دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

پروژه درس مبانی برنامه‌سازی پایتون

گروه ۷ – پاییز ۱۴۰۳

عنوان پروژه: سامانه آموزشی (LMS)

استاد: علی ابریشمی

طراحان پروژه:

مانی ابراهیمی
نیما پشابادی
محمدامین حیدری
سید امیرمحمد جزایری

تاریخ تحویل: ۱۱ بهمن ۱۴۰۳

فهرست مطالب

معرفی پروژه

۱.۱ ◆ شرح پروژه

در این پروژه قرار است شما یک LMS یا سامانه مدیریت یادگیری بسازید، یعنی سامانه‌ای مشابه با CW شریف یا Courses امیرکبیر که خوب است بدانید برپایه‌ی Moodle[1] ساخته شده‌اند. در این راستا، شما با رویکرد شی‌گرا مدل‌های سامانه را تعریف خواهید کرد و با ایجاد ارتباط منطقی میان آن‌ها یک سامانه‌ی قابل اجرا خواهید داشت. همچنین در بخش‌هایی از این پروژه از شما خواسته خواهد شد تا با استفاده از Pandas و Numpy داده‌های کلاسی را پردازش کرده و یک خروجی خوانا برای کاربران سامانه ایجاد کنید.

۱.۲ ◆ انتظارات و اهداف

انتظار می‌رود در انتهای این پروژه:

- رویکرد شی‌گرا را به خوبی آموخته و بتوانید با آن یک پروژه‌ی قابل اجرا بسازید.
- بتوانید با استفاده از Pandas و Numpy داده‌ها را پردازش کرده و یک خروجی خوانا برای کاربران سامانه ایجاد کنید.
- تا حد بسیار مبتدی با مفاهیم ذخیره‌سازی داده‌ها و پردازش آن‌ها آشنا شوید.

و همچنین در صورت انجام بخش اختیاری هم از شما انتظار می‌رود:

- بتوانید با Git تغییرات یک پروژه را مدیریت کنید.
- با گیت‌هاب آشنا شوید و بتوانید با آن یک پروژه را مدیریت کنید.
- با مفهوم GUI آشنا شوید و بتوانید با استفاده از Tkinter یا PyQt یک پنجره‌ی GUI بسازید.
- با اهمیت آزمایش آشنا شده و بتوانید برای یک پروژه Unit Test بنویسید.

۱.۳ ◆ قوانین مربوط به پروژه

- در این پروژه شما مجاز به استفاده از مدل‌های بزرگ زبانی[2] مانند ChatGPT یا Claude هستید؛ به شرط اینکه هر دو عضو گروه تسلط کامل بر کد پروژه داشته باشند و بتوانند حین تحویل آن را به خوبی توضیح دهند.
- مستند پروژه را یک بار تا انتها بخوانید، تعدادی از موارد امتیازی را (در صورتی که تمایل به انجام آن‌ها دارید) می‌بایست از همان ابتدای شروع پروژه انجام دهید و در نظر داشته باشید.
- تمام فایل‌های پروژه‌تان را در یک فایل فشرده با فرمت `.zip` روی کوئرا بارگذاری کنید.
- نام فایلی که بارگذاری می‌شود باید به فرمت `FOP_PROJ_{STDID1}_{STDID2}` باشد که STDID1 شماره دانشجویی عضو اول گروه و STDID2 شماره دانشجویی عضو دوم گروه است. برای مثال اگر یک گروه داشته باشیم که شماره دانشجویی اعضایش 403108123 و 403108987 باشد، باید فایلی به نام `FOP_PROJ_403108123_403108987.zip` آپلود کنند.
  آپلود هرگونه فایل با نامگذاری خارج از این چارچوب موجب کسر نمره خواهد شد.
- هر دو عضو گروه می‌بایست فایل مربوطه را بارگذاری نمایند. مسئولیت عدم بارگذاری یا بارگذاری فایل‌های متفاوت توسط اعضای یک گروه متوجه خود ایشان است.
- از آنجا که تحویل پروژه اجباری است، مطمئن شوید هر دو عضو گروه کاملاً بر پروژه مسلط باشند. همچنین هر دو عضو موظف به فعالیت هستند و در صورتی که یکی از اعضا فعالیت نکند، تمامی اعضا به عنوان یک گروه دچار کسر نمره خواهد شد.
- پروژه تاخیر نخواهد داشت و زمان پایان اعلام شده، نهایی (هارد ددلاین) خواهد بود.
- اطمینان حاصل شده است که پروژه‌ی شما با تمامی مطالبی که در کلاس آموخته‌اید قابل انجام باشد. پس پیش از انجام پروژه، همه‌ی مطالبی که در کلاس آموخته‌اید را به خوبی مرور بفرمایید.

[1] https://moodle.org
[2] LLM

بخش اجباری (۲۰۰۰ نمره)

توجه!

جمع نمرات این بخش ۲۰۰۰ نمره است که معادل ۲ نمره از کل درس می‌باشد. انجام موارد ذکر شده در این بخش اجباری است.

۲.۱ ◆ پیاده‌سازی ساختارهای داده‌ای برای مدل‌ها با استفاده از لغت‌نامه (۲۰۰ نمره)

در این بخش شما موظفید تا ساختارهای داده‌ای مورد نیاز برای مدل‌ها را با استفاده از لغت‌نامه[۱] پیاده‌سازی کنید. این بخش را دقیقا می‌توانید مانند سوال چهارم تمرین ۳ یا سوال آخر تمرین ۴ پیاده کنید؛ به این صورت که مشخصات هر مدل را به صورت یک کلید[۲] قرار داده و مقداردهی آن را از طریق مقدار[۳] انجام دهید. همچنین می‌توانید به منظور ذخیره‌سازی تمام این لغت‌نامه‌ها از یک لیست[۴] استفاده کنید.

۲.۱.۱ ■ مدل دانشجو (Student)

مشخصات یک مدل دانشجو به شکل زیر است:

کلید	نوع داده
`id`	`int`
`name`	`str`
`email`	`str`
`password`	`str`
`phone`	`str`

۲.۱.۲ ■ مدل استاد (Professor)

مشخصات یک مدل استاد به شکل زیر است:

کلید	نوع داده
`id`	`int`
`name`	`str`
`email`	`str`
`password`	`str`
`phone`	`str`

۲.۱.۳ ■ مدل درس (Course)

مدل هر درس به صورت زیر خواهد بود:

۲.۱.۴ ■ شکل نهایی کد شما

بخشی از کدتان که در آن داده‌ساختارها را ذخیره کرده‌اید، به شکل زیر خواهد بود:

---

[۱] dictionary
[۲] key
[۳] value
[۴] list

کلید	نوع داده
`id`	`int`
`name`	`str`
`description`	`str`
`professor`	`int`
`students`	`list[int]`

Code

```
STUDENTS = []
TEACHERS = []
COURSES = []
```

۲.۲ ◆ ایجاد فرآیند ثبت‌نام و ورود (۲۰۰ نمره)

شما باید در ابتدا فرآیند ثبت نام و ورود کاربران (هم دانشجو هم استاد) را اضافه نمایید. کاربر ابتدا وارد بخش ثبت‌نام شده و نقش خود (استاد یا دانشجو) را به همراه اطلاعات کاربری‌اش ذخیره می‌کند. توجه کنید که شماره دانشجویی یا کد استاد باید حتماً یکتا باشند. همچنین کد هیچ استادی با شماره دانشجویی هیچ دانشجویی برابر نباشد. پیشنهاد می‌شود دو سیستم متفاوت برای کد معتبر استاد و شماره معتبر دانشجویی در نظر بگیرید (مثلا تمام اساتید کد ۴ رقمی داشته باشند و تمام دانشجویان کد ۹ رقمی). در نهایت نیز باید برنامه بخشی داشته باشد که کاربر با وارد کردن شناسه کاربری (کد استاد یا شماره دانشجویی) همراه با رمز عبور خود وارد سامانه شود.

۲.۳ ◆ پیاده‌سازی فرآیند ایجاد کلاس و ثبت‌نام دانشجو (۲۰۰ نمره)

پیاده سازی فرآیند ایجاد کلاس و ثبت نام دانشجو در این بخش از سیستم، شما وظیفه دارید که کلاس‌ها را ایجاد و مدیریت کنید. هر کلاس درس اطلاعاتی مانند نام کلاس، مدرس، ظرفیت، و زمان شروع را ذخیره می‌کند و به سایر ماژول‌های سیستم مانند ماژول ثبت‌نام دانشجویان و مدیریت محتوا متصل می‌شود. همچنین هر دانشجو می‌تواند با اضافه کردن آیدی کلاس، به آن اضافه شود.

ویژگی‌ها

۱. ایجاد کلاس روم جدید:

- ایجاد آیدی منحصر به فرد برای کلاس
- تعریف نام کلاس.
- اختصاص مدرس.
- تعیین ظرفیت کلاس.
- تنظیم زمان‌های کلاس.

۲. ویرایش اطلاعات کلاس درس:

- امکان تغییر نام، مدرس، ظرفیت، یا زمان‌های کلاس.

۳. حذف کلاس درس:

- حذف کلاس درس از سیستم، با بررسی وابستگی‌ها (مانند دانشجویان ثبت‌نام‌شده).

۴. نمایش لیست کلاس درس‌ها:

- بازیابی و نمایش همه کلاس درس‌ها برای مدیریت بهتر.

۲.۴ ◆ پیاده‌سازی امکان تعیین بارم‌بندی درس توسط استاد (۱۵۰ نمره)

شما باید بتوانید شرایطی را مهیا کنید که در آن استاد بتواند بارم‌بندی درس که شامل ریزنمرات درس به صورت یک نمره مشخص از یک تا بیست و بخشی از درس که نمره به آن تعلق می‌گیرد را تعیین کند. همچنین این اطلاعات باید توسط دانشجو قابل مشاهده باشد. شایان ذکر است که استاد در زمان‌های دیگر به غیر از زمانی که بارم‌بندی اولیه را تنظیم می‌کند نیز می‌تواند بارم‌بندی را تغییر دهد.

نکته: جمع نمرات باید ۲۰ باشد.

۲.۵ ◆ پیاده‌سازی امکان ثبت نمره دانشجو (۱۵۰ نمره)

در این قسمت استاد باید توانایی ثبت درس را برای دانشجو داشته باشد. همچنین استاد باید توانایی پنهان کردن و نمایش دادن نمره را نیز هم داشته باشد. شایان ذکر است که استاد می‌تواند نمره دانشجو را عوض کند. به همچنین زمانی که استاد قابلیت دیدن را برای دانشجو انتخاب کرده است، دانشجو باید توانایی دیدن نمره را داشته باشد.

۲.۶ ◆ ذخیره‌ی اطلاعات کاربران (دانشجویان و اساتید) و کلاس‌ها روی فایل (۳۰۰ نمره)

استاد باید بتواند به طور مشخص اطلاعات مربوط به دانشجویان را ذخیره کند. در این بخش این قابلیت را دانشجویان نخواهند داشت (دانشجویان می‌توانند اطلاعات مربوط به درس مشخص و استاد همان درس را ذخیره کنند، اما دسترسی به لیست دانشجویان دیگر نخواهند داشت). همچنین استاد توانایی ذخیره کردن تمامی اطلاعات کلاس‌ها را نیز باید داشته باشد و آن‌ها را بر روی یک فایل ذخیره کند. ذخیره آن روی فایل به این معناست که پس از خروج از برنامه، اطلاعات نباید پاک شوند و در ابتدا از روی فایل خوانده شوند. توجه شود که نمرات دانشجویان هر درس نیز باید روی فایل ذخیره شود تا مراحل بعدی ممکن شوند.

۲.۷ ◆ ساخت شیت لیست کلاسی با Pandas (۲۵۰ نمره)

می‌دانیم که هر دانشجو برای هر درس ۴ نمره دارد کوییز اول میانترم کوییز دوم و پایانترم.برای هر کلاس استاد باید با استفاده از دیتا فریم pandas یک لیست کلاسی تهیه کند که نام هر فرد به اضافه نمرات او و نمره پایانی او ثبت شود. در صورت نبود یک نمره جای ان صفر قرار می‌گیرد.

۲.۸ ◆ استخراج شیت کلاسی با فرمت CSV و XLSX (۲۰۰ نمره)

هر استاد باید بتواند لیست کلاسی هر درس خود را به فرمت xlsx و یاcsv استخراج کند .اینکار با کتابخانه‌های پایتون امکان‌پذیر است.

۲.۹ ◆ محاسبه نمره و اعمال نمودار روی نمره (۱۵۰ نمره)

هر استاد با دسترسی که به فایل کلاسی درس خود دارد میانگین درس خود را رویت کند. همچنین چنانچه این میانگین کمتر از عدد مرسوم ۱۶ بود باید بتواند با استفاده از یک شیفت یونیفرم میانگین را به ۱۶ برساند و نمرات همه به مقدار یکسان اضافه شود.

۲.۱۰ ◆ رسم نمودار پیشرفت دانشجو (۲۰۰ نمره)

هر استاد باید بتواند با در اختیار داشتن نام دانش‌اموز و کلاسی که دانشجو دارد نمرات او در طول ترم را در یک نمودار ببیند. برای رسم نمودار از کتابخانه matplotlib و یا seaborn استفاده کنید. نمودار باید به صورت خطی باشد و نمرات باید به ترتیب زمانی رسم شوند.

بخش اختیاری (۱۰۰۰ نمره)

توجه!

جمع نمرات این بخش ۱۰۰۰ نمره است که معادل ۱ نمره از کل درس می‌باشد. انجام موارد ذکر شده در این بخش اجباری نیست اما نمره‌ی امتیازی بر روی کل درس دارد.

۳.۱ ◆ استفاده از رویکرد شی‌گرا (۱۵۰ نمره)

در این بخش شما باید مدل‌هایتان را به جای استفاده از لغت‌نامه، با استفاده از رویکرد شی‌گرا ایجاد کنید. سه کلاس `Student`، `Professor` و `Course` را ایجاد کرده و توابع مورد نیاز برای این کلاس‌ها را پیاده‌سازی کنید.

۳.۱.۱ ■ ساختار کلاس Student

ساختار کد شما برای کلاس Student می‌تواند به شکل زیر باشد:

Code

```
class Student:
    def __init__(self, id, name, email, password, phone):
        # TODO: Implement the Student class
        pass

    # TODO: Implement the other methods
```

۳.۱.۲ ■ ساختار کلاس Professor

ساختار کد شما برای کلاس Professor می‌تواند به شکل زیر باشد:

Code

```
class Professor:
    def __init__(self, id, name, email, password, phone):
        # TODO: Implement the Professor class
        pass

    # TODO: Implement the other methods
```

۳.۱.۳ ■ ساختار کلاس Course

ساختار کد شما برای کلاس Course می‌تواند به شکل زیر باشد:

Code

```
class Course:
    def __init__(self, id, name, description, professor):
        # TODO: Implement the Course class
        pass

    # TODO: Implement the other methods
```

توجه شود که در صورت پیاده‌سازی صحیح کلاس‌ها با ساختار شی‌گرا، نیازی به پیاده‌سازی آن‌ها با لغت‌نامه نیست و نمره‌ی آن بخش نیز به صورت کامل به شما تعلق خواهد گرفت.

۳.۲ ◆ پیاده‌سازی امکان عدم نیاز به ورود مجدد پس از ورود دوباره به برنامه (۱۰۰ نمره)

هرکاربر (اعم از استاد و دانشجو) باید بتواند پس از ورود در صورتی که بخواهد، گزینه‌ای را فعال کند تا دفعه‌ی بعدی که وارد برنامه شد نیاز به ورود نداشته باشد. همچنین توجه شود که این گزینه با خروج کاربر از حساب خودش، غیرفعال خواهد شد و کاربر باید دفعه‌ی بعد حتما اطلاعات کاربری خود را مجددا وارد کند/

۳.۳ ◆ ایجاد تست (Unit Test) (۵۰ نمره)

برای این بخش، شما باید تعداد تست واحد[۱] مناسب برای چند تابع پیاده کنید. برای یادگیری بیشتر در مورد تست‌های واحد به ضمیمه ۲ مراجعه کنید. همچنین شما باید از ماژول `unittest` استفاده کنید.
برای کسب نمره‌ی کامل این بخش، شما باید برای حداقل ۳ تابع از توابع کدتان تست واحد بنویسید.

۳.۴ ◆ استفاده از Git (۲۰۰ نمره)

برای این بخش، شما از ابتدا باید برای پروژه‌تان یک مخزن[۲] Git ایجاد نمایید. برای این کار ابتدا وارد پوشه‌ای می‌شوید که در آن بناست پروژه را ذخیره کنید و سپس دستور زیر را در ترمینال یا CMD یا PowerShell اجرا کنید:

Terminal

```
C:/Users/username/Desktop/project> git init
```

سپس هربار تغییری در کد می‌دهید می‌توانید به صورت زیر از تغییراتتان ذخیره کنید:

Terminal

```
C:/Users/username/Desktop/project> git add .
C:/Users/username/Desktop/project> git commit -m "commit message"
```

که در آن `commit message` یک متن است که شما می‌توانید برای توضیح تغییراتتان استفاده کنید.
برای مطالعه‌ی بیشتر در مورد Git به ضمیمه ۲ مراجعه کنید.
همچنین برای کسب نمره کامل از این بخش، مخزن Git شما باید دارای حداقل ۵ تغییر[۳] باشد. می‌توانید برای دیدن سوابق تغییراتتان از دستور زیر استفاده کنید:

---

[۱] Unit Test
[۲] Repository
[۳] Commit

```
Terminal

C:/Users/username/Desktop/project> git log
```

۳.۵ ◆ ایجاد رابط کاربری گرافیکی (۴۰۰ نمره)

در ابتدا کاربر باید انتخاب کند که می‌خواهد به عنوان دانشجو یا به عنوان استاد وارد پنل شود.سپس با انتخاب هر کدام پنجره متفاوتی برای او باز می‌شود.

۳.۵.۱ ■ ویژگی‌ها پنجره‌های اصلی

۱. پنجره دانشجو:

- لیست کلاس‌های ثبت‌نام شده
- امکان اضافه کردن یک کلاس جدید با وارد کردن آیدی آن کلاس
- مشاهده جزئیات کلاس(مثل نام استاد،ظرفیت،زمان برگزاری)

۲. پنجره استاد:

- ایجاد کلاس
- ویرایش اطلاعات کلاس
- حذف کلاس
- نمایش لیست کلاس‌های ایجاد شده توسط استاد

۳.۶ ◆ انتشار روی GitHub (۱۰۰ نمره)

در این بخش، شما وظیفه دارید که پروژه خود را بر روی اکانت گیتهاب خود منتشر کنید. برای اینکار،ابتدا نیاز است به وبسایت GitHub بروید و در آنجا اکانت خود را بسازید. پس از ساختن اکانت،وارد آن شوید و مراحل زیر را انجام دهید:

۳.۶.۱ ■ روش اول: استفاده از محیط ترمینال

اگر بخش Git را انجام می‌دهید، توصیه می‌کنیم از دستورات گفته شده در این روش استفاده نمایید.

۱. ساختن مخزن

(آ) روی دکمه New Repository کلیک کنید.

(ب) نامی برای مخزن انتخاب کنید.

(ج) مخزن را عمومی یا خصوصی تنظیم کنید.

(د) روی Create Repository کلیک کنید.

۲. اتصال مخزن محلی به **GitHub**:

برای اتصال مخزن Git محلی به مخزن ایجادشده در GitHub، دستور زیر را اجرا کنید:

```
Terminal

git remote add origin https://github.com/username/repository.git
```

که `repository` نام مخزن گیتهاب شما و `username` نام کاربری شما در گیتهاب است.

۳. ارسال تغییرات به **GitHub**:

برای ارسال تغییرات ثبت‌شده به GitHub، دستورات زیر را اجرا کنید

Terminal

```
git push -u origin main
```

۴. کلون کردن مخزن از **GitHub**:

برای دریافت یک مخزن موجود از GitHub روی سیستم خود، دستور زیر را اجرا کنید:

Terminal

```
git clone https://github.com/username/repository.git
```

۵. مدیریت تغییرات در **GitHub**:

برای مشاهده سوابق تغییرات، دستور زیر را اجرا کنید:

Terminal

```
git log
```

۶. برای همگام‌سازی تغییرات جدید از مخزن **GitHub**، از دستور زیر استفاده کنید::

Terminal

```
git pull origin main
```

۷. ایجاد شاخه جدید برای ویژگی‌ها:

برای توسعه ویژگی‌های جدید بدون تأثیر بر شاخه اصلی:

Terminal

```
git branch feature-branch
git checkout feature-branch
```

۳.۶.۲ ■ روش دوم: آپلود دستی فایل‌ها

پس از ایجاد مخزن کد، فایل‌های پروژه‌تان را به صورت دستی (مثلاً با drag and drop) به مخزن اضافه کنید.

ضمیمه: پیشنهادهای ما و آموزش‌ها

۴.۱ ◆ گیت (Git)

همانطور که از تمرین صفر به یاد دارید، گیت‌هاب محلی بود که در آن مخازن کد نسخه‌گذاری شده با سیستم کنترل نسخه[۱] گیت قرار می‌گرفتند. برای یادگیری کار با Git و GitHub توصیه می‌کنیم این ویدیو و این ویدیو را مشاهده بفرمایید.

۴.۲ ◆ شی‌گرایی

مفهوم شی‌گرایی در کلاس درس برایتان مورد بررسی قرار گرفت. اما برای مرور چند نکته برایتان در ادامه خواهیم آورد.

به طور کلی، از رویکرد شی‌گرا و برنامه‌نویسی شی‌گرا[۲] زمانی استفاده می‌کنیم که می‌خواهیم یک شی از دنیای واقعی را درون یک سیستم یا برنامه مدل کنیم. برای مثال با هم یک کتاب را مدل می‌کنیم.
فرض کنید مشخصه‌های یک کتاب به صورت زیر باشد:

- تعداد صفحات
- نام مولف
- باز یا بسته بودن (فرض کنید در ابتدا کتاب بسته است.)
- صفحه‌ی فعلی (فرض کنید در ابتدای کار کتاب در صفحه‌ی اول خود باشد.)

و همچنین عملیات زیر می‌تواند روی یک کتاب انجام شود:

- باز کردن کتاب
- بستن کتاب
- رفتن به صفحه‌ی `p` در کتاب

در این صورت کد زیر را خواهیم داشت:

Code

```
class Book:
    def __init__(self, number_of_pages, author):
        self.number_of_pages = number_of_pages
        self.author = author
        self.is_open = False
        self.current_page = 1

    def open_book(self):
        self.is_open = True

    def close_book(self):
        self.is_open = False

    def go_to_page_p(self, p):
        self.current_page = p
```

[۱] Version Control System (VCS)
[۲] Object-oriented programming (OOP)

۴.۳ ◆ تست واحد (Unit test)

به طور کلی تست واحد را برای آزمون عملکرد صحیح و منطقی بخش‌هایی از برنامه‌هایمان می‌نویسیم. فرض کنید یک فایل به نام `my_module.py` نوشته‌ایم که محتوای آن تابع زیر است:

Code

```
# my_module.py

def add_numbers(a, b):
    """
    Adds two numbers and returns the result.
    """
    return a + b
```

برای تست آن، از تست واحد (درون فایل `test_my_module.py`) زیر استفاده می‌کنیم.

Code

```
# test_my_module.py

import unittest
from my_module import add_numbers

class TestAddNumbers(unittest.TestCase):
    def test_add_positive_numbers(self):
        # Test adding two positive numbers
        self.assertEqual(add_numbers(2, 3), 5)

    def test_add_negative_numbers(self):
        # Test adding two negative numbers
        self.assertEqual(add_numbers(-2, -3), -5)

    def test_add_mixed_numbers(self):
        # Test adding a positive and a negative number
        self.assertEqual(add_numbers(-2, 3), 1)

if __name__ == "__main__":
    unittest.main()
```

و برای تست، فایل تستمان را اجرا می‌کنیم که قبول[۳] شدن آن به معنای عملکرد صحیح و به مشکل خوردن آن[۴] به معنای عملکرد نادرست است.

۴.۴ ◆ رسم نمودار

برای یادگیری ماژول‌های `matplotlib` و seaborn استفاده از دوره‌های آموزشی Kaggle یا مشاهده‌ی این ویدیو پیشنهاد می‌شود.

[۳] pass
[۴] fail