



برنامه‌سازی پیشرفته  
تمرین های سری پنجم

مدرس: سید صالح اعتمادی  
طرح تمرین: امید میرزاجانی

مهلت ارسال:  
شنبه ۳۰ فروردین ۹۹

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 نسل جدید C#
	۳subsection.۱.۳
۳	IPerson ۱.۱.۳
۳	IDoctor ۲.۱.۳
	۳subsection.۲.۳
۳	Patient ۱.۲.۳
۳	GeneralPractitioner ۲.۲.۳
۴	Dentist ۳.۲.۳
۴	Surgeon ۴.۲.۳
۵	۴ مجتمع

## 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 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 قسمت های تمرین نیاز به پیاده سازی بر روی هر چهار زبان `C#` ، `Python` ، `C++` و `Java` را دارند بعضی هم خیر. بنابراین روبروی هر سوال زبان های مورد نیاز برای پیاده سازی مشخص شده است.

## ۲ آماده سازی های اولیه

### ۱.۲ ساخت پروژه ی C#

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

```
1 mkdir A5_cs
2 cd A5_cs
3 dotnet new sln
4 mkdir A5_cs
5 cd A5_cs
6 dotnet new console
7 cd ..
8 dotnet sln add A5_cs\A5_cs.csproj
9 mkdir A5_cs.Tests
10 cd A5_cs.Tests
11 dotnet new mstest
12 dotnet add reference ..\A5_cs\A5_cs.csproj
13 cd ..
14 dotnet sln add A5_cs.Tests\A5_cs.Tests.csproj
```

### ۲.۲ قواعد نام گذاری

قواعد نام‌گذاری تمرین را از جدول ۱ مطالعه کنید.

جدول ۱: قراردادهای نام‌گذاری تمرین

Naming conventions		
Branch	Directory	Pull Request
fb_A5	A5	A5

\* در کل یک دیرکتوری داخل Assignments به نام A5 بسازید و داخل آن، یک دیرکتوری به نام A5\_cs داشته باشید و فایل های مربوطه را داخل دیرکتوری مربوطه بگذارید.

## ۳ نسل جدید C#

دیمو<sup>۱</sup> که برنامه نویسی را به تازگی شروع کرده، قصد دارد در این روزهای قرنطینه از وقت خود استفاده کند. از آنجا که کارکنان بیمارستان درگیر مقابله با ویروس Covid-۱۹ هستند، برای جمع آوری اطلاعات بخش های مختلف، نیرو و زمان کافی ندارند. برای همین از او خواسته اند که اطلاعات خواسته شده رو از بخش های مختلف برایشان استخراج کند.

### ۱.۳ واسط ها<sup>۲</sup>

#### IPerson ۱.۱.۳

این واسط برای پیاده سازی افراد مختلف جامعه به کار می رود که شامل ویژگی های:

- `Firstname` (نام) از نوع `string`

- `Lastname` (نام خانوادگی) از نوع `string`

است. پس هر شخصی که وجود داشته باشد، الزاماً دارای ویژگی های ذکر شده هست.

#### IDoctor ۲.۱.۳

این واسط برای پیاده سازی پزشکان بیمارستان به کار می رود که شامل ویژگی های:

- `Field` (زمینه کاری) از نوع `string`

- `Salary` (حقوق ماهیانه) از نوع `long`

- `University` (دانشگاه) از نوع `string`

است. پس هر پزشکی که وجود داشته باشد، الزاماً دارای ویژگی های ذکر شده هست.

### ۲.۳ کلاس ها<sup>۳</sup>

#### Patient ۱.۲.۳

هر بیمار منطقاً یک شخص حقیقی است و دارای ویژگی های مربوط به IPerson است. هم چنین علاوه بر آنها دارای دو ویژگی

- `Disease` (زمینه بیماری)

- `Recovered` (بهبودی)

نیز هست. پس از پیاده سازی صحیح این کلاس به همراه سازنده اش، تست `PatientClassTests` پاس خواهد شد.

#### GeneralPractitioner ۲.۲.۳

هر پزشک عمومی منطقاً یک شخص حقیقی و یک پزشک است و دارای ویژگی های مربوط به واسط های گفته شده است. هم چنین علاوه بر آنها دارای ویژگی `Patients` (بیماران تحت نظر) نیز هست که از نوع لیستی از بیماران است.

متدها

- `Acceptable` : این متد را به گونه ای پیاده سازی کنید که یک بیمار به عنوان ورودی بگیرد و اگر در زمینه بیماری آن فرد، کلید واژه هایی مانند `Cough` ، `Sneezing` و `throat Sore` بود؛ آن بیمار را به بیماران تحت نظر خود اضافه کند.

پس از پیاده سازی صحیح این متد تست `AddingPatientsToGeneralPractitioner` پاس خواهد شد.

---

Dimo<sup>۱</sup>  
Interfaces<sup>۲</sup>  
Classes<sup>۳</sup>

- **GraduatedFrom** : این متد را بگونه ای پیاده سازی کنید که یک خروجی از نوع **string** داشته باشد که؛  
با نام و نام خانوادگی پزشک شروع شود، سپس رشته **is graduated from** باشد و در انتها نیز دانشگاهی که آن پزشک در آن تحصیل کرده بیاید.  
پس از پیاده سازی صحیح این متد، تست **GraduatingGenerals** پاس خواهد شد.
- **Work** : این متد را بگونه ای پیاده سازی کنید که یک خروجی از نوع **string** داشته باشد که؛  
با رشته **This General Practitioner works on** شروع شود و در انتها نیز زمینه کاری آن پزشک مشخص شود.  
پس از پیاده سازی صحیح این متد، تست **WorkingGenerals** پاس خواهد شد.

### ۳.۲.۳ Dentist

هر دندان پزشک منطقاً یک شخص حقیقی و یک پزشک است و دارای ویژگی های مربوط به واسط های گفته شده است.  
هم چنین علاوه بر آنها دارای ویژگی **Patients** (بیماران تحت نظر) نیز هست که از نوع لیستی از بیماران است.

متدها

- **Acceptable** : این متد را به گونه ای پیاده سازی کنید که یک بیمار به عنوان ورودی بگیرد و اگر در زمینه بیماری آن فرد، کلید واژه هایی مانند **Toothache** ، **Teeth** و **Dental** بود؛ آن بیمار را به بیماران تحت نظر خود اضافه کند.  
پس از پیاده سازی صحیح این متد تست **AddingPatientsToDentist** پاس خواهد شد.
- **GraduatedFrom** : این متد را بگونه ای پیاده سازی کنید که یک خروجی از نوع **string** داشته باشد که؛  
با نام و نام خانوادگی پزشک شروع شود، سپس رشته **is graduated from** باشد و در انتها نیز دانشگاهی که آن پزشک در آن تحصیل کرده بیاید.  
پس از پیاده سازی صحیح این متد، تست **GraduatingDentists** پاس خواهد شد.
- **Work** : این متد را بگونه ای پیاده سازی کنید که یک خروجی از نوع **string** داشته باشد که؛  
با رشته **This Dentist works on** شروع شود و در انتها نیز زمینه کاری آن پزشک مشخص شود.  
پس از پیاده سازی صحیح این متد، تست **WorkingDentists** پاس خواهد شد.

### ۴.۲.۳ Surgeon

هر جراح منطقاً یک شخص حقیقی و یک پزشک است و دارای ویژگی های مربوط به واسط های گفته شده است.  
هم چنین علاوه بر آنها دارای ویژگی **Patients** (بیماران تحت نظر) نیز هست که از نوع لیستی از بیماران است.

متدها

- **Acceptable** : این متد را به گونه ای پیاده سازی کنید که یک بیمار به عنوان ورودی بگیرد و اگر در زمینه بیماری آن فرد، کلید واژه هایی مانند **Cancer** ، **Appendix** و **Kidney** بود؛ آن بیمار را به بیماران تحت نظر خود اضافه کند.  
پس از پیاده سازی صحیح این متد تست **AddingPatientsToSurgeon** پاس خواهد شد.
- **GraduatedFrom** : این متد را بگونه ای پیاده سازی کنید که یک خروجی از نوع **string** داشته باشد که؛  
با نام و نام خانوادگی جراح شروع شود، سپس رشته **is graduated from** باشد و در انتها نیز دانشگاهی که آن پزشک در آن تحصیل کرده بیاید.  
پس از پیاده سازی صحیح این متد، تست **GraduatingSurgeons** پاس خواهد شد.
- **Work** : این متد را بگونه ای پیاده سازی کنید که یک خروجی از نوع **string** داشته باشد که؛  
با رشته **This Surgeon works on** شروع شود و در انتها نیز زمینه کاری آن پزشک مشخص شود.  
پس از پیاده سازی صحیح این متد، تست **WorkingSurgeons** پاس خواهد شد.

## ۴ مجتمع

حال که همه بخش های مختلف را پیاده سازی کردید، به مسئولین بیمارستان کمک کنید که اطلاعات پزشکان را به دست بیاورند. از شما میخواهیم که به دیمو کمک کند تا کلاس Doctors را برای اقشار مختلف پزشکان پیاده سازی کنید. این کلاس را به گونه ای پیاده سازی کنید که یک ویژگی به نام Doctor از نوع لیست داشته باشد، و بتواند پزشکان را در خود ذخیره کند. سپس برای آن دو متد زیر را پیاده سازی کنید:

- **ListOfRecoveredPatients** : این متد را به گونه ای پیاده سازی کنید که بین بیماران همه پزشکان که در ویژگی Doctor ذخیره شده اند، را برگرد و اسم کامل بیمارانی که سلامتی خود را به طور کامل به دست آورده اند را بازگرداند. پس از پیاده سازی صحیح این متد تست **ListOfRecoveredPatientsTests** پاس خواهد شد.
- **SortDoctors** : این متد را به گونه ای پیاده سازی کنید که پزشکان ذخیره شده در Doctors را مرتب کند. شیوه مرتب سازی هم به اینگونه است که ابتدا پزشکان بر حسب نسبت تعداد بیماران بهبود یافته به کل بیماران آن پزشک مرتب میشوند، و اگر دو پزشک با نسبت یکسان وجود داشت، آنها را بر حسب نام آن پزشک مرتب کند. به طور مثال اگر پزشک شماره ۱، پنج بیمار داشت دو بیمار بهبود یافته داشت، و پزشک شماره ۲، سه بیمار و دو بیمار بهبود یافته داشت، در خروجی نام و نام خانوادگی پزشک دوم زودتر از اولی می آید. پس از پیاده سازی صحیح این متد، تست **SortingDoctorsTests** پاس خواهد شد.

موفق و سلامت باشید.