سوال اول: سامانه فروش اینترنتی

در این تمرین سعی داریم تا یک سامانه مدیریت فروش یک فروشگاه (مانند دیجیکالا یا بامیلو) را مدل کنیم. با فرایند مدل‌سازی بطور مفصل در درس مهندسی نرم‌افزار آشنا خواهید شد ولی بصورت کلی در فرآیند مدل‌سازی ابتدا باید مدل‌ها را استخراج کرد و سپس هر مدل را در غالب یک یا چند کلاس پیاده‌سازی کرد.

همانطور که در ابتدا گفته شد در این تمرین می‌خواهیم یک فروشگاه را مدل کنیم. این فروشگاه قرار است یک سری کالا را در انبارهای خود نگهداری کند و به مشتریان بفروشد. مشتریان می‌توانند برای خرید کالا سفارش بدهند و آنرا ویرایش و ثبت کنند. علاوه بر این فروشگاه می‌تواند یک سری تخفیف به مشتریان خود بدهد تا آنها به خرید کردن از این فروشگاه بیشتر ترغیب شوند.

در طراحی این سامانه، شما باید مدیریت مشتریان، مدیریت انبارداری و مدیریت خریدهای مشتری را در حد ساده پیاده‌سازی کنید.

نکته: ممکن است برای پیاده‌سازی کلاس‌های خود نیازمند اضافه کردن توابع private یا public به کلاس‌ها باشید. توابع زیر صرفا توابعی هستند که باید پیاده‌سازی شوند.

برای مدلسازی مساله بالا نیازمند کلاس‌های زیر هستیم.

```
public class Shop{}
```

این کلاس برای مدلسازی فروشگاه بصورت کلی استفاده می‌شود.

```
public class Customer{}
```

این کلاس برای مدلسازی مشتریان فروشگاه استفاده می‌شود.

```
public class Good{}
```

این کلاس برای توصیف کالاهایی که در فروشگاه بفروش می‌رسد استفاده می‌شود.

```
public class Repository{}
```

این کلاس انبارهای فروشگاه را مدلسازی می‌کند.

```
public class Order{}
```

این کلاس برای مدلسازی درخواست خرید مشتری استفاده می‌شود.

```
public class Discount{}
```

این کلاس برای توصیف تخفیف‌هایی است که به مشتری اختصاص داده می‌شود.

متدهای کلاس‌های زیر بدین صورت هستند:

شرح متدهای کلاس Shop:

```
public Shop(String name);
```
سازنده: اسم فروشگاه را به عنوان ورودی دریافت می‌کند.

```
public void addCustomer(Customer c);
```
این تابع یک مشتری به لیست مشتریان عضو فروشگاه اضافه می‌کند.

```
public Customer[] getCustomers();
```
لیست مشتریان رستوران را بازمی‌گرداند.

```
public void addRepository(Repository r);
```
یک انبار به لیست انبارهای فروشگاه اضافه می‌کند.

```
public Repository[] getRepositories();
```
لیست انبارهای فروشگاه را بازمی‌گرداند.

```
public int getIncome();
```
مقدار سود فروشگاه را بازمی‌گرداند که در شروع کار فروشگاه ۰ است.

```
public void setIncome(int income);
```
سود فروشگاه را مقداردهی می‌کند.

```
public void addGood(Good g);
```
یک کالا جدید را به لیست کالاهای فروشگاه اضافه می‌کند.

```
public void increamentGood(Good g, int amount);
```
یک کالا را به تعداد ورودی به یک انبار اضافه می‌کند. افزودن کالا بدین صورت است که مخزن‌ها به ترتیب ظرفیت بررسی می‌شوند بصورت صعودی. هر انبار که بیش از amount ظرفیت خالی داشت کالا را به به آن انبار اضافه می‌کنیم.

```
public Good[] getGoods();
```
یک آرایه از کالاهای موجود در فروشگاه را باز می‌گرداند.

```
public void addDiscount(Discount d, Customer c);
```
به مشتری c تخفیف d را می‌دهد.

```
public HashMap<Good, Integer> getItemsSold();
```
یک hashmap بازمی‌گرداند حاوی تمام کالاها به عنوان key به همراه تعدادی که فروخته شده به عنوان value.

```
public void addDiscount(Discount discount);
```
discount را به لیست تخفیف‌های فروشگاه اضافه می‌کند.

متدهای کلاس Good:

```
public Good(String name, int ID, int price);
```
سازنده: اسم کالا، شناسه کالا و قیمت آنرا دریافت می‌کند.

```
public int getID();
```
شناسه کالا را باز می‌گرداند

```
public int getPrice();
```
قیمت کالا را باز می‌گرداند

```
public String getName();
```
نام کالا را باز می‌گرداند

متدهای کلاس Repository:

```
public Repository(int id, int capacity);
```
این متد سازنده شناسه و ظرفیت انبار را می گیرید.

```
public int getID();
```
این تابع شناسه انبار را بازمی‌گرداند.

```
public int getCapacity();
```
این تابع ظرفیت کلی انبار را باز می‌گرداند.

```
public int getFreeCapacity();
```
این تابع ظرفیت خالی انبار را باز می‌گرداند.

```
public HashMap<Good,Integer> getGoods();
```
یک hashMap باز می‌گرداند که key آن کالاهای موجود در انبار هستند و value آنها برابر با تعداد هر کالا است.

```
public void addGood(Good g, int amount);
```
تعداد کالای g را در انبار به اندازه مقدار amount افزایش می‌دهد.

```
public void removeGood(Good g, int amount);
```
به تعداد amount کالای g از انبار خارج می‌کند.

متدهای کلاس Customer:

```
public Customer(String name, int ID);
```
سازنده: نام مشتری و شناسه آنرا می‌گیرد.

```
public String getName();
```
نام مشتری را باز می‌گرداند.

```
public int getID();
```

شناسه مشتری را باز می‌گرداند.

```
public int getBalance();
```

مقدار موجودی مشتری را باز می‌گرداند.

```
public void setBalance(int amount);
```

مقدار موجودی مشتری را ست می‌گیرد.

```
public void addOrder(Order order);
```

یک درخواست خرید به لیست درخواست‌های خرید مشتری اضافه می‌کند.

```
public Order[] getTotalOrders();
```

تمام درخواست‌های خرید مشتری را باز می‌گرداند.

```
public Order[] getPendingOrders();
```

تمام درخواست‌هایی که کاربر آنها را ثبت نکرده را باز می‌گرداند.

```
public Order[] getSubmittedOrders();
```

تمام درخواست‌هایی که کاربر آن‌ها را ثبت کرده را باز می‌گرداند.

```
public void submitOder(Order order);
```

یک درخواست خرید را ثبت می‌کند. در صورتی که موجودی او کمتر از قیمت درخواست خرید بود یا کالا به تعداد درخواستی او موجود نبود، درخواست کاربر ثبت نمی‌شود. هنگامی که کاربر درخواست خرید را ثبت کرد باید به اندازه مبلغ درخواست خرید از موجودی او کسر شود. سپس به تعداد کالاهای سفارش داده شده در درخواست از کالاهای انبار کسر می‌شود. در صورتی که کالا موجود در درخواست در چندین انبار بیشتر از تعداد درخواستی موجود بود، در انباری که شناسه آن کمتر است انتخاب می‌شود و به تعداد درخواست شده از موجودی انبار کاسته می‌شود. بعد از ثبت درخواست کاربر حق تغییر کالاهای لیست خود را ندارد.

متدهای کلاس Order:

```
public Order(int ID, Customer c);
```

سازنده: شناسه درخواست خرید و مشتری درخواست دهنده را دریافت می‌کند.

```
public int getID()
```

شناسه خرید را باز می‌گرداند.

```
public String getStatus();
```

وضعیت درخواست را باز می‌گرداند. وضعیت درخواست در ابتدا باید pending باشد. در صورتی که کاربر آنرا ثبت کرد باید وضعیت آن به submitted تغییر کند.

```
public void setStatus (String status);
```

وضعیت درخواست را به مقدار ورودی تغییر می‌دهد.

```
public void addItem (Good good, int amount);
```
کالای good را به تعداد amount به لیست خرید اضافه می‌کند.

```
public void removeItem (Good good);
```
کالای good را از لیست خرید حذف می‌کند.

```
public HashMap<Good, Integer> getItems();
```
خروجی این تابع یک HashMap است که key های آن کالاهای موجود در لیست خرید است و value های آن تعداد آن کالا در درخواست خرید.

```
public int calculatePrice();
```
هزینه درخواست خرید را باز می‌گرداند. این مبلغ برابر است به جمع کل قیمت کالاهای موجود ضرب در تعداد هر یک از کالاها در دخواست خرید. درصورتی که تخفیفی برای درخواست درنظر گرفته شده بود، باید آنرا نیز اعمال کند.

```
public void addDiscount(Discount discount);
```
این تابع تخفیف را به خرید اعمال می‌کند. اعمال این تابع بدین صورت است که هزینه درخواست به اندازه درصد discount از آن کسر می‌شود.

نکته: هر کابر فقط می‌تواند یکبار از یک تخفیف استفاده کند.

متدهای کلاس Discount:

```
public Discount(int ID, int percent);
```
ورودی این سازنده شناسه تخفیف و درصد تخفیف است.

```
public void setOrder (Order order);
```
تخفیف را به order نسبت می‌دهد.

```
public Order getOrder();
```
order مربوط به این تخفیف را باز می‌گرداند. order در ابتدا برابر null است چرا که تخفیف همان ابتدا به order نسبت داده نمی‌شود.

```
public int getPercentage();
```
درصد تخفیف را باز می‌گرداند.

نحوه‌ی ارتباط این سامانه با کاربرانش با دستورات ورودی در terminal است و برنامه شما باید در یک حلقه بی‌نهایت ورودی بخواند. قالب دستورات این سامانه بصورت سلسله مراتبی است. بدین صورت که در ابتدا نوع دستور مشخص(add, remove, report و...) می‌شود. سپس موجودیتی که قرار است

این دستور روی آن اعمال شود (good, customer یا repository و...). در نهایت نیز اطلاعات موجودیت وارد می‌شود.

برنامه با دستور terminate خاتمه می‌یابد.

دستور add:

در صورتی که این دستور وارد شد در خط بعدی کاربر موجودیت مورد نیاز خود را وارد می‌کند.

customer:

درصورتی که موجودیت مشتری بود در خط های بعدی کاربر مشخصات مشتری را بدین صورت وارد می‌کند.

شناسه مشتری

نام مشتری

good:

درصورتی که موجودیت کالا بود در خط های بعدی کاربر مشخصات کالا را بدین صورت وارد می‌کند.

شناسه کالا

نام کالا

قیمت کالا

تعداد کالایی که قرار است اضافه شود

repository:

درصورتی که موجودیت انبار بود در خط های بعدی کاربر مشخصات انبار را بدین صورت وارد می‌کند.

شناسه انبار

ظرفیت انبار

order:

شناسه درخواست خرید

شناسه کاربر درخواست دهنده

balance:

این دستور برای اضافه کردن موجودی مشتری استفاده می‌شود. در خط‌های بعدی مشخصات balance بصورت زیر وارد می‌شود.

شناسه مشتری

مبلغی که باید به موجودی مشتری افزوده شود

item:

این دستور برای اضافه کردن کالا به درخواست خرید مشتری است. در خط‌های بعدی مشخصات item بصورت زیر وارد می‌شود.

شناسه درخواست خرید

شناسه کالا

تعداد کالا

:discount

درصورتی که موجودیت تخفیف بود در خط های بعدی کاربر مشخصات تخفیف را بدین صورت وارد می‌کند.

شناسه تخفیف

درصد تخفیف

دستور report:

:customers

در این حالت باید اطلاعات مشتریان فروشگاه هر کدام در یک خط پرینت شود. فرمت خروجی هر مشتری بدین صورت است.(از آنجا که خروجی شما به زبان انگلیسی هست فرمت‌های خروجی نیز به زبان انگلیسی هست).

customer-ID, customer-name, customer-balance, total-order-size, submitted-size

:repositories

در این حالت باید اطلاعات هر انبار در یک خط با فرمت زیر چاپ شود.

repository-ID, repository-capacity, repository-free-capacity

:income

این دستور در خط بعدی درآمد فروشگاه را چاپ می‌کند.

دستور remove:

:item

این دستور یک کالا را از لیست خرید مشتری حذف می‌کند. ورودی خط‌های بعدی برای توصیف این دستور بدین صورت است.

شناسه درخواست

شناسه کالا

دستور submit:

:order

این دستور یک درخواست خرید را ثبت می‌کند. ورودی خط بعدی برای توصیف این دستور بدین صورت است.

شناسه خرید

:discount

این دستور برای اعمال تخفیف به خریدهاست. ورودی خط‌های بعدی برای توصیف این دستور بدین صورت است.
شناسه درخواست
شناسه تخفیف
نکته: در صورتی که کاربر از این دستور استفاده کرد تخفیف از بین می‌رود و دیگر روی درخواست‌های دیگر بی تاثیر است.

نکته نهایی: از آنجایی که حجم این تمرین نسبتا زیاد است، پیشنهاد می‌شود که تمرین را خرد خرد بزنید و سابمیت کنید. تست کیس‌ها بگونه ای خواهد بود که پله پله کد شما را مورد بررسی قرار می‌دهد و اگر تمرین رو بصورت کامل پیاده نکرده باشید هم باز نمره‌ای خواهید گرفت.