AssignmentId: 10\_Spring\_Assignment1

Code Snippets:

Book.java

package main.java;

import java.io.Serializable;

import java.time.LocalDate;

public class Book implements Serializable,Comparable<Book>{

long bookId;

String title;

double price;

Integer volume;

LocalDate publishDate;

@Override

public String toString() {

return "Book [bookId=" + bookId + ", title=" + title + ", price="

+ price + ", volume=" + volume + ", publishDate=" + publishDate

+ "]";

}

public Book(long bookId, String title, double price, Integer volume,

LocalDate publishDate) {

super();

this.bookId = bookId;

this.title = title;

this.price = price;

this.volume = volume;

this.publishDate = publishDate;

}

public Book(){

}

public long getBookId() {

return bookId;

}

public void setBookId(long bookId) {

this.bookId = bookId;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public Integer getVolume() {

return volume;

}

public void setVolume(Integer volume) {

this.volume = volume;

}

public LocalDate getPublishDate() {

return publishDate;

}

public void setPublishDate(LocalDate publishDate) {

this.publishDate = publishDate;

}

@Override

public int compareTo(Book b) {

if(this.title.equals(b.getTitle()))

return 1;

else

return -1;

}

}

**Subject.java**

package main.java;

import java.io.Serializable;

import java.util.HashSet;

import java.util.Set;

public class Subject implements Serializable,Comparable<Subject> {

long subjectId;

String subtitle;

int durationInHours;

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

public Subject(){

}

public long getSubjectId() {

return subjectId;

}

public void setSubjectId(long subjectId) {

this.subjectId = subjectId;

}

public String getSubtitle() {

return subtitle;

}

public void setSubtitle(String subtitle) {

this.subtitle = subtitle;

}

public int getDurationInHours() {

return durationInHours;

}

public void setDurationInHours(int durationInHours) {

this.durationInHours = durationInHours;

}

public Set<Book> getReferences() {

return references;

}

public void setReferences(Set<Book> references) {

this.references = references;

}

public Subject(long subjectId, String subtitle, int durationInHours,

Set<Book> references) {

super();

this.subjectId = subjectId;

this.subtitle = subtitle;

this.durationInHours = durationInHours;

this.references = references;

}

@Override

public String toString() {

return "Subject [subjectId=" + subjectId + ", subtitle=" + subtitle

+ ", durationInHours=" + durationInHours + ", references="

+ references + "]";

}

@Override

public int compareTo(Subject b) {

if(this.subtitle.equals(b.getSubtitle()))

return 1;

else

return -1;

}

}

**BookByDate.java**

package main.java;

import java.io.Serializable;

import java.time.LocalDate;

import java.util.Comparator;

public class BookbyDate implements Comparator<Book>{

/\*\*

\*

\*/

private static final long serialVersionUID = 1L;

@Override

public int compare(Book b1,Book b2) {

int value=0;

if(b1.getPublishDate().isAfter(b2.getPublishDate())){

value=1;

}

if(b1.getPublishDate().isBefore(b2.getPublishDate())){

value=-1;

}

if(b1.getPublishDate().isEqual(b2.getPublishDate())){

value=0;

}

return value;

}

}

**AppTest.java**

package main.java;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.time.Instant;

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

import java.util.ArrayList;

import java.util.Collection;

import java.util.Collections;

import java.util.Comparator;

import java.util.Date;

import java.util.HashMap;

import java.util.HashSet;

import java.util.List;

import java.util.Map;

import java.util.Map.Entry;

import java.util.Scanner;

import java.util.Set;

import java.util.TreeMap;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class AppTest {

public static void main(String[] args) throws IOException,ClassNotFoundException {

ApplicationContext ac=new ClassPathXmlApplicationContext("applicationcontext.xml");

Book book=(Book) ac.getBean("book");

Scanner sc = new Scanner(System.in);

int ch, k, v;

String t, d;

double p;

while (true) {

System.out.println("01. Add a Book");

System.out.println("02. Add a Subject");

System.out.println("03. Delete a Book");

System.out.println("04. Delete a Subject");

System.out.println("05. Search for a book");

System.out.println("06. Search for a subject");

System.out.println("07. sort book by title");

System.out.println("08. sort subject by subject title");

System.out.println("09. sort books by publish date");

System.out.println("10. Exit");

System.out.print("Enter Your Choice : ");

ch = sc.nextInt();

switch (ch) {

case 1:

System.out.println("01. Add a Book");

addBook();

break;

case 2:

System.out.println("02. Add a Subject");

addSubject();

break;

case 3:

System.out.println("03. Delete a Book");

Scanner sc3 = new Scanner(System.in);

System.out.println("Enter Book tittle");

String tittle1 = sc3.next();

deleteBook(tittle1);

break;

case 4:

System.out.println("03. Delete a Subject");

Scanner sc4 = new Scanner(System.in);

System.out.println("Enter Book tittle");

String tittle3 = sc4.next();

deleteSubject(tittle3);

break;

case 5:

System.out.println("05. Search for a book");

try {

Scanner sc6 = new Scanner(System.in);

System.out.println("Enter Book tittle");

String tittle5 = sc6.next();

Set<Book> sb1 = retriveBook(tittle5);

System.out.println("searched book =====" + sb1);

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

break;

case 6:

System.out.println("06. Search for a subject");

try {

// Set<Book> sb1=retriveBook("www");

Scanner sc2 = new Scanner(System.in);

System.out.println("Enter subject tittle");

String tittle = sc2.next();

retriveSubject(tittle);

// System.out.println("sb1====="+sb1);

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

break;

case 7:

sortBookbyTitle();

break;

case 8:

sortSubjectbyTitle();

break;

case 9:

sortbookBypublishDate();

break;

case 10:

System.out.println("Exited");

System.exit(0);

default:

System.out.println("Wrong Entry");

}

}

}

private static void sortbookBypublishDate() throws IOException, ClassNotFoundException {

Set<BookbyDate> sb = new HashSet<BookbyDate>();

FileInputStream fint = new FileInputStream("D:\\f.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Map<String, Book> sbook = (Map<String, Book>) oint.readObject();

System.out.println("Before sort by date:sbook is:"+sbook);

Collection<Book> collection=sbook.values();

List<Book> listbook=new ArrayList<Book>(collection);

Collections.sort(listbook, new BookbyDate());

System.out.println("After sort by date:listbook is:"+listbook);

//TreeMap<String,BookbyDate> tm=new TreeMap<String,BookbyDate>(sbook);

//System.out.println("Set of sorted books by title ----------" + tm);

}

private static void sortSubjectbyTitle() throws IOException, ClassNotFoundException {

Set<Subject> sb = new HashSet<Subject>();

FileInputStream fint = new FileInputStream("D:\\f.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Map<String, Subject> subject = (Map<String,Subject>) oint.readObject();

TreeMap<String,Subject> tm1=new TreeMap<String,Subject>(subject);

System.out.println("Set of sorted books by title ----------" + tm1);

}

private static void sortBookbyTitle() throws IOException, ClassNotFoundException {

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

FileInputStream fint = new FileInputStream("D:\\f.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Map<String, Book> sbook = (Map<String, Book>) oint.readObject();

TreeMap<String,Book> tm=new TreeMap<String,Book>(sbook);

System.out.println("Set of sorted books by title ----------" + tm);

}

private static void deleteSubject(String subjecttittle)

throws ClassNotFoundException, IOException {

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

Map<Integer, Subject> map = new HashMap<Integer, Subject>();

FileInputStream fint = new FileInputStream("D:\\f2.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Subject subject = (Subject) oint.readObject();

System.out.println("Before remove subject>>>>>>>>>>>>>>>>>>"

+ subject.toString());

if (subject.getSubtitle().equals(subjecttittle)) {

setbook = subject.getReferences();

}

for (Book book : setbook) {

if (book.getTitle().equals(subjecttittle)) {

setbook.remove(book);

}

}

System.out.println("After remove subject>>>>>>>>>>>>>>>>>>" + setbook);

}

private static void deleteBook(String title) throws ClassNotFoundException,

IOException {

Set<Book> s2 = retriveBook(title);

System.out.println("before remove " + s2);

// Scanner sc=new Scanner(System.in);

// System.out.println("enter bookid to delete");

// int bid=sc.nextInt();

for (Book book : s2) {

s2.remove(book);

}

System.out.println("after remove" + s2);

}

private static void addSubject() throws IOException {

List<Subject> s1 = new ArrayList<Subject>();

Scanner sc0 = new Scanner(System.in);

System.out.println("enter total entry--------\n");

int total = sc0.nextInt();

// Book b1=null;

for (int i = 0; i < total; i++) {

Scanner sc = new Scanner(System.in);

System.out.println("enter subjectId");

long k = sc.nextInt();

System.out.println("enter subtitle");

String t = sc.next();

System.out.println("enter durationInHours");

int p = sc.nextInt();

Set<Book> mapsubject = null;

try {

mapsubject = retriveBook(t);

System.out.println("mapsubject>>>>>>>>>>>>>" + mapsubject);

// mapsubject = getBook(t);

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

FileOutputStream fout = new FileOutputStream("D:\\f2.txt");

ObjectOutputStream out = new ObjectOutputStream(fout);

out.writeObject(new Subject(k, t, p, mapsubject));

out.flush();

System.out.println("success");

}

}

private static Set<Book> getBook(String t) throws IOException,

ClassNotFoundException {

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

FileInputStream fint = new FileInputStream("D:\\f.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Map<String, Book> sbook = (Map<String, Book>) oint.readObject();

System.out.println("Set of books----------" + sbook);

Book book = new Book();

Set<String> set = sbook.keySet();

for (String str : set) {

if (set.contains(t)) {

sb.add(sbook.get(str));

}

}

fint.close();

oint.close();

return sb;

}

private static Set<Book> retriveSubject(String tittle)

throws ClassNotFoundException, IOException {

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

Set<Book> sb1 = new HashSet<Book>();

FileInputStream fint = new FileInputStream("D:\\f2.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Subject subject = (Subject) oint.readObject();

System.out.println("subject----------" + subject.toString());

if (subject.getSubtitle().equals(tittle)) {

System.out.println("retriveSubject>>>>>>>>>>>>>>"

+ subject.getSubjectId() + "----"

+ subject.getDurationInHours() + "--------"

+ subject.getSubtitle() + "----" + subject.getReferences());

}

fint.close();

oint.close();

return sb;

}

private static Set<Book> retriveBook(String tittle)

throws ClassNotFoundException, IOException {

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

FileInputStream fint = new FileInputStream("D:\\f.txt");

ObjectInputStream oint = new ObjectInputStream(fint);

Map<String, Book> sbook = (Map<String, Book>) oint.readObject();

System.out.println("Set of books----------" + sbook);

Book book = new Book();

Set<String> set = sbook.keySet();

for (String str : set) {

if (str.equals(tittle)) {

sb.add(sbook.get(str));

}

}

fint.close();

oint.close();

return sb;

}

public static Map<String, Book> addBook() throws IOException {

// Set<Book> s1 = new HashSet<Book>();

Map<String, Book> map = new HashMap<String, Book>();

Scanner sc0 = new Scanner(System.in);

System.out.println("enter total entry--------");

int total = sc0.nextInt();

// Book b1=null;

for (int i = 0; i < total; i++) {

Scanner sc = new Scanner(System.in);

System.out.println("enter bookid");

int k = sc.nextInt();

System.out.println("enter book title");

String t = sc.next();

System.out.println("enter book price");

double p = sc.nextDouble();

System.out.println("enter book volume");

int v = sc.nextInt();

System.out.println("enter book publish date");

String date = sc.next();

LocalDate d = LocalDate.parse(date);

map.put(t, new Book(k, t, p, v, d));

}

FileOutputStream fout = new FileOutputStream("D:\\f.txt");

ObjectOutputStream out = new ObjectOutputStream(fout);

out.writeObject(map);

out.flush();

System.out.println("success");

return map;

}

}

**applicationContext.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"*

*http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd"*>

<bean id=*"book"* class=*"main.java.Book"*></bean>

<bean id=*"subject"* class=*"main.java.Subject"*>

</bean>

</beans>

**Results:**

May 30, 2019 12:06:24 PM org.springframework.context.support.ClassPathXmlApplicationContext prepareRefresh

INFO: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@7c3df479: startup date [Thu May 30 12:06:24 IST 2019]; root of context hierarchy

May 30, 2019 12:06:24 PM org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions

INFO: Loading XML bean definitions from class path resource [applicationcontext.xml]

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 01

01. Add a Book

enter total entry--------

01

enter bookid

102

enter book title

physicsone

enter book price

450

enter book volume

1

enter book publish date

2017-01-01

success

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 05

05. Search for a book

Enter Book tittle

physicsone

Set of books----------{physicsone=Book [bookId=102, title=physicsone, price=450.0, volume=1, publishDate=2017-01-01]}

searched book =====[Book [bookId=102, title=physicsone, price=450.0, volume=1, publishDate=2017-01-01]]

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 01

01. Add a Book

enter total entry--------

1

enter bookid

103

enter book title

physicsone

enter book price

457

enter book volume

2

enter book publish date

2015-01-01

success

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 02

02. Add a Subject

enter total entry--------

02

enter subjectId

15

enter subtitle

physicsone

enter durationInHours

10

Set of books----------{physicsone=Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]}

mapsubject>>>>>>>>>>>>>[Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]]

success

enter subjectId

12

enter subtitle

physicstwo

enter durationInHours

21

Set of books----------{physicsone=Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]}

mapsubject>>>>>>>>>>>>>[]

success

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 07

Set of sorted books by title ----------{physicsone=Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]}

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 08

Set of sorted books by title ----------{physicsone=Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]}

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 09

Before sort by date:sbook is:{physicsone=Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]}

After sort by date:listbook is:[Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]]

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 03

03. Delete a Book

Enter Book tittle

102

Set of books----------{physicsone=Book [bookId=103, title=physicsone, price=457.0, volume=2, publishDate=2015-01-01]}

before remove []

after remove[]

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 04

03. Delete a Subject

Enter Book tittle

physicstwo

Before remove subject>>>>>>>>>>>>>>>>>>Subject [subjectId=12, subtitle=physicstwo, durationInHours=21, references=[]]

After remove subject>>>>>>>>>>>>>>>>>>[]

01. Add a Book

02. Add a Subject

03. Delete a Book

04. Delete a Subject

05. Search for a book

06. Search for a subject

07. sort book by title

08. sort subject by subject title

09. sort books by publish date

10. Exit

Enter Your Choice : 10

Exited