مدرسان: **رامتین خسروی، محمدامین صادقی**
طراحان: **برخی از دستیاران آموزش درس**

مهلت تحویل: چهارشنبه ۶ دی‌ماه ۱۳۹۶، ساعت ۲۳:۵۵

گردابیفای:

یکی از دانشجویان رشته مهندسی کامپیوتر دانشگاه متخلص به گرداب! اخیرا به موسیقی علاقه‌مند شده و در پی آهنگسازی است. از آن جایی که او در دنیای موسیقی تازه وارد است، نگران این است که موسیقی‌هایش با اقبال روبه‌رو نشود. برای همین تصمیم می‌گیرد که خودش یک رسانه موسیقی را پیاده‌سازی کند تا در کنار رقابت با Spotify , Apple Music و ... از این راه اقبال مردم به موسیقی‌هایش را بیشتر کند. حال او می‌خواهد تا هر یک دانشجویان درس برنامه‌سازی پیشرفته این کار را انجام دهند تا او در نهایت بتواند یکی از آن‌ها را به عنوان رسانه‌ی خود برگزیند.

در این تمرین شما یک رسانه‌ی موسیقی را، که از این پس گردابیفای نامیده می‌شود، پیاده‌سازی خواهید کرد.
گردابیفای یک سامانه برای پخش و مدیریت موسیقی است. در فاز یکم، این سامانه به شکل یک برنامه روی کامپیوتر هر نفر اجرا خواهد شد و به او اجازه‌ی پخش و مدیریت موسیقی را خواهد داد.
برای سادگی بیشتر پروژه، شما در فاز یکم شما رابط برنامه‌نویسی[1] این برنامه را هم پیاده‌سازی می‌کنید تا عملکرد برنامه‌ی شما به وسیله‌ی آن آزموده شود.

[1] Application Programming Interface

سناریوی کارکردن با برنامه به شکل زیر است:

هر کاربر برای اینکه بتواند با برنامه کار کند ابتدا باید در سیستم ثبت‌نام کند. پس از ثبت‌نام می‌تواند با وارد کردن نام‌کاربری و گذرواژه‌ی خود وارد برنامه شود. کاربر پس از ورود می‌تواند موسیقی یا آلبوم به [2]ی موسیقی خود اضافه کند. کاربر پس از اضافه کردن موسیقی‌ها و آلبوم‌های موردنظر خود، می‌تواند با ورود به حالت گشت‌وگذار به موسیقی‌ها دسترسی پیدا کند و مشخصات آن‌ها را ببیند تا موسیقی یا موسیقی‌های مورد علاقه خود برای پخش را پیدا کند. برای همین کاربر پس از ورود، در «خانه» (که در واقع نقطه‌ی شروع گشت‌وگذار در میان موسیقی‌های موجود است) قرار می‌گیرد. پس از آن دو انتخاب دارد که می‌تواند وارد کتابخانه موسیقی‌ها یا بخش لیست پخش های خود شود.پس از ورود به یکی از آن‌ها می‌تواند لیست آلبوم‌ها، موسیقی‌ها، هنرمندان یا لیست پخش‌ها[3] را دریافت و وارد یکی از آن‌ها شود. به عنوان مثال کاربر می‌تواند یک آلبوم را انتخاب کند. سپس لیست موسیقی‌های موجود در آن آلبوم را دریافت کند و از میان آن‌ها یک موسیقی را انتخاب کند و وارد آن شود. پس از ورود به یک موسیقی می‌تواند مشخصات آن را از جمله مدت‌زمان، نام هنرمند و ... دریافت کند. سپس کاربر می‌تواند چیزی که وارد آن شده است (انتخاب کرده) را به صف پخش[4] خود اضافه کند. مثلا می‌تواند وارد یک آلبوم شود و این آلبوم را به لیست پخش خود اضافه کند، با این کار تمام موسیقی‌های این آلبوم به صف پخش کاربر اضافه می‌شود. پس از آن کاربر می‌تواند وارد حالت پخش شود تا بتواند پخش موسیقی‌هایی که در صف پخش قرار دارند را مدیریت کند. مثلا می‌تواند پخش موسیقی بعدی، یا قبلی در صف را آغاز کند ، پخش موسیقی فعلی را متوقف کند، موسیقی را جلو یا عقب ببرد و ... .

علاوه بر این کاربر می‌تواند به مدیریت سیستم هم بپردازد. به عنوان مثال می‌تواند به برنامه موسیقی یا آلبوم اضافه کند، آن ها را تایید کند و ...

همچنین با توجه به اینکه این که شیوه‌ی کارکرد این برنامه مشابه Spotify است، توصیه می‌شود که آن را نصب و استفاده کنید تا با شیوه‌ی کارکرد برنامه بیشتر آشنا شوید.

واژه‌نامه

واژه	شرح
خانه	کاربر با وارد شدن به برنامه و رفتن به حالت مرورگر در «خانه» قرار می‌گیرد که در واقع نقطه شروع حالت مرورگر است که شامل دو بخش کتابخانه‌ی موسیقی و لیست پخش‌ها است
بخش کتابخانه‌ی موسیقی	مجموعهای تمامی موسیقی‌های موجود در برنامه است
بخش لیست پخش‌ها	مجموعه‌ی تمامی لیست پخش‌های کاربر است
صف	در هر لحظه اگر پخش موسیقی متوقف نباشد باید نخستین موسیقی صف در حال پخش باشد و در صورتی که پخش این موسیقی تمام شد از صف حذف شود
لیست پخش	یک لیست از موسیقی‌ها است که هر کاربری می‌تواند بسازد
پخش به‌هم‌ریخته	با روشن شدن این حالت ترتیب پخش موسیقی‌ها متفاوت با ترتیب صف و به شکل تصادفی باید انجام شود

در ادامه متدهای رابط که از نظر عملکرد به بخش‌های مرورگر، پخش‌کننده، ورود و ثبت‌نام و مدیریت برنامه تقسیم می‌شوند، شرح داده می‌شود.

---

[2] Library

[3] Playlist

[4] Queue

مرورگر

- در صورتی که متد زیر فراخوانی شود برنامه به حالت مرورگر می‌رود.

```
void enterBrowserMode()
```

که در این حالت کاربر به جستجو و گشت وگذار در کتابخانه موسیقی‌ها می‌پردازد. در این حالت می‌تواند دستورات زیر را وارد کند

• پیمایش:

- با فراخوانی این متد کاربر به خانه می‌رود.

```
void goToHome()
```

- این متد تنها زمانی قابل فراخوانی است که کاربر در خانه باشد. با فراخوانی این متد کاربر به مجموعه‌ی لیست پخش‌های خود می‌رود. در صورتی که کاربر در خانه نباشد باشد باید exception شماره پنج پرتاب شود.

```
void goToPlaylists()
```

- این متد هم تنها زمانی قابل فراخوانی است که کاربر در خانه باشد. با فراخوانی این متد کاربر به کتاب‌خانه‌ی موسیقی‌های برنامه می‌رود. در صورتی که کاربر در خانه نباشد باشد باید exception شماره پنج پرتاب شود.

```
void goToLibrary()
```

- با فراخوانی این متد کاربر به آلبوم موسیقی‌ای که در آن قرار دارد می‌رود. در صورتی که کاربر در یک موسیقی نباشد باید exception شماره پنج پرتاب شود.

```
void goToAlbum()
```

- با فراخوانی این متد کاربر به هنرمند موسیقی یا آلبوم فعلی می‌رود. در صورتی که کاربر در یک موسیقی یا آلبوم نباشد باید exception شماره پنج پرتاب شود.

```
void goToArtist()
```

• جستجو:

- این متد باید یک لیست از نام آلبوم‌ها، موسیقی‌ها، هنرمندها و لیست پخش‌های موجود در کتابخانه که در نام آن‌ها، آرگومان متد وجود دارد را برگرداند. توجه کنید که این جستجو بر اساس نام‌کامل انجام می‌شود.

```
 List search(string <completeMusicName||completeAlbumName||completeArtistName||
completePlaylistName>)
```

- **انتخاب:**
  - با استفاده از این متد کاربر باید به موسیقی، آلبوم، هنرمند یا لیست پخشی که نام آن آرگومان داده‌شده به متد باشد برود. در صورتی که این نام در لیست موجود نباشد باید exception شماره‌ی ۴ پرتاب شود.

```
void select(string <musicName||albumName>)
```

- **لیست:**
  - این متد با توجه به چیزی که کاربر در آن قرار دارد خروجی‌های متفاوتی خواهد داشت که در زیر به آن‌ها اشاره می‌شود. اگر کاربر در:
    - خانه باشد باید دو عبارت playlists و library برگردانده شود.
    - کتابخانه باشد باید یک لیست از نام تمام هنرمندان موجود برگردانده شود.
    - یک آلبوم باشد باید یک لیست از نام موسیقی‌های آن آلبوم برگردانده شود.
    - یک هنرمند باشد باید یک لیست از نام آلبوم‌های آن هنرمند برگردانده شود.
    - یک لیست پخش باشد باید یک لیست از نام موسیقی‌های آن لیست پخش برگردانده شود.

```
List list()
```

- این متد باید یک لیست از نام موسیقی‌ها را برگرداند. با فراخوانی این متد، باید یک لیست شامل نام موسیقی‌های آلبوم، هنرمند یا لیست پخشی که کاربر در آن قرار دارد بازگردانده شود. اگر کاربر در کتابخانه‌ی موسیقی باشد باید یک لیست از نام تمام موسیقی‌های موجود در سیستم برگردانده شود. اگر چیزی که کاربر در آن قرار دارد یک موسیقی باشد، باید exception شماره پنج پرتاب شود.

```
List listMusics()
```

- این متد باید یک لیست از نام آلبوم‌ها را برگرداند. با فراخوانی این متد، باید یک لیست شامل نام آلبوم‌های هنرمند، یا موجود در لیست پخشی که کاربر در آن قرار دارد بازگردانده شود. اگر کاربر در کتابخانه‌ی موسیقی‌ها باشد باید یک لیست از نام تمام آلبوم‌های موجود در سیستم برگردانده شود. اگر چیزی که کاربر در آن قرار دارد یک موسیقی باشد، باید exception شماره پنج پرتاب شود.
-

```
List listAlbums()
```

- این متد باید یک لیست از نام لیست پخش‌ها را برگرداند. این متد تنها زمانی قابل فراخوانی است که کاربر در بخش لیست پخش‌ها باشد. اگر کاربر در این بخش نباشد، باید exception شماره پنج پرتاب شود.

```
List listPlaylists()
```

- این متد باید یک لیست از نام هنرمندهای موجود در کتاب‌خانه‌ی موسیقی را برگرداند. اگر کاربر در کتابخانه‌ی موسیقی نباشد باید exception شماره پنج پرتاب شود.

```
List listArtists()
```

- **پخش:**

• این متد باید چیزی که کاربر در آن قرار دارد را پخش کند که می‌تواند یک قطعه‌ی موسیقی، همه‌ی موسیقی‌های یک آلبوم، همه‌ی موسیقی‌های یک لیست پخش و یا همه‌ی موسیقی‌های یک هنرمند را به ابتدای صف پخش اضافه و سپس پخش آن را آغاز کند.

```
void play()
```

- **مدیریت لیست‌های پخش:**

- این متد باید همه‌ی موسیقی‌های یک آلبوم یا موسیقی‌ای که کاربر در آن قرار دارد را به آلبوم، یا لیست پخشی که نام آن برابر با آرگومان مشخص شده است اضافه کند.
- اگر مقصد یک لیست پخش که مالک آن خود کاربر است باشد، می‌توان یک موسیقی، یا موسیقی‌های یک آلبوم را به آن لیست پخش اضافه کرد.
- اگر مقصد لیست پخشی بود که مالک آن خود کاربر نیست، باید exception شماره سه پرتاب شود.
- اگر مقصد آلبومی باشد که کاربر فعلی، هنرمند آن نیست یا هنرمند موسیقی فعلی کاربر فعلی نباشد باید exception شماره سه پرتاب شود.
- اگر مقصد موجود نباشد باید exception شماره چهار پرتاب شود.
- اگر آخرین چیزی که انتخاب شده یک هنرمند، یا یک لیست پخش باشد باید exception شماره پنج پرتاب شود.

```
void addTo(string <completePlaylistName||completeAlbumName||"queue">)
```

• با فراخوانی این متد کاربر باید به مقصد مشخص شده که می‌تواند «queue» یا نام یک لیست پخش باشد، برود. در صورتی که لیست‌پخش مقصد وجود نداشت باید exception شماره چهار پرتاب شود.

```
void goTo(string <playlistName>)
```

• این متد باید یک لیست پخش با نام مشخص شده بسازد. در صورتی که لیست پخش‌ای با نام مشخص شده وجود داشته‌باشد باید exception شماره نه پرتاب شود.

```
void createPlaylist(string <playlistName>)
```

• این متد باید لیست پخش مشخص‌شده را خالی کند.

```
void clearPlaylist(string [playlistName])
```

• این متد باید موسیقی مشخص شده را از لیست پخش یا آلبومی که کاربر در آن قرار دارد حذف کند. در صورتی که کاربر فعلی هنرمند آلبوم یا مالک لیست پخش نباشد باید exception شماره‌ی سه پرتاب شود.

```
void remove(string [completeMusicName])
```

• این متد باید موسیقی مشخص شده را به مکان جدید در لیست پخشی که کاربر در آن قرار دارد منتقل کند. در صورتی که موسیقی مشخص شده در لیست پخش موجود نباشد باید exception شماره‌ی چهار پرتاب شود. اگر مکان جدید کوچکتر از یک یا بزرگتر از تعداد اعضای لیست پخش باشد باید exception شماره‌ی صفر پرتاب شود.

```
void orderItem(string <completeMusicName>, int <newOrder>)
```

- این متد باید از سوی کاربر به چیزی که کاربر در آن قرار دارد، امتیاز دهد. اگر چیزی که کاربر در آن قرار دارد یک موسیقی یا آلبوم نباشد باید exception شماره‌ی پنج پرتاب شود. در صورتی که امتیاز در بازه‌ی مورد نظر نباشد باید exception شماره‌ی صفر پرتاب شود.

```
void rate(int <score>)
```

- **دریافت مشخصات:**
  - این متد باید نام هنرمند آلبوم یا موسیقی‌ای که کاربر در آن قرار دارد را برگرداند. اگر کاربر در یک آلبوم یا موسیقی نباشد باید exception شماره‌ی پنج پرتاب شود.

```
string getArtist()
```

- این متد باید نام هنرمند آلبوم یا موسیقی‌ای که کاربر در آن قرار دارد را برگرداند. اگر کاربر در یک آلبوم یا موسیقی نباشد باید exception شماره‌ی پنج پرتاب شود.

```
string getAlbum()
```

- با فراخوانی این متد اگر کاربر در یک موسیقی باشد باید مدت زمان آن و اگر در یک آلبوم باشد باید مجموع مدت زمان موسیقی‌های این آلبوم بازگردانده شود. اگر کاربر در یک موسیقی یا آلبوم نباشد باید exception شماره‌ی پنج پرتاب شود.

```
int getLength()
```

- این متد باید نام چیزی که کاربر در آن قرار دارد را برگرداند.

```
string getName()
```

- این متد نام کامل چیزی که کاربر در آن قرار دارد را برمی‌گرداند.
  - نام کامل آلبوم به شکل <artistName>-<albumName>
  - نام کامل موسیقی به شکل <albumCompleteName>-<musicName>
  - نام کامل لیست پخش به شکل <username>-<playlistName>
  - نام کامل برای موارد دیگر همان نام آن‌هاست.

```
string getCompleteName()
```

- این متد باید امتیاز داده‌شده توسط کاربر به هنرمند، آلبوم یا موسیقی‌ای که در آن قرار دارد را برگرداند. اگر کاربر در یک هنرمند، آلبوم یا موسیقی نباشد باید exception شماره‌ی پنج پرتاب شود.

```
int getRate()
```

- این متد باید میانگین امتیاز داده‌شده توسط همه‌ی کاربران عادی یا هنرمند سیستم به هنرمند، آلبوم یا موسیقی‌ای که در آن قرار دارد را برگرداند. اگر کاربر در یک هنرمند، آلبوم یا موسیقی نباشد باید exception شماره‌ی پنج پرتاب شود.

```
int getUserRate()
```

- این متد باید میانگین امتیاز داده‌شده توسط همه‌ی کاربران منتقد سیستم به هنرمند، آلبوم یا موسیقی‌ای که در آن قرار دارد را برگرداند. اگر کاربر در یک هنرمند، آلبوم یا موسیقی نباشد باید exception شماره‌ی پنج پرتاب شود.

```
int getCriticRate()
```

در صورتی که متدهای حالت مرورگر زمانی که کاربر در حالت مرورگر نیست فراخوانی شوند باید exception شماره پنج پرتاب شود.

پخش‌کننده

- در صورتی که متد زیر فراخوانی شود برنامه به حالت پخش‌کننده می‌رود. که در این حالت کاربر کنترل پخش‌کننده‌ی موسیقی را در اختیار می‌گیرد.

```
Void enterPlayerMode()
```

- این متد باید پخش آهنگ بعدی در صف را آغاز کند.

```
void next()
```

- این متد باید پخش آهنگ قبلی در صف را آغاز کند.

```
void previous()
```

- این متد باید پخش آهنگ اول صف را آغاز کند/ ادامه دهد.

```
void play()
```

- این متد باید پخش آهنگ در حال پخش را متوقف کند.

```
void pause()
```

- این متد باید دقیقه و ثانیه مشخص شده‌ی آهنگ در حال پخش برود.

```
void seek(int [minute], int [second])
```

- این متد باید حالت پخش به‌هم‌ریخته صف را روشن/خاموش(اگر خاموش است روشن و اگر روشن است خاموش) کند به این صورت که ترتیب پخش موسیقی‌ها به شکل تصادفی و متفاوت با ترتیب آن‌ها در صف باشد.

```
void shuffle()
```

- این متد باید حالت تکرار پخش موسیقی در حال‌پخش را روشن/خاموش کند.

```
void repeatOne()
```

- این متد باید حالت تکرار پخش موسیقی‌های موجود در صف را روشن/خاموش کند.

```
void repeat()
```

در صورتی که متدهای حالت پخش‌کننده (به جز متد play()) زمانی که کاربر در حالت پخش‌کننده نیست فراخوانی شوند باید exception شماره پنج پرتاب شود.

ورود و ثبت‌نام

- این متد باید کاربر را در سیستم ثبت‌نام کند[5]. داده‌های مورد نیاز برای ثبت‌نام شدن در سیستم در قالب ساختار UserData به این تابع داده می‌شود. در صورتی که نوع کاربر مشخص شود، پس از ثبت‌نام شدن تا زمانی که کاربر تایید نشود، به عنوان کاربر ساده در سیستم در نظر گرفته می‌شود. درصورتی که نام‌کاربری از قبل در سیستم موجود باشد باید exception شماره‌ی نه پرتاب شود.

```
void signup(UserData <userData>, string <password>)
```

ساختار UserData شامل درایه‌های زیر است:

[5] Signup

- نام
- نام‌خانوادگی
- نام‌کاربری
- نوع کاربر(در صورتی که نوع کاربر مشخص نشود، به عنوان کاربر ساده در نظر گرفته خواهد شد)

- این متد باید کاربر را وارد سیستم کند. در صورتی که نام‌کاربری یا گذرواژه درست نباشد باید exception شماره‌ی یک پرتاب شود.

```
void login(string <username>, string <password>)
```

مدیریت برنامه

- با این متد کاربر فعلی می‌تواند یک موسیقی که فایل آن در محل مشخص شده وجود دارد، در سیستم ایجاد کند که هنرمند آن کاربر فعلی است. این موسیقی تا زمانی که توسط مدیر سیستم تایید نشود، برای کاربران دیگر قابل مشاهده نیست. در صورتی که کاربری که این متد را فراخوانی می‌کند هنرمند نباشد باید exception شماره‌ی سه پرتاب شود.

```
void addMusic(string <musicName>, string <musicFilePath>)
```

- با این متد کاربر فعلی می‌تواند یک آلبوم ایجاد کند که هنرمند آن خودش است. این آلبوم تا زمانی که توسط مدیر سیستم تایید نشود، برای کاربران دیگر قابل مشاهده نیست. در صورتی که کاربری که این متد را فراخوانی می‌کند هنرمند نباشد باید exception شماره‌ی سه پرتاب شود.

```
void addAlbum(string <artistName>, string <albumName>)
```

- با این متد کاربر فعلی می‌تواند آلبوم، هنرمند یا موسیقی‌ای که در آن قرار دارد را تایید کند. در صورتی که کاربر فعلی مدیر نباشد باید exeption شماره سه پرتاب شود. اگر کاربر در یک آلبوم، موسیقی یا هنرمند نباشد باید exception شماره پنج پرتاب شود.

```
void validate()
```

- با این متد کاربر مشخصات یک کاربر را در قالب ساختار UserData برمی‌گرداند. در صورتی که چنین کاربری وجود نداشته باشد باید exception شماره پنج پرتاب شود.

```
UserData getUserDetails(string <username>)
```

انواع Exception

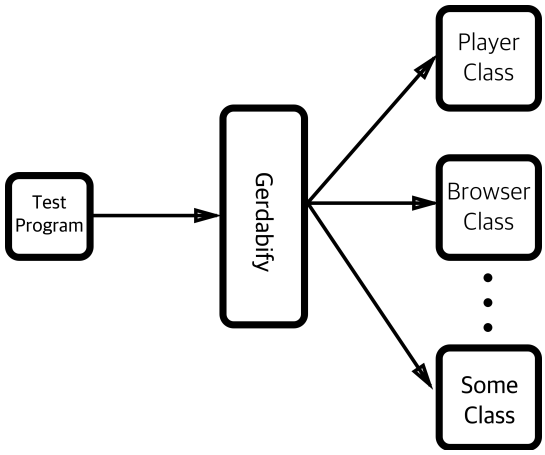
نام Exception	شماره Exception
ItemDoesNotExist	۴
NotAuthorized	۳
MethodNotAvaliable	۵
InvalidArguments	۰
InvalidCredentials	۱
AlreadyExists	۹

انواع کاربر در ساختار UserData

نوع کاربر	مقدار
هنرمند	artist
کاربرساده	-
منتقد	editor
مدیرسیستم	admin

نکات پایانی:

- برای انجام فازهای بعدی نیاز به انجام این فاز خواهید داشت.
- الگوی طراحی شی‌گرا در این تمرین بسیار مهم است، برای همین باید تا جای ممکن از این الگو استفاده کنید و از الگوی طراحی رویه‌ای[6] استفاده نکنید. پیروی نکردن از این موضوع موجب از دست دادن نمره خواهد شد. (تشخیص کیفیت این موضوع بر عهده دستیاران آموزشی است)
- برای پخش و رمزگشایی [7] موسیقی از چارچوبQT [8] استفاده کنید. آشنایی و استفاده از چارچوب QT بخشی از این پروژه و بر دوش خود شماست.
- رابط برنامه‌نویسی شما باید از کلاس Gerdabify (که در اختیار شما قرار خواهد گرفت) ارث ببرد. البته توجه کنید که این کلاس تنها یک رابط بین برنامه‌ی شما و برنامه‌های دیگر است (مثلا کدی که برای تست برنامه شما استفاده می‌شود) و منطق‌های اجرایی برنامه‌ی شما نباید در پیاده‌سازی آن قرار داده شود (یا حداقل منطق ممکن در آن پیاده‌سازی شود). همچنین این کلاس واسط ممکن است از چندین کلاس مختلف برای اجرای توابع خود استفاده کند. برای مثال به طور مشخص توابع مربوط به پخش موسیقی و توابع مربوط به گشت‌وگذار در موسیقی‌ها در یک کلاس پیاده‌سازی نمی‌شوند. پیشنهاد می‌کنیم ابتدا مستقل از کلاس گردابیفای، برنامه‌ی خود را توسعه دهید و سپس سعی کنید تا امکانات برنامه‌ی خود را (به کمک کلاس‌هایی که طراحی کرده‌اید)، از طریق کلاس گردابیفای در دسترس ما قرار دهید. در شکل زیر، Test Program شامل main ما خواهد بود که به کمک کلاس گردابیفای (facade) برنامه‌ی شما را تست خواهد کرد.



نحوه‌ی تحویل

تمامی فایل‌های برنامه‌ی خود را در آرشیوی با نام A7-SID.zip در صفحه‌ی CECM درس بارگذاری کنید. برای مثال، اگر شماره‌ی دانشجویی شما ۸۱۰۱۱۲۳۴۵ است، نام فایل شما باید A7-810112345.zip باشد.لطفاً از روش‌های دیگر فشرده‌سازی مانند rar یا tar.gz استفاده نکنید.

- برنامه‌ی شما باید در سیستم‌عامل لینوکس و با مترجم g++ با استاندارد C++98 ترجمه و در زمان معقول برای ورودی‌های آزمون اجرا شود.
- به فرمت و نام فایل‌های خود دقت کنید.
- از صحت فرمت ورودی‌ها و خروجی‌های برنامه‌ی خود مطمئن شوید.
- هدف این تمرین یادگیری شماست. لطفاً تمرین را خودتان انجام دهید. در صورت کشف تقلب مطابق قوانین درس با آن برخورد خواهد شد.

---

[6] Procedural

[7] Decode

[8] Framework