دانشکده‌ی مهندسی کامپیوتر

برنامه‌سازی پیشرفته (سی‌شارپ)
تمرین جبرانی آزمون عملی سوم

علی حیدری
استاد: سید صالح اعتمادی

مهلت ارسال: ۲۵ خرداد ۹۸

فهرست مطالب

۱ مقدمه و آماده‌سازی ۳
۱.۱ نکات مورد توجه ۳
۲.۱ آماده‌سازی‌های اولیه ۳
۱.۲.۱ آماده‌سازی‌های مربوط به git ۳
۲.۲.۱ آماده‌سازی‌های مربوط به visual studio ۴

۲ پیاده‌سازی تمرین ۴
۱.۲ مجموعه تست‌های Basics ۴
۱.۱.۲ تست CalculateSum ۴
۲.۱.۲ تست CalculateSumInvalid ۴
۳.۱.۲ تست CalculateSumFormatException ۵
۴.۱.۲ تست TryCalculateSum ۵
۵.۱.۲ تست PIPrecision ۵
۶.۱.۲ تست Fibonacci ۵
۷.۱.۲ تست RemoveDuplicatesDebug ۵
۲.۲ مجموعه تست‌های ThreadsAndEvents ۵
۱.۲.۲ تست GetElapsedTimes ۵
۲.۲.۲ تست DuplicateNumberAdded ۵
۳.۲.۲ تست MakeItFaster ۵
۳.۲ مجموعه تست‌های MessageAnalysis ۶
۱.۳.۲ تست MostRepliedMessage ۶
۲.۳.۲ تست MostPostedMessagePersons ۶
۳.۳.۲ تست MostActiveAtMidNight ۶
۴.۳.۲ تست MostQuestionsWithNoAnswer ۶
۴.۲ مجموعه تست‌های Inheritance ۶
۱.۴.۲ تست Person ۶
۲.۴.۲ تست Student ۶
۳.۴.۲ تست Employee ۶
۴.۴.۲ تست Teacher ۷
۵.۲ مجموعه تست‌های CustomStringOperator ۷
۱.۵.۲ تست StringExplicitOperator ۷
۲.۵.۲ تست StringReverseExplicitOperator ۷
۳.۵.۲ تست StringPlusPlusOperator ۷
۴.۵.۲ تست StringMinusMinusOperator ۷

۳ ارسال تمرین ۷
۱.۳ مشاهده‌ی وضعیت اولیه‌ی فایل‌ها ۷
۲.۳ اضافه کردن فایل‌های تغییر یافته به stage ۷
۳.۳ commit کردن تغییرات انجام شده ۸
۴.۳ ارسال تغییرات انجام شده به Remote repository ۸
۵.۳ ساخت Pull Request ۹
۶.۳ ارسال Pull Request به بازبیننده ۹

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 تعیین‌شده 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 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.
- 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 اعلام‌شده است. توصیه می‌شود نوشتن تمرین را به روزهای پایانی موکول نکنید.
- هم‌کاری و هم‌فکری شما در حل تمرین مانعی ندارد، اما پاسخ ارسالی 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.

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

✓ تغییرات انجام‌شده در `Remote Repository` را دریافت کنید.

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

```
 E2/E2/Program.cs                                |     15 +
 E2/E2/Properties/AssemblyInfo.cs                |     36 +
 .
 .
 .
```

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

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

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:

```
E2-C
  +---Project
  |   |   E2-C-Basics.cs
  |   |   E2-C-DotNetInterfaces.cs
  |   |   E2-C-Events.cs
  |   |   E2-C-Inheritance.cs
  |   |   E2-C-Threading.cs
  |   |
  |   \---Linq
  |           E2-C-MessageAnalysis.cs
  |           E2-C-MessageData.cs
  |
  \---ProjectTests
          chats.csv
          E2-C-BasicsTests.cs
          E2-C-CustomStringOperatorTests.cs
          E2-C-InheritanceTests.cs
          E2-C-MessageAnalysisTests.cs
          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-CTests) 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. [۲۱/۱]

پاسخ.

```
public static int CalculateSum(string expression)
  {
      var toks = expression.Split(new char [ ]  { '+' },
          StringSplitOptions.RemoveEmptyEntries);
```

```
        if (toks.Length < 2)
            throw new InvalidDataException();

        int sum = 0;
        foreach (var tok in toks)
            sum += int.Parse(tok);

        return sum;
    }
```

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. ۲۰/۲

پاسخ.

```
public static int CalculateSum(string expression)
    {
        var toks = expression.Split(new char [ ] { '+' },
            StringSplitOptions.RemoveEmptyEntries);
        if (toks.Length < 2)
            throw new InvalidDataException();

        int sum = 0;
        foreach (var tok in toks)
            sum += int.Parse(tok);

        return sum;
    }
```

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

پاسخ.

```
public static int CalculateSum(string expression)
    {
        var toks = expression.Split(new char [ ] { '+' },
            StringSplitOptions.RemoveEmptyEntries);
        if (toks.Length < 2)
            throw new InvalidDataException();

        int sum = 0;
        foreach (var tok in toks)
            sum += int.Parse(tok);

        return sum;
    }
```

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

پاسخ.

```
public static bool TryCalculateSum(string expression, out int value)
  {
      bool bSuccess = false;
      value = 0;
      try
      {
          value = CalculateSum(expression);
          bSuccess = true;
      }
      catch
      { }
      return bSuccess;

  }
```

۵.۱.۲ تست PIPrecision

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

$$۱ - \frac{۱}{۳} + \frac{۱}{۵} - \frac{۱}{۷} + \frac{۱}{۹} - \cdots = \frac{\pi}{۴}.$$

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

پاسخ.

```
public static int PIPrecision()
  {

      double piSum = 0;
      double denominator = 1;
      double sign = 1;
      int iteration = 0;
      while (Math.Round(4 * piSum, 7)  !=  Math.Round(Math.PI, 7))
      {
          piSum  +=  (sign / denominator);
          sign = sign * -1;
          denominator  +=  2;
          iteration++;
      }

      return iteration;

  }
```

۶.۱.۲ تست Fibonacci

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

پاسخ.

```
 public static int Fibonacci(this int n)
  {

      return n  <=  1 ? 1 : Fibonacci(n - 1) + Fibonacci(n - 2);

  }
```

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

پاسخ.

```
public static void RemoveDuplicates<T>(ref T[] list)
  {
      List<T> newList = new List<T>();
      foreach(var item in list)
      {
          if (!Contains(newList, item))
              newList.Add(item);
      }
      list = newList.ToArray();
  }
```

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 میگرداند 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

پاسخ.

```
public static IEnumerable<long> GetElapsedTimes(int max=100)
  {
      Stopwatch sw = Stopwatch.StartNew();
      while (true)
      {
          var elapsed = sw.ElapsedMilliseconds;
          sw.Restart();
          yield return elapsed;

          if (max-- == 0)
              break;
      }
  }
```

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 می کند. 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

پاسخ.

```
public class DuplicateNumberDetector
  {
      private List<int> _Numbers = new List<int>();

      public IEnumerable<int> Numbers => Numbers;

      public void AddNumber(int n)
```

```
        {
            bool bDuplicate = _Numbers.Contains(n);
            _Numbers.Add(n);
            if (bDuplicate && DuplicateNumberAdded != null)
                DuplicateNumberAdded(n);
        }

        public event Action<int> DuplicateNumberAdded;
    }
```

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

پاسخ.

```
public static void MakeItFaster(params Action[] actions)
    {
        List<Task> tList = new List<Task>();
        foreach (var action in actions)
        {
            Task t = Task.Run(action);
            tList.Add(t);
        }
        Task.WaitAll(tList.ToArray());
    }
```

۳.۲ مجموعه تست‌های `MessageAnalysis`

فایل `chats.csv` حاوی پیام‌های گروه تلگرامی درس AP این ترم به شما داده شده است. مشابه تمرین ۱۲ `Reference` VisualBasic 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[1] را دریافت کرده در قابل یک شی از نوع `MessageData` برگردانید. ۱۱/۱۱

پاسخ.

```
public MessageData MostRepliedMessage()
    {

        int? messageId = Messages
            .Where(m => m.ReplyMessageId != null)
            .GroupBy(m => m.ReplyMessageId)
            .OrderByDescending(gp => gp.Count())
            .First()
            .Key;
        return Messages.FirstOrDefault(m => m.Id == messageId);

    }
```

[1] reply

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[۲] 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

پاسخ.

```
public class DuplicateNumberDetector
  {
      private List < int >  _Numbers = new List < int > ();

      public IEnumerable < int >  Numbers  =>  Numbers;

      public void AddNumber(int n)
      {
          bool bDuplicate = _Numbers.Contains(n);
          _Numbers.Add(n);
          if (bDuplicate && DuplicateNumberAdded  !=  null)
              DuplicateNumberAdded(n);
      }

      public event Action < int >  DuplicateNumberAdded;
  }
```

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

پاسخ.

```
public Tuple < string, int > [ ]  MostActivesAtMidNight()
  {
      return Messages
          .Where(m  =>  m.DateTime.Hour  >=  0 && m.DateTime.Hour  <=  4)
          .GroupBy(m  =>  m.Author)
          .Select(m  =>  Tuple.Create(m.Key, m.Count()))
          .OrderByDescending(gp  =>  gp.Item2)
          .Take(5)
          .ToArray();
  }
```

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

پاسخ.

```
public string StudentWithMostUnansweredQuestions()
  {
      return Messages
          .Where(m  =>  m.Content.Contains('?') || m.Content.Contains('؟')
          .GroupBy(m  =>  m.Author)
          .Select(m  =>  new {Count = m.Count(), Author = m.Key, ids = m.Select(x
              =>  x.Id)})
```

[۲] Tuple

```
            .Where(g =>
                Messages
                    .Select(m => g.ids.Any(x => x == m.ReplyMessageId)).Any())
            .OrderByDescending(gp => gp.Count)
            .First().Author;
    }
```

۴.۲ مجموعه تست‌های Inheritance

کلاس های `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

پاسخ.

```
public abstract class Person
    {
        protected readonly string _Name;
        public virtual string Name => $"{PreFix} {_Name}";
        protected string PreFix => IsFemale ? FemalePrefix : MalePrefix;
        protected const string FemalePrefix = "خانم";
        protected const string MalePrefix = "آقای";
        public readonly bool IsFemale;

        public Person(string Name, bool isFemale)
        {
            _Name = Name;
            IsFemale = isFemale;
        }

        public abstract int LunchRate { get; }
    }
```

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

پاسخ.

```
public class Student : Person
    {
        public Student(string Name, bool isFemale) : base(Name, isFemale)
        {}

        public override int LunchRate => 2000;
    }
```

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

پاسخ.

```
public class Employee : Person
  {
      public Employee(string Name, bool isFemale) : base(Name, isFemale)
      {}

      public virtual int CalculateSalary(int hours) => hours * 5000;
      public override int LunchRate => 5000;
  }
```

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

پاسخ.

```
public class Teacher: Employee
  {
    public Teacher(string Name, bool isFemale) : base(Name, isFemale)
    {}

    public override string Name => $"استاد {base._Name}";
    public override int CalculateSalary(int hours) => hours * 20000;
    public override int LunchRate => 10000;

  }
```

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 پیاده سازی کنید که این تست پاس شود. ۳/۱۹

پاسخ.

```
public class MyString
  {
    public MyString(string s)
    {
        this.Text = s;
    }

    private string Text;
    public static explicit operator MyString(string s)
    {
        return new MyString(s);
    }
  }
```

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 پیاده سازی کنید که این تست پاس شود. ۲/۲۰

پاسخ.

```
public class MyString
  {
    public MyString(string s)
    {
        this.Text = s;
    }

    private string Text;
    public static explicit operator MyString(string s)
    {
        return new MyString(s);
    }
  }
```

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

پاسخ.

```
public static MyString operator++ (MyString s)
  {
      s.Text = s.Text.ToUpper();
      return s;
  }
```

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

پاسخ.

```
public static MyString operator-- (MyString s)
  {
      s.Text = s.Text.ToLower();
      return s;
  }
```

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:

```
Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
$ git status
On branch fb_E2-C
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    E2-C/

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.

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

حال دوباره وضعیت فایل‌ها و فولدرها را مشاهده می‌کنیم:

```
Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
$ git status
On branch fb_E2-C
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   E2-C/E2-C.sln
    new file:   E2-C/E2-C/E2-C.csproj
    new file:   E2-C/E2-C/App.config
    new file:   E2-C/E2-C/Program.cs
    new file:   E2-C/E2-C/Properties/AssemblyInfo.cs
    new file:   E2-C/E2-CTests/E2-CTests.csproj
    new file:   E2-C/E2-CTests/Properties/AssemblyInfo.cs
    new file:   E2-C/E2-CTests/packages.config
    .
    .
    .
```

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:

```
Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
$ git commit -m "Implement Exam2 Compensation"
[fb_E2-C c1f21df] Implement Exam2 Compensation
 15 files changed, 595 insertions(+)
 create mode 100644 E2-C/E2-C.sln
 create mode 100644 E2-C/E2-C/E2-C.csproj
 create mode 100644 E2-C/E2-C/App.config
 create mode 100644 E2-C/E2-C/Program.cs
 create mode 100644 E2-C/E2-C/Properties/AssemblyInfo.cs
 create mode 100644 E2-C/E2-CTests/E2-CTests.csproj
 create mode 100644 E2-C/E2-CTests/Properties/AssemblyInfo.cs
 create mode 100644 E2-C/E2-CTests/packages.config
.
.
.
```

۴.۳ ارسال تغییرات انجام شده به Remote repository

گام بعدی ارسال تغییرات انجام شده به مخزن[۳] `Remote` است.

```
Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_E2-C)
$ git push origin fb_E2-C
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
```

[۳] Repository

```
Compressing objects: 100% (22/22), done.
Writing objects: 100% (25/25), 9.56 KiB | 890.00 KiB/s, done.
Total 25 (delta 4), reused 0 (delta 0)
remote: Analyzing objects... (25/25) (5 ms)
remote: Storing packfile... done (197 ms)
remote: Storing index... done (84 ms)
To https://9752XXXX.visualstudio.com/AP97982/_git/AP97982
 * [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 درصورتی 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.