Contents

[Code: 2](#_Toc525159432)

[ConsoleApp.java: 2](#_Toc525159433)

[Book.java: 9](#_Toc525159434)

[Subject.java: 11](#_Toc525159435)

[Screenshots: 14](#_Toc525159436)

[ Adding a Subject and Searching a subject: 14](#_Toc525159437)

[ . Adding a Book & Searching a Book: 15](#_Toc525159438)

[ Delete a Book & then search for it: 16](#_Toc525159439)

[ Deleting a Subject, and then searching for it to ensure it does NOT exist: 17](#_Toc525159440)

[ Exiting the app: 18](#_Toc525159441)

# Code:

## ConsoleApp.java:

package com.bhat.fsd.corejava;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.time.LocalDate;

import java.util.ArrayList;

import java.util.HashSet;

import java.util.Scanner;

import java.util.Set;

public class ConsoleApp{

public static File binaryFile = new File("C:/Users/bhats/Desktop/FSD-Content/Assignments/07-CoreJava\_1/SubjectsAndBooks");

public static ArrayList<Object> contents = new ArrayList<>();

public static void main(String[] args){

Scanner scnr = null;

try {

readFromFile();

scnr = new Scanner (System.in);

System.out.println(" Welcome to the Console App!");

char mainMenuOption = 'X';

while (mainMenuOption != 'g' || mainMenuOption != 'G' ){

System.out.println(" Select one of the options below : \n" +

"a.Add a Subject\n" +

"b.Add a Book\n" +

"c.Delete a Subject\n" +

"d.Delete a book\n" +

"e.Search for a book\n" +

"f.Search for a subject\n" +

"g.Exit");

mainMenuOption = scnr.nextLine().charAt(0);

switch(mainMenuOption) {

case 'a':

System.out.println("You have chosen to add a Subject");

addNewSubject(scnr);

break;

case 'b':

System.out.println("You have chosen to Add a Book");

addNewBook(scnr);

break;

case 'c':

System.out.println("Delete a Subject \n");

deleteSubject(scnr);

break;

case 'd':

System.out.println("Delete a Book\n");

deleteBook(scnr);

break;

case 'e':

System.out.println("Search for a Book");

Book b = searchBook (scnr) ;

break;

case 'f':

System.out.println("Search for a Subject\n");

Subject s = searchSubject (scnr);

break;

case 'g':

System.out.println ("Exiting");

System.exit (0);

break;

default:

System.out.println("Incorrect Choice- Please select one of the choices from the menu");

break;

}

}

} catch (Exception e) {

e.printStackTrace();

}finally{

if (scnr != null)

scnr.close();

}

}

public static void readFromFile(){

contents=new ArrayList<Object>();

try(ObjectInputStream ios = new ObjectInputStream(new FileInputStream(binaryFile))){

while(ios.available() > 0){

Object obj = ios.readObject();

if( obj instanceof Book || obj instanceof Subject){

contents.add(obj);

}

}

} catch (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

public static void writeToFile(){

try(FileOutputStream fos = new FileOutputStream(binaryFile);

ObjectOutputStream oos = new ObjectOutputStream(fos);) {

for( Object obj : contents){

oos.writeObject(obj);

}

} catch (IOException e) {

e.printStackTrace();

}

}

private static void addNewBook(Scanner scnr) {

System.out.println("Enter the Book ID ");

int bookId = Integer.parseInt(scnr.nextLine());

System.out.println("Enter the Book Title:");

String bookTitle = scnr.nextLine();

System.out.println("Enter the Book volume ");

int vol = Integer.parseInt(scnr.nextLine());

System.out.println("Enter the Book Price : ");

Double price = Double.parseDouble(scnr.nextLine());

System.out.println("Enter the Date of publishing in DD-MHM-YY format - For eg. 2007-12-03 ");

LocalDate dt = LocalDate.parse(scnr.nextLine());

System.out.println("Entered Values : Book(" + bookId + "," + bookTitle + "," + vol + "," + price + "," + dt);

Book newBook = new Book(bookId, bookTitle, price, vol, dt);

contents.add(newBook);

writeToFile();

}

private static void addNewSubject(Scanner scnr) {

System.out.println("Enter the Subject ID: ");

int id = Integer.parseInt(scnr.nextLine());

System.out.println("Enter the Subject Title:");

String title = scnr.nextLine();

System.out.println("Enter the Subject Duration:");

int duration = Integer.parseInt(scnr.nextLine());

Set<Book> ref = new HashSet<Book>();

String input = "";

while (!"EXIT".equalsIgnoreCase(input)) {

System.out.println("To add References, Enter the Book Id OR Book Title OR type EXIT to exit");

input = scnr.nextLine();

Book b1 = new Book();

try {

int refId = Integer.parseInt(input);

b1.setBookId(refId);

} catch (NumberFormatException ne) {

b1.setTitle(input);

}

ref.add(b1);

}

Subject s1 = new Subject(id, title, duration, ref);

contents.add(s1);

writeToFile();

}

private static void deleteBook (Scanner scnr){

Book b = searchBook(scnr);

if ( b != null) {

contents.remove(b);

writeToFile();

}

}

private static Book searchBook (Scanner scnr) {

System.out.println("Enter the Book Id Or Book Title : ");

String input=scnr.nextLine();

Book b1 = new Book();

boolean searchById = false;

try{

int refId = Integer.parseInt(input);

b1.setBookId(refId);

searchById = true;

}catch( NumberFormatException ne) {

b1.setTitle(input);

}

boolean found = false;

for (Object obj : contents) {

if (obj instanceof Book) {

Book l1 = (Book) obj;

if ( (searchById && l1.getBookId() == b1.getBookId()) ||

(!searchById && l1.getTitle().equalsIgnoreCase(b1.getTitle())) ){

b1 = l1;

found = true;

}

}

}

if(found) {

System.out.println("Details Of Book Found : ID:" + b1.getBookId() + "; Title:" + b1.getTitle() + "; Price" + b1.getPrice() + "; Volume:" + b1.getVolume() + "; PublishDate" + b1.getPublishDate() +"\n");

return b1;

}

else{

System.out.println("Book not Found");

return null;

}

}

private static Subject searchSubject (Scanner scnr) {

System.out.println("Enter the Subject Id Or Title : ");

String input=scnr.nextLine();

boolean searchById = false;

int refId = 0;

Subject subjFound = null;

try{

refId = Integer.parseInt(input);

searchById = true;

}catch( NumberFormatException ne) {

}

for (Object obj : contents) {

if (obj instanceof Subject) {

Subject l1 = (Subject) obj;

if ( (searchById && l1.getSubjectId() == refId) ||

(!searchById && l1.getSubtitle().equalsIgnoreCase(input)) ){

subjFound = l1;

}

}

}

if(subjFound != null) {

System.out.println("Details Of Subject Found : ID:" + subjFound.getSubjectId() + "; Title:" + subjFound.getSubtitle() + "; Duration " +subjFound.getDurationInHours() + ";Size of Ref list:" + subjFound.getReferences().size());

return subjFound;

}

else{

System.out.println("Subject not Found");

return null;

}

}

private static void deleteSubject (Scanner scnr) {

Subject s = searchSubject(scnr);

if(s != null) {

contents.remove(s);

writeToFile();

}

}

}

## Book.java:

package com.bhat.fsd.corejava;

import java.io.Serializable;

import java.time.LocalDate;

public class Book implements Serializable{

private static final long serialVersionUID = 1L;

private long bookId;

private String title;

private Double price;

private int volume;

private LocalDate publishDate;

public Book(){

}

public Book(long id, String title, Double priceToSet, int vol, LocalDate dt){

this.bookId = id;

this.title = title;

this.price = priceToSet;

this.volume = vol;

this.publishDate = dt;

}

public long getBookId(){

return this.bookId;

}

public void setBookId(long id){

this.bookId = id;

}

public void setTitle(String t){

this.title = t;

}

public String getTitle(){

return this.title;

}

public Double getPrice(){

return this.price;

}

public void setPrice(Double p){

this.price = p;

}

public int getVolume(){

return this.volume;

}

public void setVolume(int v){

this.volume = v;

}

public LocalDate getPublishDate(){

return this.publishDate;

}

public void setPublishDate(LocalDate dt){

this.publishDate = dt;

}

}

## Subject.java:

package com.bhat.fsd.corejava;

import java.io.Serializable;

import java.util.HashSet;

import java.util.Set;

public class Subject implements Serializable{

private static final long serialVersionUID = 1L;

private long subjectId;

private String subtitle;

private int durationInHours;

private Set<Book> references = new HashSet<Book>();

public Subject(long id, String title, int duration, Set<Book>ref){

this.subjectId = id;

this.subtitle = title;

this.durationInHours = duration;

this.references = ref;

}

public long getSubjectId(){

return this.subjectId;

}

public void setSubjectId(long id){

this.subjectId = id;

}

public String getSubtitle(){

return this.subtitle;

}

public void setSubtitle(String newTitle){

this.subtitle = newTitle;

}

public int getDurationInHours(){

return this.durationInHours;

}

public void setDurationInHours(int t){

this.durationInHours = t;

}

public Set<Book> getReferences(){

return this.references;

}

public void setReferences(Set<Book> newRef){

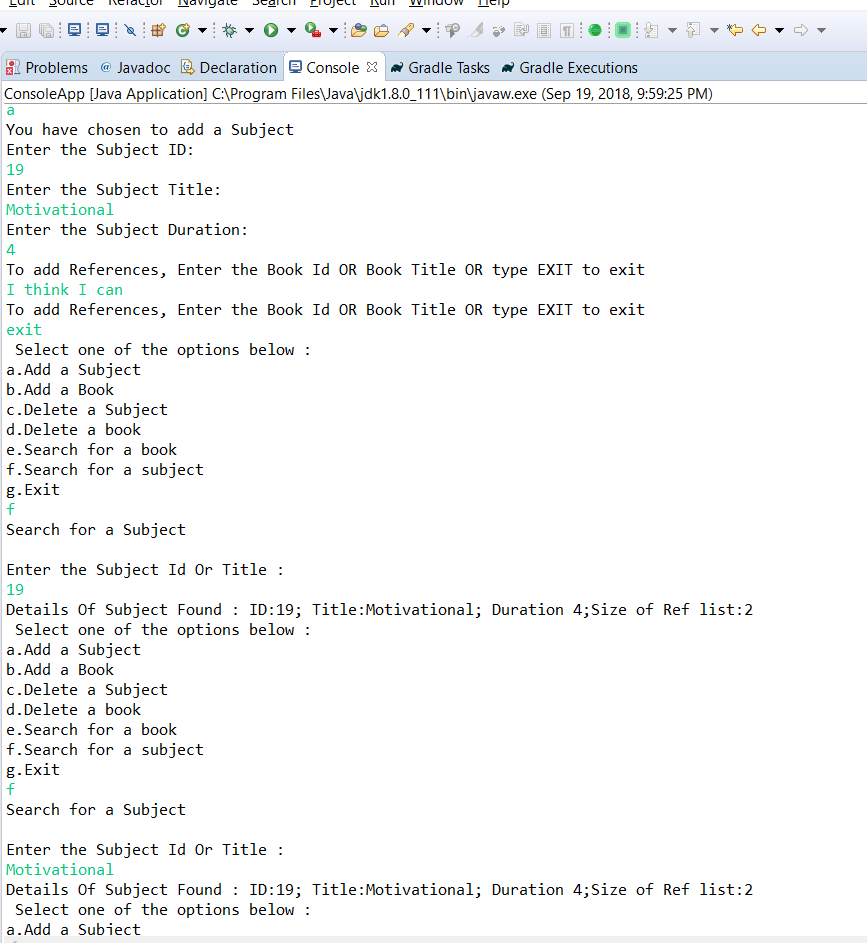
this.references = newRef;

}

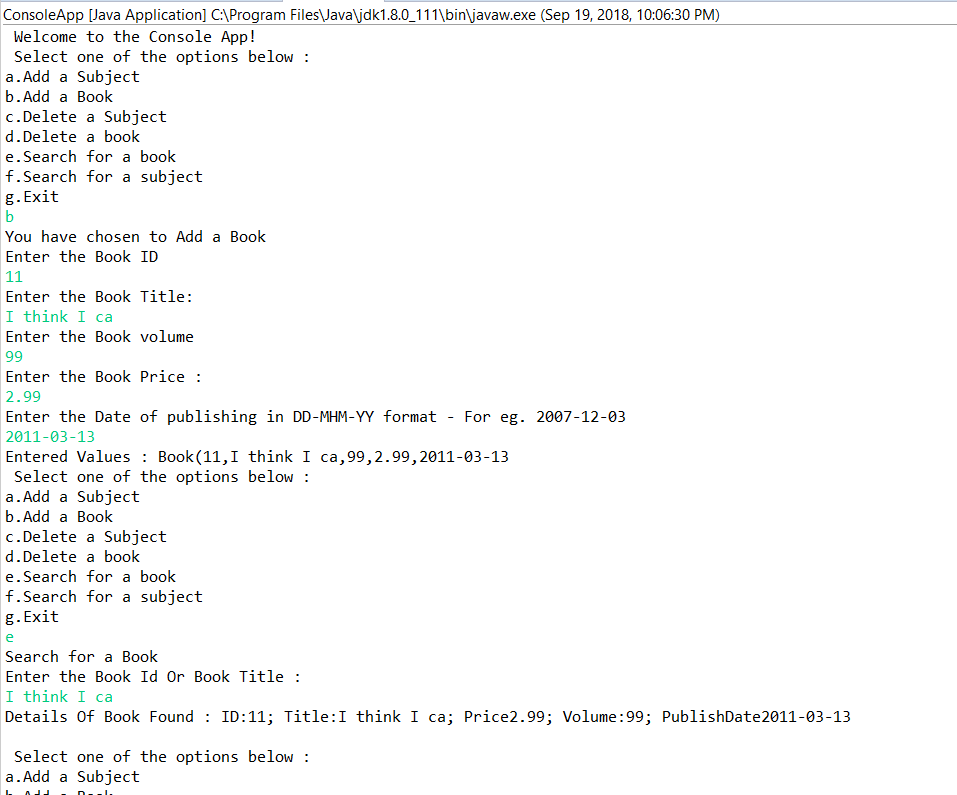
}

# Screenshots:

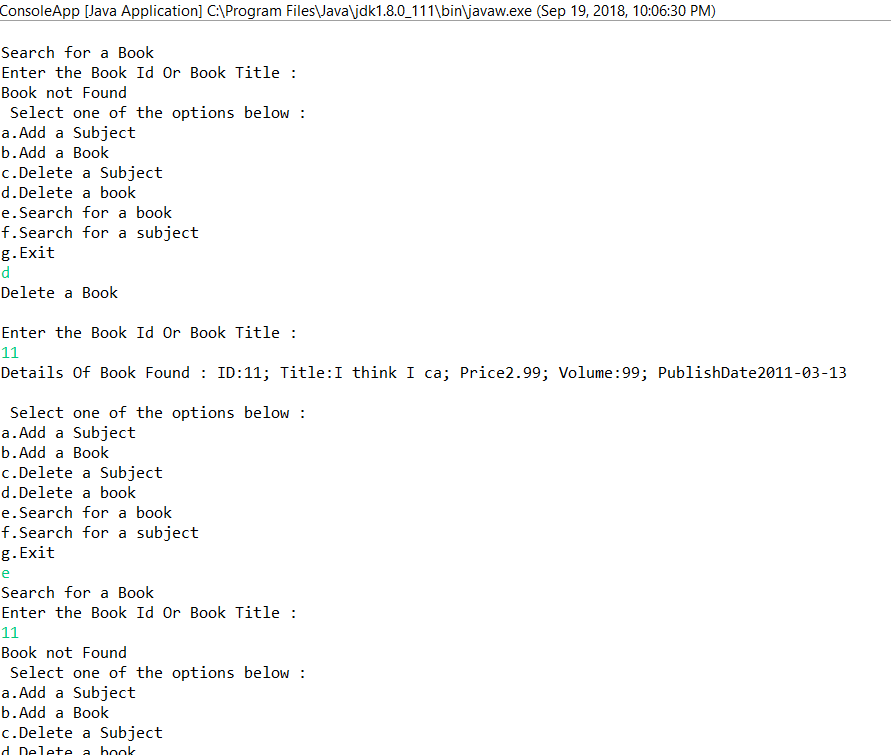
## Adding a Subject and Searching a subject:



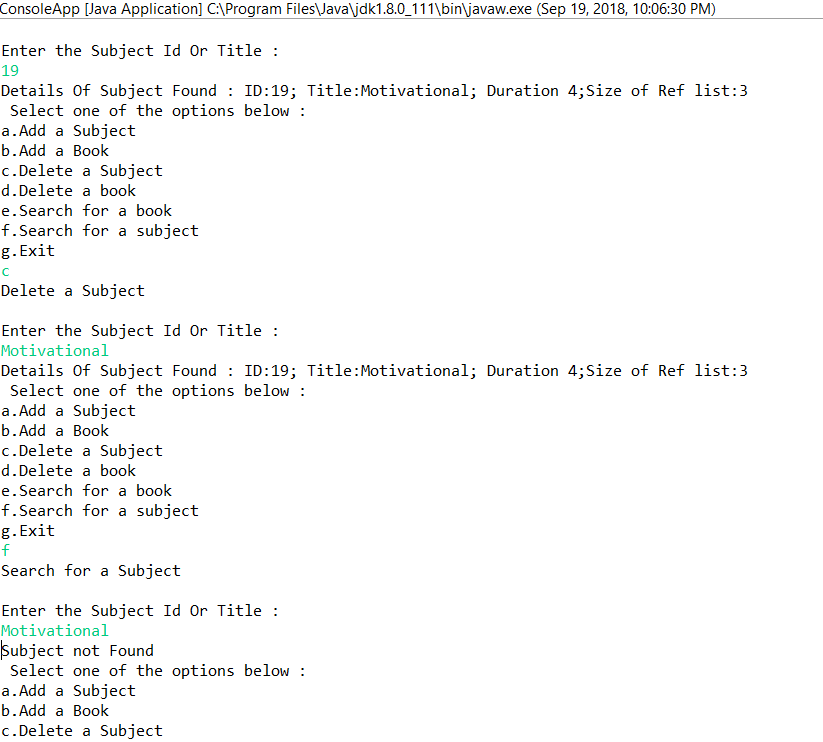
## . Adding a Book & Searching a Book:



## Delete a Book & then search for it:



## Deleting a Subject, and then searching for it to ensure it does NOT exist:



## Exiting the app:

