دانشکده ریاضی و علوم کامپیوتر

برنامه‌سازی پیشرفته و کارگاه

تمرین ارث‌بری

استاد درس

دکتر مهدی قطعی

استاد دوم

بهنام یوسفی مهر

نگارش

شهاب الدین گریوانی

بهار ۱۴۰۳

فهرست

مقدمه

این هفته ارث‌بری و مباحث مربوط به اون رو یاد گرفتید. توی این تمرین قراره از این مباحث استفاده کنید و یک اپلیکیشن تجاری ساده رو شبیه‌سازی کنید که توی اون مشتری‌های مختلف می‌تونن با روش های مختلف پرداخت هاشون رو انجام بدن. در ادامه همه‌ی چیزهایی که شما باید پیاده‌سازی کنید آورده شده. توجه کنید که همه چیز به جز کلاس Main رو توی یک پکیج به اسم payment قرار بدید.

بخش اول: روش های پرداخت

اینترفیس PaymentStrategy

این اینترفیس قراره روش‌های مختلف پرداخت رو به صورت abstract نشون بده و دو تا متد داره:

void pay(double amount): متد پرداخت.

String getPaymentDetails(): این متد، باید اطلاعات مربوط به هر روش پرداخت رو به شکل یک String برگردونه.

کلاس هایی که PaymentStrategy رو implement می‌کنند

این کلاس‌ها روش‌های مختلف پرداخت رو نشون میدن و همگی باید PaymentStrategy رو implement کنند و متد های pay و getPaymentDetails رو پیاده‌سازی کنند.

کلاس CreditCardPayment

این کلاس پرداخت با کارت اعتباری رو نشون میده و باید دو تا فیلد داشته باشه:

String cardNumber: شماره کارت.

String cardHolderName: نام صاحب کارت.

کلاس PayPalPayment

این کلاس پرداخت با پی‌پال رو نشون میده و شامل یک فیلد میشه:

String email: ایمیل مشتری.

کلاس BitcoinPayment

این کلاس پرداخت با بیت‌کوین رو نشون میده و شامل یک فیلد میشه:

String walletAddress: آدرس والت بیت‌کوین مشتری.

توجه کنید که تمام این کلاس‌ها باید فیلدهاشون رو توی constructor ورودی بگیرن و ست کنن.

برای پیاده‌سازی متد pay در هرکدوم از این کلاس‌ها، کافیه مقداری که قراره پرداخت بشه همراه با اطلاعات مربوط به اون نوع پرداخت رو چاپ کنید. توجه کنید توی پیامی که چاپ می‌کنید حتماً باید

مقدار پرداخت شده و اطلاعات روش پرداخت مشخص باشه. اطلاعات روش پرداخت رو با متد getPaymentDetails بگیرید.

برای پیاده‌سازی getPaymentDetails هم صرفاً اسم روش پرداخت و اطلاعات مربوط به روش پرداخت (مقادیر فیلدهای اون کلاس) رو به شکل یک String برگردونید. یعنی مثلاً برای BitcoinPayment توی Stringی که برمی‌گردونید باید مقدار walletAddress مشخص باشه و همچنین مشخص باشه که پرداخت با بیت‌کوین انجام شده.

بخش دوم: مشتریان

کلاس ابسترکت Customer

این کلاس قراره نشون‌دهنده‌ی مشتری‌ها باشه.

Fields

String name: نام مشتری.

ArrayList<String> paymentHistory: این لیست، تاریخچه پرداخت‌های مشتری رو نشون می‌ده.

Methods

public Customer(String name): کانستراکتور Customer که نام مشتری را مقداردهی می‌کند.
توی این کانستراکتور باید paymentHistory رو هم به یک ArrayList خالی مقداردهی کنید.

public abstract void displayCustomerInfo(): این متد برای چاپ کردن اطلاعات مشتریه و برای برای هر زیرکلاس Customer جدا پیاده‌سازی میشه.

public void makePayment(PaymentStrategy paymentStrategy, double amount): این متد برای پرداخت توسط مشتریه. توی این متد باید متد paymentStrategy.pay() رو صدا بزنید تا مشتری با توجه به روش پرداختی که متد ورودی گرفته پرداخت رو انجام بده. همچنین اطلاعات پرداخت رو به تاریخچه پرداخت‌ها اضافه کنید. اطلاعات پرداخت یک String شامل مقدار پرداخت شده و اطلاعات روش پرداخته. اطلاعات روش پرداخت رو با صدا زدن paymentStrategy.getPaymentDetails() بگیرید.

public void showPaymentHistory(): توی این متد با استفاده از یک حلقه تاریخچه پرداخت مشتری رو چاپ کنید.

زیرکلاس های Customer

دو کلاس **RegularCustomer** و **PremiumCustomer** بسازید. این دو کلاس به ترتیب مشتری‌های معمولی و مشتری‌های پرمیوم رو نمایش میدن. این دو کلاس باید از Customer ارث‌بری کنند. برای هرکدوم باید کانستراکتور مناسبی قرار بدید که اسم رو ورودی بگیره و با استفاده از super و صدا زدن

کانستراکتور Customer، name و paymentHistory رو مقداردهی کنه. همچنین هر دوی این کلاس ها باید متد displayCustomerInfo رو به شکل مناسبی پیاده‌سازی کنند. توی پیامی که displayCustomerInfo چاپ می‌کنه باید اسم مشتری و عادی یا پرمیوم بودن مشتری مشخص باشه.

بخش سوم: کلاس Main

توی متد main از کلاس Main، سه مشتری متفاوت ایجاد کنید. حداقل یکی از این مشتری‌ها عادی و حداقل یکیشون پرمیوم باشند. از هر کدوم از سه روش پرداخت یک آبجکت بسازید. اطلاعات هر مشتری رو با displayCustomerInfo چاپ کنید. بعد با استفاده از روش‌های پرداختی که ساختید برای هرکدوم از مشتری‌ها دو پرداخت انجام بدید و بعد تاریخچه پرداخت هر مشتری رو چاپ کنید.

UML

اینجا می‌تونید یک نمای کلی از تمام چیزهایی که باید پیاده‌سازی کنید و ارتباط بینشون رو ببینید: