



دانشگاه علم و صنعت ایران

دانشکده مهندسی کامپیوتر

برنامه‌سازی پیشرفته
تمرین های سری دوم

مدرس: سید صالح اعتمادی
طرح تمرین: امید میرزاجانی

مهلت ارسال:
دوشنبه ۱۹ اسفند ۹۸

فهرست مطالب

۲	۱	مقدمه
۲	۱.۱	موارد مورد توجه
۲	۲	آماده سازی های اولیه
۲	۱.۲	ساخت پروژه ی C#
۲	۲.۲	قواعد نام گذاری
۳	۳	APKALA
۳	۱.۳	کلاس Product
۳	۲.۳	کلاس Category
۳	۳.۳	کلاس Cart
۳	۴.۳	کلاس Store
۴	۴	Telegram
۴	۱.۴	کلاس Person
۴	۲.۴	کلاس Message
۴	۳.۴	ارتباط دو کلاس

۱ مقدمه

۱.۱ موارد مورد توجه

- توجه داشته باشید که برای کسب نمره‌ی قبولی درس کسب حداقل نصف نمره‌ی هر سری تمرین الزامی می‌باشد.
- مهلت ارسال پاسخ تمرین تا ساعت ۲۳:۵۹ روز اعلام شده است. توصیه می‌شود نوشتن تمرین را به روزهای نهایی موکول نکنید.
- همکاری و هم‌فکری شما در حل تمرین مانعی ندارد، اما پاسخ ارسالی هر کس حتماً باید توسط خود او نوشته شده باشد.
- مبنای درس، اعتماد بر پاسخ ارسالی از سوی شماست؛ بنابراین ارسال پاسخ در ریپازیتوری گیت شما به این معناست که پاسخ آن تمرین، توسط شما نوشته شده است. در صورت تقلب یا اثبات عدم نوشتار پاسخ حتی یک سوال از تمرین، برای هر دو طرف تقلب‌گیرنده و تقلب‌دهنده نمره‌ی مردود برای درس در نظر گرفته خواهد شد.
- توجه داشته باشید که پاسخ‌ها و کدهای مربوط به هر مرحله را بایستی تا قبل از پایان زمان مربوط به آن مرحله، در سایت [Azure DevOps](#) (طبق توضیحات کارگاه‌ها و کلاس‌ها) بفرستید. درست کردن `Pull request` و `Complete` کردن `Pull request` و انتقال به شاخه‌ی `master` پس از تکمیل تمرین فراموش نشود!
- پس از پایان مهلت ارسال تا ۲ روز به ازای هر روز تاخیر ۱۰ درصد از نمره مربوط به تمرین کسر خواهد شد و پس از ۲ روز نمره‌ای به تمرین تعلق نخواهد گرفت.
- بعضی از قسمت‌های تمرین نیاز به پیاده‌سازی بر روی هر چهار زبان `C#` ، `Python` ، `C++` و `Java` را دارند بعضی هم خیر. بنابراین روبروی هر سوال زبان‌های مورد نیاز برای پیاده‌سازی مشخص شده است.

۲ آماده‌سازی‌های اولیه

۱.۲ ساخت پروژه‌ی C#

برای ایجاد پروژه C# کافی است کد زیر را در ترمینال خود اجرا کنید:

```
۱ mkdir A2_cs
۲ cd A2_cs
۳ dotnet new sln
۴ mkdir A2_cs
۵ cd A2_cs
۶ dotnet new console
۷ cd ..
۸ dotnet sln add A2_cs\A2_cs.csproj
۹ mkdir A2_cs.Tests
۱۰ cd A2_cs.Tests
۱۱ dotnet new mstest
۱۲ dotnet add reference ..\A2_cs\A2_cs.csproj
۱۳ cd ..
۱۴ dotnet sln add A2_cs.Tests\A2_cs.Tests.csproj
```

۲.۲ قواعد نام‌گذاری

قواعد نام‌گذاری تمرین را از جدول ۱ مطالعه کنید.

جدول ۱: قراردادهای نام‌گذاری تمرین

Naming conventions		
Branch	Directory	Pull Request
fb_A2	A2	A2

* در کل یک دیرکتوری در شاخه `Assignments` به نام `A2` بسازید و داخل آن، دو دیرکتوری به نام‌های `A2_cs` ، `A2_python` داشته باشید و فایل‌های مربوط به هر زبان را داخل دیرکتوری مربوطه بگذارید.

۱.۳ کلاس Product

در اینجا کمی تبدیل فضای مسئله به فضای راه حل با استفاده از زبان شی‌گرای سی‌شارپ را تمرین می‌کنیم: کلاس محصول دارای ویژگی‌های مولفه مشخصه (`Id`) ، نام (`Name`) ، قیمت (`Price`) و نمره از لحاظ محبوبیت (`Rate`) است که یک محصول را از دیگری تمایز میدهد. ممکن است موجودیتی که در نظر می‌گیریم ویژگی‌های دیگری هم داشته باشد اما ما فقط ویژگی‌هایی را در نظر می‌گیریم که به حل مسئله کمک می‌کند.

گام اول: برای هر `property` ای که در کلاس وجود دارد `getter` و `setter` مناسب بنویسید. **گام دوم:** شما باید سازنده (`constructor`) این کلاس را تکمیل کنید تا شی‌ای که از کلاس ساخته می‌شود معتبر باشد یعنی هر `Product` ای که ساخته می‌شود لزوماً دارای ویژگی‌های ذکر شده باشد. پس از پیاده‌سازی صحیح سازنده‌ی این کلاس تست `ProductConstructorTest` پاس خواهد شد. بعد از پیاده‌سازی این کلاس و پاس شدن تست‌های آن کار شما با این کلاس و فایل آن تمام شده است. دیگر نیازی به تغییر این کد نخواهید داشت.

۲.۳ کلاس Category

کلاس دسته بندی دارای ویژگی‌های مولفه مشخصه (`Id`) و محصولات (`Products`) است که یک دسته را از دیگری تمایز میدهد. برای مثال تمام گوشی‌های موبایل در یک دسته بندی قرار گیرند.

گام اول: برای هر `property` ای که در کلاس وجود دارد `getter` و `setter` مناسب بنویسید. **گام دوم:** شما باید سازنده (`constructor`) این کلاس را تکمیل کنید تا شی‌ای که از کلاس ساخته می‌شود معتبر باشد یعنی هر `Category` ای که ساخته می‌شود لزوماً دارای ویژگی‌های ذکر شده باشد.

پس از پیاده‌سازی صحیح سازنده‌ی این کلاس تست `CategoryConstructorTest` پاس خواهد شد. **گام سوم:** شما باید متد `AddProduct` را بگونه‌ای پیاده سازی کنید که ورودی آن یک محصول باشد و با فراخوانی متد آن محصول را به محصولات خود اضافه کند. پس از پیاده‌سازی صحیح تست `CategoryAddProductTest` پاس خواهد شد. **گام چهارم:** شما باید متد `FilterByPrice` را بگونه‌ای پیاده سازی کنید که دو ورودی از جنس `int` بگیرد و تمامی محصولات آن دسته بندی، که قیمتشان بین دو عدد ورودی است را به عنوان خروجی بازگرداند. پس از پیاده‌سازی صحیح تست `CategoryFilterByPriceTest` پاس خواهد شد.

۳.۳ کلاس Cart

کلاس سبد خرید دارای ویژگی‌های نام صاحب سبد (`Owner`) و محصولات (`Products`) است که یک دسته را از دیگری تمایز میدهد. **گام اول:** برای هر `property` ای که در کلاس وجود دارد `getter` و `setter` مناسب بنویسید.

گام دوم: شما باید سازنده (`constructor`) این کلاس را تکمیل کنید تا شی‌ای که از کلاس ساخته می‌شود معتبر باشد یعنی هر `Cart` ای که ساخته می‌شود لزوماً دارای ویژگی‌های ذکر شده باشد.

پس از پیاده‌سازی صحیح سازنده‌ی این کلاس تست `CartConstructorTest` پاس خواهد شد. **گام سوم:** شما باید متد `AddProduct` را بگونه‌ای پیاده سازی کنید که ورودی آن یک محصول باشد و با فراخوانی متد آن محصول را به محصولات خود اضافه کند.

پس از پیاده‌سازی صحیح تست `CartAddProductTest` پاس خواهد شد. **گام چهارم:** شما باید متد `CalculatePrice` را بگونه‌ای پیاده سازی کنید که با فراخوانی آن قیمت کل آن سبد خرید را بازگرداند. پس از پیاده‌سازی صحیح تست `CartCalculatePriceTest` پاس خواهد شد.

۴.۳ کلاس Store

کلاس فروشگاه دارای ویژگی‌های نام (`Name`) ، دسته بندی‌ها (`Categories`) و سبد های خرید (`Carts`) است که یک فروشگاه را از دیگری تمایز میدهد.

گام اول: برای هر `property` ای که در کلاس وجود دارد `getter` و `setter` مناسب بنویسید. **گام دوم:** شما باید سازنده (`constructor`) این کلاس را تکمیل کنید تا شی‌ای که از کلاس ساخته می‌شود معتبر باشد یعنی هر `Store` ای که ساخته می‌شود لزوماً دارای ویژگی‌های ذکر شده باشد.

پس از پیاده‌سازی صحیح سازنده‌ی این کلاس تست `StoreConstructorTest` پاس خواهد شد.

گام سوم: شما باید متد `AddCart` را بگونه‌ای پیاده‌سازی کنید که ورودی آن یک سبد خرید باشد و با فراخوانی متد آن سبد خرید را به لیست سبدهای خود اضافه کند.

پس از پیاده‌سازی صحیح تست `StoreAddCartTest` پاس خواهد شد.

گام چهارم: شما باید متد `AddCategory` را بگونه‌ای پیاده‌سازی کنید که ورودی آن یک دسته بندی باشد و با فراخوانی متد، آن دسته بندی را به لیست سبدهای خود اضافه کند.

پس از پیاده‌سازی صحیح تست `StoreAddCategoryTest` پاس خواهد شد.

گام پنجم: شما باید متد `Bestselling` را به گونه‌ای پیاده‌سازی کنید که با فراخوانی آن، محصولی که از همه بیشتر به فروش رفته را بازگرداند.

پس از پیاده‌سازی صحیح تست `StoreBestsellingTest` پاس خواهد شد.

گام ششم: شما باید متد `MostPopular` را به گونه‌ای پیاده‌سازی کنید که با فراخوانی آن، محصولی که از همه محبوب تر است، را بازگرداند.

پس از پیاده‌سازی صحیح تست `StoreMostPopularTest` پاس خواهد شد.

۱.۴ کلاس Person

این کلاس دارای ویژگی های مولفه مشخصه (ID) ، نام (Name) ، مخاطبین (Contacts) و چت (Chats) است. لازم به ذکر است که هر عضو، مولفه مشخصه ی منحصر به فردی دارد که او را با بقیه اعضا تمایز میدهد.

گام اول: شما باید سازنده (constructor) این کلاس را تکمیل کنید تا شی ای که از کلاس ساخته می شود معتبر باشد یعنی هر Person ای که ساخته می شود لزوما دارای ویژگی های ذکر شده باشد.

پس از پیاده سازی صحیح سازنده ی این کلاس تست `test_PersonConstructor` پاس خواهد شد.

گام دوم: متد `AddContact` را به گونه ای پیاده سازی کنید که یک عضو (Person) بگیرد، و اگر با آن عضو قبلاً مخاطب نبوده، آن عضو را به مخاطبین خود اضافه کند.

پس از پیاده سازی صحیح تست `test_PersonAddContact` پاس خواهد شد.

۲.۴ کلاس Message

این کلاس دارای ویژگی های مبدا (Source) ، مقصد (Destination) و متن پیام (Context) است.

گام اول: شما باید سازنده (constructor) این کلاس را تکمیل کنید تا شی ای که از کلاس ساخته می شود معتبر باشد یعنی هر Message ای که ساخته می شود لزوما دارای ویژگی های ذکر شده باشد.

پس از پیاده سازی صحیح سازنده ی این کلاس تست `test_MessageConstructor` پاس خواهد شد.

۳.۴ ارتباط دو کلاس

شما باید متد `SendMessage` را برای کلاس Person را بگونه ای پیاده سازی کنید که با گرفتن یک پیام (Message) ، به چت هر دو نفر، آن پیام را اضافه کند. همچنین اگر دو نفر با یکدیگر مخاطب نبودند ، ابتدا یکدیگر را به مخاطبین خود اضافه کنند.

پس از پیاده سازی صحیح تست `test_SendMessage` پاس خواهد شد.