



دانشگاه علم و صنعت ایران

دانشکده مهندسی کامپیوتر

برنامه‌سازی پیشرفته
تمرین‌های سری دهم

مدرس: سید صالح اعتمادی
طرح تمرین: امید میرزاجانی

مهلت ارسال:
شنبه ۱۰ خرداد ۹۹

فهرست مطالب

۲	۱	مقدمه
۲	۱.۱	موارد مورد توجه
۲	۲	آماده‌سازی‌های اولیه
۲	۱.۲	ساخت پروژه ی C#
۲	۲.۲	قواعد نام‌گذاری
۳	۳	کتابخانه مجازی
۳	۱.۳	Person
۳	۱.۱.۳	Constructor
۳	۲.۱.۳	Print
۳	۲.۳	Book
۳	۳.۳	Author
۳	۱.۳.۳	Constructor
۳	۲.۳.۳	Print
۳	۴.۳	Notifiable
۳	۵.۳	Email
۳	۶.۳	Mobile
۳	۷.۳	Reader
۳	۱.۷.۳	Print()

۴	borrow_book()	۲.۷.۳
۴	return_book()	۳.۷.۳
۴	show_book()	۴.۷.۳
۵	Management	۸.۳
۵	Reload	۱.۸.۳

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 اعلام‌شده 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 حتما باید توسط خود او نوشته شده باشد.
- مبنای درس، اعتماد بر پاسخ ارسالی از سوی شماست؛ بنابراین ارسال پاسخ در رپایزتوری 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 A10_cs
۲ cd A10_cs
۳ dotnet new sln
۴ mkdir A10_cs
۵ cd A10_cs
۶ dotnet new console
۷ cd ..
۸ dotnet sln add A10_cs\A10_cs.csproj
۹ mkdir A10_cs.Tests
۱۰ cd A10_cs.Tests
۱۱ dotnet new mstest
۱۲ dotnet add reference ..\A10_cs\A10_cs.csproj
۱۳ cd ..
۱۴ dotnet sln add A10_cs.Tests\A10_cs.Tests.csproj

```

۲.۲ قواعد نام‌گذاری

قواعد نام‌گذاری تمرین را از جدول ۱ مطالعه کنید.

* در کل یک دیرکتوری داخل Assignments به نام A10 بسازید و داخل آن، یک دیرکتوری به نام A10_cs داشته باشید و فایل‌های مربوطه را داخل دیرکتوری مربوطه بگذارید.

جدول ۱: قراردادهای نام‌گذاری تمرین

Naming conventions		
Branch	Directory	Pull Request
fb_A10	A10	A10

۳ کتابخانه مجازی

هدف از این تمرین کمک به کتابخانه‌ها و بهبود مدیریت آنها است. هم چنین پیشنهاد میشود که در کنار حل تمرین، تست‌ها را نیاز نگاه کنید. همانطور که میدانید اشیا و افرادی که با کتابخانه در ارتباط هستند، کتاب‌ها، اعضا و نویسندگان هستند که از شما میخواهیم ما را در پیاده‌سازی قسمت‌های مختلف کمک کنید.

۱.۳ Person

۱.۱.۳ Constructor

این کلاس برای پیاده‌سازی افراد مختلف است و کاربردهای بیشتری آن، برای حل مساله را جلوتر خواهیم دید. برای این کلاس دو ویژگی

- name از نوع `string`

- gender از نوع `Gender`

را قرار دهید.

توجه: برای این کلاس `getter`، `setter` مناسب بنویسید؛ به طوری که با توجه به مقدار `gender`، به ابتدای اسم آن شخص، `Mr.` یا `Ms.` اضافه کند. پس از پیاده‌سازی صحیح، تست `PersonConstructor_Test` پاس خواهد شد.

۲.۱.۳ Print

برای این کلاس، متد `Print` را به گونه‌ای پیاده‌سازی کنید که نام آن شخص را به عنوان خروجی بازگرداند. پس از پیاده‌سازی صحیح، تست `PersonPrint_Test` پاس خواهد شد.

۲.۳ Book

این کلاس برای پیاده‌سازی کتاب‌های کتابخانه به کار میرود. هر کتاب شامل ویژگی‌های

- `book_id` از نوع `long`

- `author` از نوع `string`

- `name` از نوع `string`

- `status` از نوع `Status`

هست. پس از پیاده‌سازی صحیح، تست `BookConstructor_Test` پاس خواهد شد.

۳.۳ Author

۱.۳.۳ Constructor

این کلاس را به گونه‌ای پیاده‌سازی کنید که اولاً از کلاس `Person` ارث بری کند. سپس یک ویژگی به نام `books` از نوع لیستی از کتاب‌ها، اضافه کنید. هم چنین متد `new_book` را به گونه‌ای پیاده‌سازی کنید که کتاب ورودی را، به لیست کتاب‌ها اضافه کند. این کلاس را به همراه سازنده‌اش پیاده‌سازی کنید تا تست `AuthorConstructor_Test` پاس شود.

۲.۳.۳ Print

متد `Print()` که از کلاس `Person` به ارث رسیده، را به گونه‌ای تغییر دهید تا علاوه بر نام، تعداد کتاب‌های آن نویسنده را نیز به عنوان خروجی بازگرداند. پس از پیاده‌سازی صحیح، تست `AuthorPrint_Test` پاس خواهد شد.

Notifiable ۴.۳

برای ارتباط با اعضای کتابخانه نیاز به یک دستگاهی مثل موبایل یا ایمیل نیاز است، تا بتوانیم آنان را از بعضی اخبار مطلع کنیم. برای این کلاس یک ویژگی به نام Messages از نوع لیستی از رشته، در نظر بگیرید. همچنین برای این کلاس متدی به نام Notif ایجاد کنید که یک رشته به عنوان ورودی بگیرد و با توجه به خواسته مساله خروجی مناسب دهد. پس از پیاده سازی صحیح موارد بالا، تست `NotifiableConstructor_Test` پاس خواهد شد.

Email ۵.۳

همانطور که گفتیم، یکی از راه های ارتباطی ایمیل است. این کلاس را به گونه ای پیاده سازی کنید که اولاً از کلاس Notifiable ارث بری کند؛ دوماً متدی به نام Notif را به گونه ای تغییر دهید که ابتدای آن عبارت `"Sent email"` بیاید. پس از پیاده سازی صحیح موارد بالا، تست `EmailConstructor_Test` پاس خواهد شد.

Mobile ۶.۳

این کلاس مشابه کلاس Email است، با این تفاوت که در خروجی متدی `Notif`، عبارت `"Sent short message"` خواهد آمد. پس از پیاده سازی صحیح موارد بالا، تست `MobileConstructor_Test` پاس خواهد شد.

Reader ۷.۳

این کلاس برای پیاده سازی اعضای کتابخانه ایجاد شده و همانطور که انتظار میرود، از کلاس Person ارث بری میکند. علاوه بر آن، شامل ویژگی های

- `enterTime`: همان زمانی است که آن فرد، در کتابخانه ثبت نام کرده است.

- `borrowed_Books`: شامل Tuple هایی است که در آن کتاب امانت گرفته شده و زمان امانت ذخیره خواهد شد

- `notifiables`: شامل همه دستگاه هایی است که کتابخانه میتواند با آن عضو در ارتباط باشد.

پس از پیاده سازی صحیح سازنده این کلاس، تست `ReaderConstructor_Test` پاس خواهد شد.

Print() ۱.۷.۳

این متدی را بگونه ای پیاده سازی کنید که متدی به ارث رسیده ی `Print` را تغییر دهد و علاوه بر نام، تاریخ ورود آن عضو را نیز نشان دهد. از پیاده سازی صحیح موارد بالا، تست `ReaderBorrowBook_Test` پاس خواهد شد.

borrow_book() ۲.۷.۳

این متدی را بگونه ای پیاده سازی کنید که دو ورودی کتاب و تاریخ بگیرد، و اگر این عضو، شرایط امانت را داشت، به کتاب های امانت گرفته شده اش اضافه کند. شرط امانت این است که اولاً آن کتاب توسط کسی، از قبل امانت گرفته نشده باشد. و هم چنین هیچ شخصی نمیتواند بیش از ۲ کتاب را امانت بگیرد. پس از پیاده سازی صحیح موارد بالا، تست `ReaderBorrowBook_Test` پاس خواهد شد.

return_book() ۳.۷.۳

هر کتاب مجاز به امانت گرفته شدن، به مدت ۲ هفته است و به ازای هر روزی که بیشتر از آن امانت گرفته شود، باید به عنوان تاخیر ۱۰۰۰ تومان به مسئول کتابخانه داده شود. این متدی، قرار است کار آقای مسئول کتابخانه را راحت کند.

ورودی های این متدی شامل شناسه^۱ کتاب، و تاریخ است. حال شما باید یک `Task<long>` به عنوان خروجی بازگردانید، که با اجرای آن `Task`، اولاً آن کتاب از لیست کتاب های آن عضو حذف شود، دوماً خروجی آن `Task` مقدار هزینه ای است که فرد باید خسارت دهد. پس از پیاده سازی صحیح موارد بالا، تست `ReaderReturnBook_Test` پاس خواهد شد.

show_book() ۴.۷.۳

این متدی را به گونه ای پیاده سازی کنید، که نام کتابهای امانت گرفته شده را بازگرداند تا تست `ReaderShowBook_Test` پاس شود.

ID^۱

۸.۳ Management

این کلاس برای مدیریت کتابخانه کارا است. برای آن دو متد به نام های `add_book` و `add_member` ایجاد کنید. متد `add_book` یک کتاب به عنوان ورودی میگیرد و آن کتاب را به مجموعه کتابهای کتابخانه اضافه میکند. و نیز متد `add_member` یک Reader به عنوان ورودی میگیرد و به نوعی آن را ثبت نام میکند. پس از پیاده سازی صحیح موارد بالا، تست `ManagementConstructor_Test` پاس خواهد شد.

۱.۸.۳ Reload

هر عضو کتابخانه، حداکثر یک سال عضو است و بعد از آن باید اکانت خود را تمدید کند. از شما میخواهیم که به مسئول کتابخانه کمک کنید تا با همه اعضای که بیش از ۱ سال از زمان ثبت نامشان گذشته، تماس بگیرد و آن ها را از این اتفاق **به طور همزمان** مطلع سازد. پیام باید به همه دستگاه های آن عضو ارسال شود و متن آن باید شامل عبارت `"Your Account Closed."` باشد.

توجه: برای گرفتن نمره این بخش، باید از `Linq` استفاده کنید. پس از پیاده سازی صحیح موارد بالا، تست `ManagementReload_Test` پاس خواهد شد.

پراز خبرای خوب باشید!