

**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 3](#_Toc191937294)

[بخش اول: روش های پرداخت 4](#_Toc191937295)

[اینترفیس PaymentStrategy 4](#_Toc191937296)

[کلاس هایی که PaymentStrategy رو implement میکنند 4](#_Toc191937297)

[کلاس CreditCardPayment 4](#_Toc191937298)

[کلاس PayPalPayment 4](#_Toc191937299)

[کلاس BitcoinPayment 4](#_Toc191937300)

[بخش دوم: مشتریان 6](#_Toc191937301)

[کلاس ابسترکت Customer 6](#_Toc191937302)

[Fields 6](#_Toc191937303)

[Methods 6](#_Toc191937304)

[زیرکلاس های Customer 6](#_Toc191937305)

[بخش سوم: کلاس Main 8](#_Toc191937306)

[UML 9](#_Toc191937307)

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.

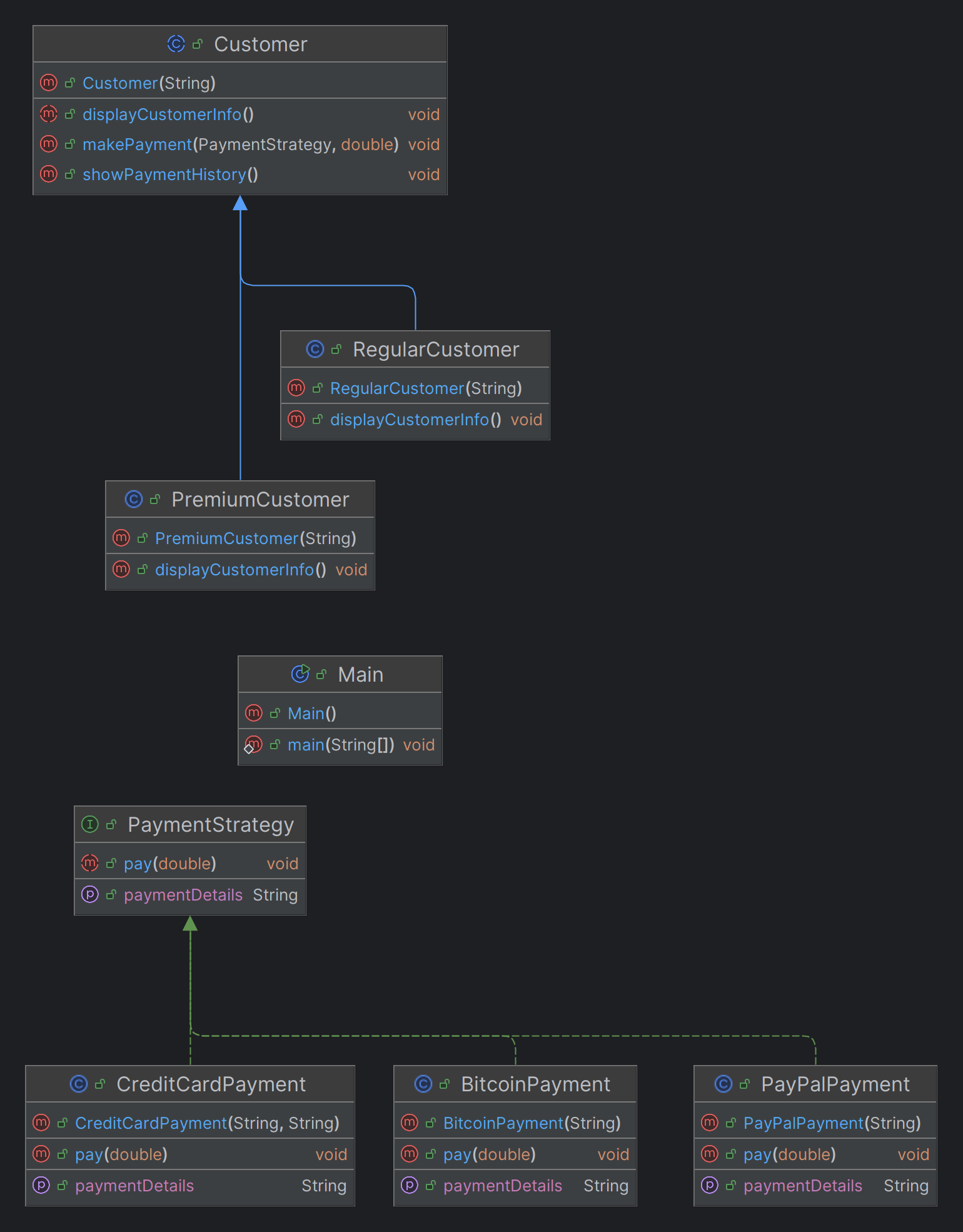
زیرکلاس های Customer

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: