.8 Метод, който изчислява и връща броя книги, които са издадени след определена година;

2.9 Метод, който намира и връща заглавието на книгата, издадена най-късно.

3. Да се състави MainClass с главна функция, която:

3.1 Създава обект от т.2, извежда;

3.2 Сортира масива, извежда;

3.3 Намира общият брой страници, извежда;

3.4 Намира брой книги със страници над зададена стойност, извежда;

3.5 Намира брой книги от даден автор, извежда;

3.6 Намира средната стойност на страниците, извежда;

3.7 Намира брой книги, издадени след определена година, извежда;

3.8 Намира заглавието на книгата, издадена последна, извежда.

\*/

class Book implements Comparable {

// Частни полета: заглавие, име на автор, година на издаване, страници;

String strTitle;

String strAuthor;

int year;

int pages;

// 1.1 Подразбиращ се конструктор;

public Book() {

this.year = 0;

this.pages = 0;

}

// 1.2 Експлицитен конструктор – по всички променливи;

public Book(String strTitle, String strAuthor, int year, int pages) {

this.strTitle = strTitle;

this.strAuthor = strAuthor;

this.year = year;

this.pages = pages;

}

// 1.3 Методи за четене/запис на член-променливите;

public String getStrTitle() {

return strTitle;

}

public void setStrTitle(String strTitle) {

this.strTitle = strTitle;

}

public String getStrAuthor() {

return strAuthor;

}

public void setStrAuthor(String strAuthor) {

this.strAuthor = strAuthor;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getPages() {

return pages;

}

public void setPages(int pages) {

this.pages = pages;

}

// 1.4 Метод за стрингова интерпретация;

@Override

public String toString() {

return " [strTitle=" + strTitle + ", strAuthor=" + strAuthor + ", year=" + year + ", pages=" + pages + "]";

}

/\*

@Override

public int hashCode() {

final int prime = 31;

int result = 1;

result = prime \* result + pages;

result = prime \* result + ((strAuthor == null) ? 0 : strAuthor.hashCode());

result = prime \* result + ((strTitle == null) ? 0 : strTitle.hashCode());

result = prime \* result + year;

return result;

}

\*/

// 1.5 Метод за еквивалентност;

@Override

public boolean equals(Object obj) {

// if (this == obj)

// return true;

// if (obj == null)

// return false;

// if (getClass() != obj.getClass())

// return false;

// Book other = (Book) obj;

// if (pages != other.pages)

// return false;

// if (strAuthor == null) {

// if (other.strAuthor != null)

// return false;

// } else if (!strAuthor.equals(other.strAuthor))

// return false;

// if (strTitle == null) {

// if (other.strTitle != null)

// return false;

// } else if (!strTitle.equals(other.strTitle))

// return false;

// if (year != other.year)

// return false;

// return true;

//

//

Book other = (Book) obj;

return pages == other.pages

&& year == other.year

&& strTitle.equals(other.strTitle)

&& strAuthor.equals(other.strAuthor);

}

// 1.6 Имплементация на интерфейса.

@Override

public int compareTo(Object obj) {// само за int

return this.year-((Book)obj).year;

}

}

class Library {

//2. Да се състави клас Library

//Частен масив от книги;

private Book[] books=new Book[] {new Book("","",0,0),

new Book("","",0,0),

new Book("","",0,0),

new Book("","",0,0),

new Book("","",0,0),

new Book("","",0,0),

new Book("","",0,0)};

//2.1 Експлицитен конструктор с име на файл - public Library(String filename)

public Library(String fileName) {

//Формат на файла:

//<заглавие>SP<име на автор>SP<година на издаване>SP<страници>CR

// try {

// RandomAccessFile myFile = new RandomAccessFile(fileName,"r");

// String strLine;

// String[] res;

// int ind=0;

// while( (strLine=myFile.readLine()) != null){

// res=strLine.split(" ");

// Book oBook = new Book(

// res[0],

// res[1],

// Integer.valueOf(res[2]).intValue(),

// Integer.valueOf(res[3]).intValue()

// );

// books[ind++] = oBook;

// }

// myFile.close();

// }catch (IOException e) {

// e.printStackTrace();

// }

try {

Scanner iStream = new Scanner(new File(fileName));

int index = 0;

while(iStream.hasNext()) {

books[index++]=new Book(iStream.next(),iStream.next(), iStream.nextInt(),iStream.nextInt());

}

iStream.close();

} catch (FileNotFoundException e) {

e.printStackTrace();

}

}

//2.2 Метод за извеждане на масива на конзола;

public void showToCout() {

System.out.println(Arrays.toString(books));

}

//2.3 Метод за сортиране на масива по година на издаване;

public void sortArray() {

Arrays.sort(books);

}

//2.4 Метод, който изчислява и връща общия брой страници на книгите от масива;

public int calcSumPages() {

int res=0;

for(Book current : books) {

res += current.pages;

}

return res;

}

//2.5 Метод, който изчислява и връща броя книги, чиито страници са повече от зададен параметър;

public int greatherPages(int from) {

int res=0;

for(Book current : books) {

if( current.pages>from) {

res ++;

}

}

return res;

}

//2.6 Метод, който изчислява и върща колко книги от даден автор има в масива;

public int calcBooksByAuthor(String from) {

int res=0;

for(Book current : books) {

if( current.strAuthor.equals(from)) {

res ++;

}

}

return res;

}

//2.7 Метод, който изчислява и връща средната стойност на страниците на книгите от масива;

public double calcAvgPages() {

double res=calcSumPages();

return books.length>0?res/books.length:0.0;

}

//2.8 Метод, който изчислява и връща броя книги, които са издадени след определена година;

public int calcBooksAfterYear(int from) {

int res=0;

for(Book current : books) {

if( current.year>from) {

res ++;

}

}

return res;

}

//2.9 Метод, който намира и връща заглавието на книгата, издадена най-късно.

public String calcBookAfterYear() {

String res="";

int from = 0;

for(Book current : books) {

if( current.year>from) {

from = current.year;

res = current.strTitle;

}

}

return res;

}

}

public class MainClass {

public static void main(String[] args) {

Library test = new Library("books.txt");

test.sortArray();

test.showToCout();

System.out.println("calcAvgPages : "+test.calcAvgPages());

System.out.println("calcBookAfterYear : "+test.calcBookAfterYear());

System.out.println("calcBooksAfterYear 2011 : "+test.calcBooksAfterYear(2011));

System.out.println("calcBooksByAuthor author1 : "+test.calcBooksByAuthor("author1"));

System.out.println("calcSumPages : "+test.calcSumPages());

System.out.println("greatherPages 105 : "+test.greatherPages(105));

}

}