**A simple application for Book lending shop**

**Overview**

You are required to write a program to manage a small book lending shop that lends books to readers.

The main intention of this section is to allow you to demonstrate your understanding of the concepts of encapsulation and inheritance. You are encouraged to add comments to explain your program design. Where the specification is subject to interpretation, you may make any reasonable assumptions but you are required to justify these using comments.

The following notes may be helpful to you:

* Use java.util.Date class to represent date objects (rent dates, return dates).
* new Date() returns the current date and time.
* Assume that a utility class named DateUtil is provided with a static method getDaysDiff(Date d1, Date d2) which returns the number of elapsed days from d1 to d2. Use this method to compute the rental charge.
  + Example: **int days = DateUtil.getDaysDiff(rentDate, returnDate);**
* DateUtil.newDate(“15/5/2010”) returns the Date object that represents 15/05/2010.
* DateUtil.toString(d) takes a Date object as argument and returns a String representation of the date in the format dd/mm/yyy, for example 15/05/2010.
* Write **S.o** to mean **System.out**, so you can write **S.o.println();** instead of **System.out.println();**

**Section A: Class Book (10 marks)**

This class should provide instance variables to store Book**ID**, Book Name, and **book lent fee (daily basis) of each book**. This class must also provide a method to compute lending charge based on the lending date and book return date. Additional instance variables are needed to store the availability status of the book (available/lent/reserved), customer name and contact details together with the borrow date when the book is lent.

1. Write a class named **Book** with instance variables for book ID, book name, lent fee, availability status, customer name, contact details, and lent date. (1.5 marks)

public class Book {

String bookID;

String bookName;

double lentFee;

String status;

String custName;

String contactDetails;

Date lentDate;

1. Provide a constructor taking as arguments values for book ID, Book name, and lent fee. Initialize availability status of book to “available”.

(1.5 marks)

Book(String ID, String name, double dailyFee){

bookID=ID;

bookName=name;

lentFee=dailyFee;

status="Avaliable";

}

1. Provide accessor methods to book name and availability status
   1. mark)

public String getBookName(){

return bookName;

}

public String getStatus(){

return status;

}

1. Provide a method named **lend()** taking as arguments values for **customer name, contact details and lent date**. This method should display **an appropriate error message** if the book is not available for rent.
   1. marks)

public void lend(String name, String contact, Date lentDate){

if(status.equals("Available")){

status="Rented";

custName=name;

contactDetails=contact;

this.lentDate=lentDate;}

else

{

System.out.println("sorry book is not available");

}

}

1. Provide a method named **print()** that prints all necessary details of this book object. Also print the rental details if the book is currently rented.

(1 mark)

public void print(){

System.out.print("Book ID :"+bookID+" Book Name:"+bookName+" lent fee :"+lentFee);

if(status.equals("lent"))

System.out.println(" Status :"+status+"Customer Name :"+custName+ " contact Details :"+contactDetails +" lent Date:"+lentDate)

}

1. Provide a method named **returnBook()** taking one Date argument for **return-date** and returning a double value, which is the rental charge. Rental charge for book is the multiplication of the daily lent Fee and the number of rental days.

Hint: use the DateUtil class as mentioned in the beginning. This method must also throw error messages if return date or invalid status issues are found. (3 marks)

public double returnBook(Date returnDate){

int days = Dateutil.getDaysDiff(rentedDate,returnDate);

if(days <0)

System.out.println("Invalid attempt – return date should be");

else if(days==0) {

status ="A";

return 1\*dailyRate;

}

else if(getStatus().equals("Available"))

System.out.println ("Invalid attempt – Book is available");

else {

status = "Available";

return days\*dailyRate;

}

}

**2. Writing an BookTest class (15 marks)**

In this section you need to write an application that creates, stores, and uses a collection of Bike objects. You will need to write code for the following in the main method:

1. Define an array of books references that can store up to four objects (elements). (1 mark)

Book[] books= new Book[100];

1. Create the book objects specified below and store them in the array of books references mentioned above. You should pass the values shown below as arguments to the constructor when creating each of the objects. ( 4 marks)

|  |  |  |
| --- | --- | --- |
| **Book Id** | **Book Name** | **Lent fee(daily) (in $)** |
| 1 | C++ | 2.30 |
| 2 | Programming in Java | 1.90 |
| 3 | Photo Shop for design | 2.50 |
| 4 | Cinderella | 2.00 |

B

books[0]= new Book("1", "C++", 3.90);

books[1]= new Book("2", " Programming in Java ", 1.90);

books[2]= new Book("3", " Photo Shop for design ", 2.50);

books[3]= new Book("4", " Cinderella ", 2.00)

1. Lend a book: Get the customer name, contact details to lend a book. Rent date must be today. Implement an appropriate method to execute the rent operation. Hint: use DateUtil class as mentioned in the beginning ( 3 marks)

Scanner sc= new Scanner(System.in);

System.out.println("Enter the customer name :");

String name=sc.nextLine();

System.out.println("Enter the customer contact details :");

String contact=sc.nextLine();

Date rentDate = new Date();

for(int i=0; i<4; i++){

bikes[i].rent(name, contact, rentDate);

}