



دانشگاه علم و صنعت ایران

دانشکده مهندسی کامپیوتر

## برنامه‌سازی پیشرفته (سی شارپ) تمرین‌های سری هشتم (سربارگذاری عمل‌گرها)

علی حیدری  
استاد: سید صالح اعتمادی

مهلت ارسال: ۲۰ اردیبهشت ۱۳۹۸

### فهرست مطالب

۲	مقدمه و آماده‌سازی	۱
۲	نکات مورد توجه	۱.۱
۲	آماده‌سازی‌های اولیه	۲.۱
۲	آماده‌سازی‌های مربوط به git	۱.۲.۱
۳	آماده‌سازی‌های مربوط به visual studio	۲.۲.۱
۴	پیاده‌سازی تمرین	۲
۴	ساخت کلاس Human	۱.۲
۴	سربارگذاری عملگر جمع	۲.۲
۴	سربارگذاری عملگرهای < > ، < = > = و !=	۳.۲
۴	پیاده‌سازی متدهای Equals و GetHashCode	۴.۲
۴	ارسال تمرین	۳
۴	مشاهده وضعیت اولیه فایل‌ها	۱.۳
۴	اضافه کردن فایل‌های تغییر یافته به stage	۲.۳
۴	commit کردن تغییرات انجام شده	۳.۳
۵	ارسال تغییرات انجام شده به Remote repository	۴.۳
۵	ساخت Pull Request	۵.۳
۵	ارسال Pull Request به بازبیننده	۶.۳

## ۱ مقدمه و آماده‌سازی

### ۱.۱ نکات 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 حتماً 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.
- برای طرح سوال و پرسش و پاسخ از صفحه درس در [Quera](#) استفاده کنید.

### 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_A8	A8	A8	A8	A8Tests	HW8

#### ۱.۲.۱ آماده‌سازی‌های مربوط به git

اگر چه در کارگاه git مفاهیم و روش کار با آن آموزش داده شد اما بار دیگر در اینجا کارهایی را که باید در ابتدای تمرین انجام دهید را مرور می‌کنیم.

✓ ابتدا به شاخه‌ی `master` بروید.

```
1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_A6)
2 $ git checkout master
3 Switched to branch 'master'
4 Your branch is up to date with 'origin/master'.
```

✓ تغییرات انجام‌شده 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 A8/A8.sln | 37 +
13 A8/A8/A8.csproj | 61 +
14 A8/A8/App.config | 6 +
15 A8/A8/Program.cs | 15 +
```

```

16 A8/A8/Properties/AssemblyInfo.cs | 36 +
17 .
18 .
19 .

```

✓ یک شاخه‌ی جدید با نام `fb_A8` بسازید و تغییر شاخه دهید.

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (master)
2 $ git checkout -b fb_A8
3 Switched to a new branch 'fb_A8'
4 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_A8)
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. توجه داشته باشید که در این تمرین نوشتن تست برای تمامی متدهایی که پیاده‌سازی کرده‌اید بر عهده‌ی خودتان است و درستی تست‌ها از نظر کارایی و حسن انجام تست بررسی توسط بازبینندگان بررسی خواهد شد.

## ۲ پیاده‌سازی تمرین

### ۱.۲ ساخت کلاس `Human`

یک کلاس به نام `Human` بیازید که دارای ۴ ویژگی زیر است:

- نام: `FirstName`
- تاریخ تولد: `BirthDate`
- نام خانوادگی: `LastName`
- طول قد: `Height`

سپس برای این کلاس یک سازنده بسازید که تمامی ویژگی‌های آن را از ورودی بگیرد و یک شی معتبر از این نوع کلاس ایجاد کند.

### ۲.۲ سربارگذاری عملگر جمع

عملگر جمع را به گونه‌ای سربارگذاری کنید که شی حاصل ویژگی‌هایی به صورت زیر داشته باشد:

- نام شی عبارت `"ChildFirstName"` باشد
- تاریخ تولد آن روز ایجاد شی (روز فعلی سیستم) باشد
- نام خانوادگی شی عبارت `"ChildLastName"` باشد
- طول قد آن مقدار 30 باشد.

### ۳.۲ سربارگذاری عملگرهای `<`، `>`، `<=`، `>=` و `!=`، `==`

عملگرهای `<`، `>`، `<=`، `>=` و `!=`، `==` را به گونه‌ای سربارگذاری کنید که بر اساس سن شی آن‌ها را مقایسه کند.

### ۴.۲ پیاده‌سازی متدهای `Equals` و `GetHashCode`

#### `Equals`

این متد را بر اساس مقایسه‌ی تمامی ویژگی‌های دو شی پیاده‌سازی کنید. یعنی تنها در صورتی خروجی این متد `true` است که تمام ویژگی‌های هر دو شی با هم برابر باشند.

#### `GetHashCode`

این متد را به صورتی پیاده‌سازی کنید که `HashCode` ناشی از تلفیق \* تمامی ویژگی‌های شی را برگرداند.

\*concatenation

## ۳ ارسال تمرین

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

### ۱.۳ مشاهده وضعیت اولیه فایل‌ها

ابتدا وضعیت فعلی فایل‌ها را مشاهده کنید:

```
1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_A8)
2 $ git status
3 On branch fb_A8
4 Untracked files:
5   (use "git add <file>..." to include in what will be committed)
6
7   A8/
8
9 nothing added to commit but untracked files present (use "git add" to track)
```

همان‌طور که مشاهده می‌کنید فولدر A8 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_A8)
2 $ git add A8/*
```

حال دوباره وضعیت فایل‌ها و فولدرها را مشاهده می‌کنیم:

```
1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_A8)
2 On branch fb_A8
3 Changes to be committed:
4   (use "git reset HEAD <file>..." to unstage)
5
6   new file:   A8/A8.sln
7   new file:   A8/A8/A8.csproj
8   new file:   A8/A8/App.config
9   new file:   A8/A8/Program.cs
10  new file:   A8/A8/Properties/AssemblyInfo.cs
11  new file:   A8/A8Tests/A8Tests.csproj
12  new file:   A8/A8Tests/Properties/AssemblyInfo.cs
13  new file:   A8/A8Tests/packages.config
14  .
15  .
16  .
```

همان‌طور که مشاهده می‌کنید فولدر A8 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_A8)
2 $ git commit -m "Implement HW8"
3 [fb_A8 c1f21df] Implement HW8
4 15 files changed, 595 insertions(+)
5 create mode 100644 A8/A8.sln
6 create mode 100644 A8/A8/A8.csproj
```

```

7 create mode 100644 A8/A8/App.config
8 create mode 100644 A8/A8/Program.cs
9 create mode 100644 A8/A8/Properties/AssemblyInfo.cs
10 create mode 100644 A8/A8Tests/A8Tests.csproj
11 create mode 100644 A8/A8Tests/Properties/AssemblyInfo.cs
12 create mode 100644 A8/A8Tests/packages.config
13 .
14 .
15 .

```

### ۴.۳ ارسال تغییرات انجام شده به Remote repository

گام بعدی ارسال تغییرات انجام شده به Remote Repository است.

```

1 Ali@DESKTOP-GS7PR56 MINGW64 /c/git/AP97982 (fb_A8)
2 $ git push origin fb_A8
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
13 * [new branch] fb_A8 -> fb_A8

```

### ۵.۳ ساخت Pull Request

با مراجعه به سایت [Azure DevOps](#) یک Pull Request جدید با نام `HW8` بسازید به طوری که امکان `merge` کردن شاخه `fb_A8` را بر روی شاخه `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 گزینه `set auto complete` در صفحه‌ی Pull Request مربوطه تعیین کنید که در صورت وجود شرایط `merge` این کار انجام شود. دقت کنید که گزینه `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.