به نام خدا

|  |  |  |  |
| --- | --- | --- | --- |
| مدرس: مظفر بگ محمدي | دانشگاه ايلام | ترم اول سال تحصيلي 97-96 | سری هشتم تمرینات برنامه‌نویسی جاوا |

هدف این تمرین آشنایی بیشتر با واسط و استثنا و انجام یک تمرین نسبتا پیچیده است.

1. **واسط و استثنا**: در این تمرین شما باید یک فروشگاه مجازی کتاب درست کنید. فروشگاه شما دارای لیستی از مشتریان و لیستی از کتابهای موجود است. هر مشتری دارای یک id یکتا، کارت اعتباری، اسم، آدرس و کلمه ی عبور است. آدرس شامل اسم خیابان ، شهر، استان و کد پستی است. کارت اعتباری دارای سقف اعتبار برابر 500000 تومان می باشد. با توجه به واسطهای زیر کلاسهای فروشگاه، مشتری، آدرس و کارت اعتباری خرید را پیاده کنید. از واسطها و کلاسهای زیر استفاده کنید.

دقت کنید که در هنگام فراخوانی متد buy اگر موجودی کتاب کافی نباشد، باید استثناء NotEnoughBook تولید شود. اگر موجودی کارت اعتباری مشتری کافی نباشد، باید استثناء NotEnoughMoney تولید شود.

interface Store {

public void register(Customer customer);

public void remove(Customer customer);

public Customer getCustomer(int id);

}

//----

interface BookStore extends Store {

// invoke this method when a customer tries to buy books.

public boolean buy(Customer customer, Book book, int amount);

// The BookStore tries to get more books by this methods.

public void importBooks(Book book, int amount);

// get all the books with the specified title from the book list

// we assume no repeated book title in the book list.

public Book getBooks(String title);

// get the amount with the specified book title.

public int getBookAmount(String title);

// get all the books in the book list.

public Book[] getAllBooks();

}

//-------

class MyStore implements BookStore {

private MyCustomer [] customers;

private MyBook [] books;

//how many books are available for each title

private int [] numberOfBook;

}

//-----

interface Book {

// get the title of this book.

public String getTitle();

// get the date of publishing this book.

public java.util.Date getDate();

// get the edition of this book.

public int getEdition();

// get the brief description of this book.

public String getDescription();

// get the price of this book.

public float getPrice();

}

//----

public class MyBook implements Book{

private String title;

private java.util.Date date;

private int edition;

private String description;

private float price;

public MyBook (String title, java.util.Date date, int edition, String desc, float price ) {

// a constructor to create a new MyBook.

}

}

//-----

interface Person {

public String getName();

public Address getAddress();

}

//-----

interface Customer extends Person {

public int getID();

public Credit getCredit();

public void setPassword(String passwd);

}

//-----

class MyCustomer implements Customer{

private int accountID; // an unique ID given by the store

private String name; // the name of this customer

private MyAddress address; // the address of this customer

private Credit credit; // the credit of this customer

private String passwd; // the password set by this customer

public Customer(String name, Address addr) {

// a constructor to create a new customer.

}

public void setAddress(Address addr) {

// set a new address.

}

protected void setID(int i) {

// set the id to i

}

protected void setCredit(Credit credit) {

// set the credit to the input credit.

}

public boolean equals(Customer customer) {

// check whether this customer is same as customer.

}

public boolean makePurchase(int amount) {

/\*

buy by debit.

return true if the debit of this customer is less than the

maximum limitation, otherwise return false.

\*/

}

}

//----

interface Address {

public String getStreetName();

public String getCity();

public String getState();

public String getZipCode();

public String toString();

public boolean equals(Address addr);

}

// this class represents an address

class MyAddress implements Address {

private String streetName;

private String city;

private String state;

private String zipcode;

public Address(String stName, String city, String state,

String zipcode) {

// constructor

}

}

// this class represents credit

class Credit {

final public static int MAX\_AMOUNT = 500000;

private int debit;

public int getDebit() {

// get the current debit.

}

protected void addDebit(int amount) {

// update the debit, also make sure not bigger than the limitation.

}

}