



دانشگاه علم و صنعت ایران

دانشکده‌ی مهندسی کامپیوتر

برنامه‌سازی پیشرفته (سی شارپ)  
تمرین جبرانی آزمون عملی سوم

علی حیدری  
استاد: سید صالح اعتمادی

مهلت ارسال: ۲۵ خرداد ۹۸

## فهرست مطالب

۳	مقدمه و آماده‌سازی	۱
۳	نکات مورد توجه	۱.۱
۳	آماده‌سازی‌های اولیه	۲.۱
۳	آماده‌سازی‌های مربوط به git	۱.۲.۱
۴	آماده‌سازی‌های مربوط به visual studio	۲.۲.۱
۴	پیاده‌سازی تمرین	۲
۴	مجموعه تست‌های Basics	۱.۲
۴	تست CalculateSum	۱.۱.۲
۴	تست CalculateSumInvalid	۲.۱.۲
۵	تست CalculateSumFormatException	۳.۱.۲
۵	تست TryCalculateSum	۴.۱.۲
۵	تست PIPrecision	۵.۱.۲
۵	تست Fibonacci	۶.۱.۲
۵	تست RemoveDuplicatesDebug	۷.۱.۲
۵	مجموعه تست‌های ThreadsAndEvents	۲.۲
۵	تست GetElapsedTimes	۱.۲.۲
۵	تست DuplicateNumberAdded	۲.۲.۲
۵	تست MakeItFaster	۳.۲.۲
۶	مجموعه تست‌های MessageAnalysis	۳.۲
۶	تست MostRepliedMessage	۱.۳.۲
۶	تست MostPostedMessagePersons	۲.۳.۲
۶	تست MostActiveAtMidNight	۳.۳.۲
۶	تست MostQuestionsWithNoAnswer	۴.۳.۲
۶	مجموعه تست‌های Inheritance	۴.۲
۶	تست Person	۱.۴.۲
۶	تست Student	۲.۴.۲
۶	تست Employee	۳.۴.۲
۷	تست Teacher	۴.۴.۲
۷	مجموعه تست‌های CustomStringOperator	۵.۲
۷	تست StringExplicitOperator	۱.۵.۲
۷	تست StringReverseExplicitOperator	۲.۵.۲
۷	تست StringPlusPlusOperator	۳.۵.۲
۷	تست StringMinusMinusOperator	۴.۵.۲
۷	ارسال تمرین	۳
۷	مشاهده‌ی وضعیت اولیه‌ی فایل‌ها	۱.۳
۷	اضافه کردن فایل‌های تغییر یافته به stage	۲.۳
۸	commit کردن تغییرات انجام شده	۳.۳
۸	ارسال تغییرات انجام شده به Remote repository	۴.۳
۹	ساخت Pull Request	۵.۳
۹	ارسال Pull Request به بازبیننده	۶.۳

## ۱ مقدمه و آماده‌سازی

### ۱.۱ نکات مورد توجه

- دقیقا از فایل‌های پایه‌ای که برای این تمرین در اختیار شما قرار گرفته است استفاده کنید.
- استفاده از کدهایی که برای امتحان زده‌اید بلامانع است.
- به نمره‌ی کسانی که تا پایان مهلت تعیین شده تست‌های بیشتری پاس کنند به ازای هر تست ۲۰ درصد از نمره‌ی تست به نمره‌ی تست متناظر آن در امتحان افزوده خواهد شد.
- مبنای درس، اعتماد بر پاسخ‌های ارسال شده است؛ بنابراین ارسال پاسخ در ریبازیتوری 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.
- انجام هیچ تغییری روی شاخه‌ی `fb_E2` مجاز نیست.
- 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 پایانی موکول نکنید.
- همکاری و هم‌فکری شما در حل تمرین مانعی ندارد، اما پاسخ‌های ارسال هر کس حتما باید توسط خود او نوشته شده باشد.
- 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!

### ۲.۱ آماده‌سازی‌های اولیه

قواعد نام‌گذاری تمرین را از جدول ۱ مطالعه کنید.

جدول ۱: قراردادهای نام‌گذاری تمرین

Naming conventions					
Branch	Directory	Solution	Project	Test Project	Pull Request
fb_E2-C	E2-C	E2-C	E2-C	E2-CTests	E2-C

#### ۱.۲.۱ آماده‌سازی‌های مربوط به git

✓ ابتدا به شاخه‌ی `master` بروید.

```
1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2)
2 $ git checkout master
3 Switched to branch 'master'
4 Your branch is up to date with 'origin/master'.
```

✓ تغییرات انجام شده در `Remote Repository` را دریافت کنید.

```
1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (master)
2 $ git pull
3 remote: Azure Repos
4 remote: Found 8 objects to send. (90 ms)
5 Unpacking objects: 100% (8/8), done.
6 From https://9752XXXX.visualstudio.com/AP97982/_git/AP97982
7    e7fd3b5..2cc74de master -> origin/master
8 Checking out files: 100% (266/266), done.
9 Updating e7fd3b5..2cc74de
10 Fast-forward
11 .gitattributes | 63 +
12 E2/E2.sln | 37 +
13 E2/E2/E2.csproj | 61 +
14 E2/E2/App.config | 6 +
```

```

15 E2/E2/Program.cs | 15 +
16 E2/E2/Properties/AssemblyInfo.cs | 36 +
17 .
18 .
19 .
    
```

✓ یک شاخه‌ی جدید با نام `fb_E2-C` بسازید و تغییر شاخه دهید.

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (master)
2 $ git checkout -b fb_E2-C
3 Switched to a new branch 'fb_E2-C'
4 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
5 $
    
```

توصیه می‌شود پس از پیاده‌سازی هر کلاس تغییرات انجام شده را `commit` و `push` کنید.

## ۲.۲.۱ آماده‌سازی‌های مربوط به visual studio

یک پروژه‌ی جدید طبق قراردادهای نام‌گذاری موجود در جدول ۱ در ریشه‌ی ریپازیتوری git خود بسازید. ساختار فایل پایه‌ای که در اختیار شما قرار می‌گیرد به صورت زیر است:

```

1 E2-C
2 +---Project
3 | | E2-C-Basics.cs
4 | | E2-C-DotNetInterfaces.cs
5 | | E2-C-Events.cs
6 | | E2-C-Inheritance.cs
7 | | E2-C-Threading.cs
8 | |
9 | \---Linq
10 | E2-C-MessageAnalysis.cs
11 | E2-C-MessageData.cs
12 |
13 \---ProjectTests
14 chats.csv
15 E2-C-BasicsTests.cs
16 E2-C-CustomStringOperatorTests.cs
17 E2-C-InheritanceTests.cs
18 E2-C-MessageAnalysisTests.cs
19 E2-C-ThreadsAndEvents.cs
    
```

در فایل پایه دو پوشه وجود دارد شما باید فایل(های) موجود در پوشه‌ی Project را به پروژه‌ی اصلی (E2-C) و فایل(های) موجود در پوشه‌ی ProjectTests را به پروژه‌ی تست (E2-C-Tests) اضافه کنید.

## ۲ پیاده‌سازی تمرین

### ۱.۲ مجموعه تست‌های Basics

هدف این بخش سنجش آشنایی ابتدایی با پردازش رشته‌ها، پارامترهای خروجی و مدیریت خطا می‌باشد.

#### ۱.۱.۲ تست CalculateSum

متد `CalculateSum` را بگونه‌ای پیاده‌سازی کنید که یک رشته شامل تعداد عدد و عملگر جمع را به عنوان پارامتر دریافت کرده و مقدار معادل عددی را برگرداند.  $\frac{21}{1}$

#### ۲.۱.۲ تست CalculateSumInvalid

در صورتیکه رشته ورودی یک عبارت ریاضی صحیح نباشد لازم است متد `CalculateTest` یک `Exception` از نوع `InvalidDataException` پرتاب کند.  $\frac{20}{2}$

### ۳.۱.۲ تست CalculateSumFormatException

همچنین لازم است در صورتیکه زیررشته موجود در عبارت بین علامت‌های جمع از نوع عدد صحیح نباشد `Exception` متناظر آن رخ دهد. ۱۹/۳

### ۴.۱.۲ تست TryCalculateSum

متد `TryCalculate` را به گونه‌ای پیاده‌سازی کنید که در صورت بروز خطا `false` برگرداند و در هیچ شرایطی `Exception` رخ ندهد. در این متد مقدار عددی محاسبه شده بصورت یک پارامتر از نوع `out` بازگردانده می‌شود. ۱۸/۴

### ۵.۱.۲ تست PIPrecision

یکی از راه‌های محاسبه عدد پی استفاده از دنباله زیر است.

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \dots = \frac{\pi}{4}$$

سوال این است که برای محاسبه عدد پی تا هفت رقم معنادار این دنباله را تا عبارت چندم باید محاسبه کنیم. راهنمایی: `Math.PI` عدد تا دقت بیش از ده رقم اعشار. `Math.Round(x, 10)`: عدد ورودی را به ده رقم معنادار رند می‌کند. حال متد `PIPrecision` را به گونه‌ای پیاده‌سازی کنید که پاسخ این سوال را پیدا کند. ۱۷/۵

### ۶.۱.۲ تست Fibonacci

`Extension` متد به نام `Fibonacci` را برای نوع داده‌ای `int` به گونه‌ای پیاده‌سازی کنید که عدد `n`ام در دنباله بیبوناچی را برگرداند و این تست پاس شود. ۱۶/۶

### ۷.۱.۲ تست RemoveDuplicatesDebug

کلاس `FullName` متد `RemoveDuplicates` پیاده‌سازی شده‌اند ولی این تست پاس نمی‌شود. کلاس `FullName` را بگونه‌ای تغییر دهید که تست زیر پاس شود. ۱۵/۷

## ۲.۲ مجموعه تست‌های ThreadsAndEvents

تست‌های این بخش مربوط به `IEnumerable/foreach/yield` و `Event` و `Thread` هستند

### ۱.۲.۲ تست GetElapsedTimes

لازم است متد `GetElapsedTimes` به گونه‌ای پیاده‌سازی شود که فاصله دسترسی به مقداری این مجموعه را برگرداند. به این شکل که مقدار اولی که بر میگردد همیشه صفر است و مقادیر بعدی برابر با تعداد میلی ثانیه‌ای است که از زمان دسترسی به مقدار قبلی گذشته است. پارامتر ورودی این متد حداکثر طول این رشته است. در این متد می‌توانید از کلاس `Stopwatch` برای اندازه‌گیری زمان استفاده کنید. برای شیوه استفاده از آن به `MSDN` یا تست‌های پیاده‌سازی شده از که این کلاس استفاده می‌کنند مراجعه کنید. ۱۴/۸

### ۲.۲.۲ تست DuplicateNumberAdded

کلاس `DuplicateNumberDetector` را به گونه‌ای پیاده‌سازی کنید که یک متد `AddNumber` داشته باشد که یک عدد صحیح به عنوان ورودی دریافت می‌کند. چنانچه این متد با یک عدد تکراری صدا زده شود لازم است `Event` به نام `DuplicateNumberAdded` اتفاق بیافتد که به عنوان پارامتر تعداد کل دفعاتی که عدد تکراری اضافه شده را برگرداند. ۱۳/۹

### ۳.۲.۲ تست MakeItFaster

متد `MakeItFasterTest` را به گونه‌ای پیاده‌سازی کنید که تعدادی `Action` از ورودی دریافت کرده و آنها را همزمان اجرا کند. این متد تنها زمانی باید برگردد که تمام `delegate` ها اجرا شده باشند. ۱۲/۱۰

## ۳.۲ مجموعه تست‌های MessageAnalysis

فایل `chats.csv` حاوی پیام‌های گروه تلگرامی درس AP این ترم به شما داده شده است. مشابه تمرین ۱۲ `VisualBasic Reference` را به پروژه‌ی اصلی (E2-C) اضافه کنید. کلاس‌های لازم برای پارس کردن مشابه با تمرین ۱۲ به شما داده شده است.

### ۱.۳.۲ تست MostRepliedMessage

برای پاس شدن این تست شما باید از بین کل پیام‌های داده شده پیامی را که بیشترین پاسخ<sup>۱</sup> را دریافت کرده در قابل یک شی از نوع `MessageData` برگردانید. ۱۱/۱۱

### ۲.۳.۲ تست MostPostedMessagePersons

برای پاس شدن این تست شما باید ۵ نفری را که بیشترین تعداد پیام را در گروه ارسال کرده‌اند به جز استاد "`Sauleh Eetemadi`" و سرتی‌ای "`Ali Heydari`" به صورت یک آرایه از زوج مرتب‌ها<sup>۲</sup> برگردانید. عضو اول این زوج مرتب نام نویسنده‌ی پیام، و عضو دوم آن زوج مرتب تعداد کل پیام‌هایی است که آن فرد در گروه ارسال کرده می‌باشد. ۱۰/۱۲

### ۳.۳.۲ تست MostActiveAtMidNight

برای پاس شدن این تست شما باید ۵ نفر اولی که بیشترین تعداد پیام را در ساعات اولیه‌ی بامداد `00:00` تا `04:00` ارسال کرده‌اند به صورت آرایه‌ای از زوج مرتب‌ها برگردانید. عضو اول این زوج مرتب نام نویسنده‌ی پیام، و عضو دوم آن زوج مرتب تعداد کل پیام‌هایی است که آن فرد در گروه ارسال کرده می‌باشد. ۹/۱۳

### ۴.۳.۲ تست MostQuestionsWithNoAnswer

با فرض این که هر پیامی که دارای کاراکتر علامت سوال (هم کاراکتر فارسی علامت سوال '؟' و هم کاراکتر انگلیسی علامت سوال '?') باشد یک سوال محسوب می‌شود کدام فرد بیشترین تعداد سوال بدون پاسخ را پرسیده است؟ نام آن فرد را برگردانید. ۸/۱۴

## ۴.۲ مجموعه تست‌های Inheritance

کلاس‌های `Person`، `Employee`، `Student`، `Teacher` باید به گونه‌ای پیاده‌سازی شوند که ارث‌بری‌ها، متدهای `virtual` `abstract`، `override` به درستی و مطابق با تست‌ها تعریف شوند.

### ۱.۴.۲ تست Person

`Person` یک کلاس `abstract` است با یک سازنده که نام و مونث بودن را به عنوان پارمتر دریافت می‌کند. این کلاس یک `virtual getter` به نام `Name` دارد که نام با پیشوند خانم یا آقا را برمی‌گرداند. علاوه بر این یک `getter` به نام `IsFemale` نیز دارد. همچنین دارای یک `abstract getter` به نام `LunchRate` ۷/۱۵

### ۲.۴.۲ تست Student

کلاس `Student` از `Person` به ارث برده و نرخ ناهار را برابر ۲۰۰۰ تومان قرار می‌دهد. این کلاس را بگونه‌ای پیاده‌سازی کنید که تست زیر پاس شود. ۶/۱۶

### ۳.۴.۲ تست Employee

کلاس `Employee` از `Person` به ارث برده و نرخ ناهار را برابر ۵۰۰۰ تومان قرار می‌دهد. علاوه بر این یک متد `virtual` به نام `CalculateSalary` اضافه می‌کند که حقوق کارمند را برابر ساعتی ۵۰۰۰ تومان حساب می‌کند. این کلاس را بگونه‌ای پیاده‌سازی کنید که تست زیر پاس شود. ۵/۱۷

<sup>1</sup>reply

<sup>2</sup>Tuple

## ۴.۴.۲ تست Teacher

کلاس `Teacher` از `Employee` به ارث برده و نرخ ناهار را برابر ۱۰۰۰۰ تومان قرار می‌دهد. علاوه بر این متد `CalculateSalary` را `override` می‌کند که حقوق استاد را برابر ساعتی ۲۰۰۰۰ تومان حساب می‌کند. این کلاس را بگونه‌ای پیاده‌سازی کنید که تست زیر پاس شود. ۴/۱۸

## ۵.۲ مجموعه تست‌های CustomStringOperator

کلاس `MyString` باید بگونه‌ای پیاده‌سازی شده و اپراتورهای لازم برای آن `overload` شده یا متدهای لازم `override` شوند که تست‌های این مجموعه پاس بشوند.

### ۱.۵.۲ تست StringExplicitOperator

اپراتور تبدیل نوع داده‌ای از `System.String` به `MyString` را پیاده‌سازی کرده و متدها و اپراتورهای لازم برای مقایسه این دو نوع را به گونه‌ای پیاده‌سازی کنید که این تست پاس شود. ۳/۱۹

### ۲.۵.۲ تست StringReverseExplicitOperator

اپراتور تبدیل نوع داده‌ای از `MyString` به `System.String` را پیاده‌سازی کرده و متدها و اپراتورهای لازم برای مقایسه این دو نوع را به گونه‌ای پیاده‌سازی کنید که این تست پاس شود. ۲/۲۰

### ۳.۵.۲ تست StringPlusPlusOperator

اپراتور `++` را برای کلاس `MyString` به گونه‌ای پیاده‌سازی کنید که رشته حرفی را به حروف بزرگ تبدیل کند. ۱/۲۱

### ۴.۵.۲ تست StringMinusMinusOperator

اپراتور `--` را برای کلاس `MyString` به گونه‌ای پیاده‌سازی کنید که رشته حرفی را به حروف کوچک تبدیل کند. ۰/۲۲

## ۳ ارسال تمرین

در اینجا یک‌بار دیگر ارسال تمرین را با هم مرور می‌کنیم:

### ۱.۳ مشاهده‌ی وضعیت اولیه‌ی فایل‌ها

ابتدا وضعیت فعلی فایل‌ها را مشاهده کنید:

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
2 $ git status
3 On branch fb_E2-C
4 Untracked files:
5   (use "git add <file>..." to include in what will be committed)
6
7   E2-C/
8
9 nothing added to commit but untracked files present (use "git add" to track)
    
```

همان‌طور که مشاهده می‌کنید فولدر `E2-C` و تمام فایل‌ها و فولدرهای درون آن در وضعیت `Untracked` قرار دارند و همان‌طور که در خط آخر خروجی توضیح داده شده برای `commit` کردن آن‌ها ابتدا باید آن‌ها را با دستور `git add` وارد `stage` کنیم.

### ۲.۳ اضافه کردن فایل‌های تغییر یافته به stage

حال باید فایل‌ها و فولدرهایی را که در `stage` قرار ندارند را وارد `stage` کنیم. برای این کار از دستور `git add` استفاده می‌کنیم.

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
2 $ git add E2-C/*
    
```

حال دوباره وضعیت فایل‌ها و فولدرها را مشاهده می‌کنیم:

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
2 $ git status
3 On branch fb_E2-C
4 Changes to be committed:
5   (use "git reset HEAD <file>..." to unstage)
6
7   new file:   E2-C/E2-C.sln
8   new file:   E2-C/E2-C/E2-C.csproj
9   new file:   E2-C/E2-C/App.config
10  new file:   E2-C/E2-C/Program.cs
11  new file:   E2-C/E2-C/Properties/AssemblyInfo.cs
12  new file:   E2-C/E2-C/Tests/E2-C/Tests.csproj
13  new file:   E2-C/E2-C/Tests/Properties/AssemblyInfo.cs
14  new file:   E2-C/E2-C/Tests/packages.config
15  .
16  .
17  .
    
```

همانطور که مشاهده می‌کنید فولدر E2-C و تمام فولدرها و فایل‌های درون آن (به جز فایل‌هایی که در gitignore معین کرده‌ایم) وارد stage شده‌اند.

### ۳.۳ commit کردن تغییرات انجام شده

در گام بعدی باید تغییرات انجام شده را commit کنیم. فراموش نکنید که فقط فایل‌هایی را می‌توان commit کرد که در stage قرار داشته باشند. با انتخاب یک پیام مناسب تغییرات صورت گرفته را commit می‌کنیم:

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
2 $ git commit -m "Implement Exam2 Compensation"
3 [fb_E2-C c1f21df] Implement Exam2 Compensation
4 15 files changed, 595 insertions(+)
5 create mode 100644 E2-C/E2-C.sln
6 create mode 100644 E2-C/E2-C/E2-C.csproj
7 create mode 100644 E2-C/E2-C/App.config
8 create mode 100644 E2-C/E2-C/Program.cs
9 create mode 100644 E2-C/E2-C/Properties/AssemblyInfo.cs
10 create mode 100644 E2-C/E2-C/Tests/E2-C/Tests.csproj
11 create mode 100644 E2-C/E2-C/Tests/Properties/AssemblyInfo.cs
12 create mode 100644 E2-C/E2-C/Tests/packages.config
13 .
14 .
15 .
    
```

### ۴.۳ ارسال تغییرات انجام شده به Remote repository

گام بعدی ارسال تغییرات انجام شده به مخزن<sup>۳</sup> Remote است.

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
2 $ git push origin fb_E2-C
3 Enumerating objects: 25, done.
4 Counting objects: 100% (25/25), done.
5 Delta compression using up to 8 threads
6 Compressing objects: 100% (22/22), done.
7 Writing objects: 100% (25/25), 9.56 KiB | 890.00 KiB/s, done.
8 Total 25 (delta 4), reused 0 (delta 0)
9 remote: Analyzing objects... (25/25) (5 ms)
10 remote: Storing packfile... done (197 ms)
11 remote: Storing index... done (84 ms)
12 To https://9752XXXX.visualstudio.com/AP97982/_git/AP97982
    
```

<sup>3</sup>Repository



```
13 * [new branch] fb_E2-C -> fb_E2-C
```

### ۵.۳ ساخت Pull Request

با مراجعه به سایت [Azure DevOps](#) یک Pull Request جدید با نام E2-C بسازید به طوری که امکان `merge` کردن شاخه‌ی `fb_E2-C` را بر روی شاخه‌ی `master` را بررسی کند. (این کار در صورتی انجام می‌شود که کد شما کامپایل شود و همچنین تست‌های آن پاس شوند) در نهایت با انتخاب گزینه‌ی `set auto complete` در صفحه‌ی Pull Request مربوطه تعیین کنید که در صورت وجود شرایط `merge` این کار انجام شود. دقت کنید که گزینه‌ی `Delete source branch` نباید انتخاب شود.

### ۶.۳ ارسال Pull Request به بازبیننده

در نهایت Pull Request ساخته شده را برای بازبینی، با بازبیننده‌ی خود به اشتراک بگذارید.