دانشکده مهندسی کامپیوتر

برنامه‌سازی پیشرفته
آزمون عملی اول(بخش اول)

مدرس: سید صالح اعتمادی
طرح تمرین: امید میرزاجانی

چهارشنبه ۱۰ اردیبهشت ۹۹

فهرست مطالب

۱ مقدمه

۱.۱ موارد مورد توجه

- توجه داشته باشید که برای کسب نمره‌ی قبولی درس کسب حداقل نصف نمره‌ی هر سری تمرین و آزمون الزامی می‌باشد.
- 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!
- استفاده از ویدیوهای آموزشی حین امتحان مجاز نیست.
- هرگونه استفاده از تلفن همراه حین امتحان مجاز نیست.
- صدا و صفحه نمایش شما باید از طریق نرم‌افزار Flashback recorder به طور کامل از ابتدا تا انتهای امتحان ضبط و ذخیره شود.

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:

```
mkdir Exam1B_cs
cd Exam1B_cs
dotnet new sln
mkdir Exam1B_cs
cd Exam1B_cs
dotnet new console
cd ..
dotnet sln add Exam1B_cs\Exam1B_cs.csproj
mkdir Exam1B_cs.Tests
cd Exam1B_cs.Tests
dotnet new mstest
dotnet add reference ..\Exam1B_cs\Exam1B_cs.csproj
cd ..
dotnet sln add Exam1B_cs.Tests\Exam1B_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_Exam1B`	`Exam1B`	`Exam1B`

* در کل یک دیرکتوری داخل Assignments به نام Exam۱ بسازید و داخل آن،دو فولدر به نام های Exam۱B و Exam۱ داشته باشید و فایل های مربوطه را داخل دیرکتوری مربوطه بگذارید.

۳ مسائل بامزه C#

همانطورکه **دیمو** [۱] را میشناسید، به تازگی علاقه مند به صنعت کامپیوتر شده است و مانند بقیه برای پیشرفت اوضاع درسی خود، در حال تمرین برنامه نویسی است. به او کمک کنید تا تکلیف خود را انجام دهد.

۱.۳ Reversing

متد `Reverse` را در کلاس Program به گونه ای پیاده سازی کنید، که یک آرایه بگیرد و آن را برعکس کند. به طور مثال اگر ورودی شامل اعداد (۱و۴و۵و۲و۳) باشد، خروجی باید (۳و۲و۵و۴و۱) باشد. دقت کنید که این متد باید برای همه انواع داده ای کار کند. [۲]
پس از پیاده سازی صحیح تست های `ReverseIntTest` و `ReverseStringTest` پاس خواهند شد.

۲.۳ Interface / Class

۱.۲.۳ IPlayer

اینترفیس IPlayer را به گونه ای پیاده سازی کنید که شامل ویژگی های

- `height` از نوع `bool`
- `speed` از نوع `bool`

و متد `Post()` با نوع خروجی `string` باشد.
پس از پیاده سازی صحیح تست `IPlayerTest` پاس خواهند شد.

۲.۲.۳ Athlete

کلاس Athlete را به گونه ای پیاده سازی کنید که اولاً یک Player باشد، یعنی ویژگی ها و متدهای IPlayer را دارا باشد. و هم چنین علاوه بر آن دارای یک ویژگی به نام `name` از نوع `string` و متد `Post()` با نوع خروجی `string` باشد.
ابتدا سازنده این کلاس را تکمیل کنید و با پیاده سازی صحیح آن، تست `AthleteCostructorTest` پاس خواهد شد.
سپس متد `Post()` را به گونه ای پیاده سازی کنید که مشخص کنید این ورزشکار برای چه ورزشی مناسب است.

- اگر فرد قد بلند و سرعت بالایی داشت، برای بسکتبال مناسب است.
- اگر فرد فقط قد بلندی داشت، برای والیبال مناسب است.
- اگر فرد فقط سرعت بالایی داشت، برای دویدن مناسب است.
- اگر فرد هیچ یک از دو ویژگی ذکر شده را نداشت، برای کشتی مناسب است.

هم چنین دقت کنید که خروجی داده شده باید به صورتِ نامِ فرد در ابتدای رشته و سپس عبارت `is appropriate for` و سپس رشته ورزشی مناسب آن شخص باشد. پس از پیاده سازی صحیح تست `AthletePost` پاس خواهند شد.

۳.۳ Exceptions

خواندن متون فینگیلیش یکی از معضلات دیمو شده است و میخواهد کلماتی که دارای حروف صدادار نیستند را شناسایی کند.
متد `Vowels` را در کلاس Program به گونه ای پیاده سازی کنید که یک ورودی از نوع رشته بگیرد و اگر آن رشته شامل حروف صدادار بود، استثنایی از نوع `FormatException` پرتاب کند. در غیر این صورت رشته `Not Found!` را به عنوان خروجی بازگرداند.
پس از پیاده سازی صحیح، تست های `VowelSound` و `VowelNotFound` پاس خواهند شد.

موفق و سلامت باشید.

---

[۱] Dimo
[۲] Generic Method