دانشگاه صنعتی همدان

پروژه‌ی درس برنامه‌سازی پیشرفته، نیمسال تحصیلی ۹۳۲

عنوان:

پیاده‌سازی سیستم مدیریت اطلاعات تحصیلی دانشجویان

۱- هدف

نوشتن برنامه‌ای برای مدیریت اطلاعات فردی و دروس اخذ/حذف شده‌ی دانشجویان یک دانشگاه در هر نیمسال تحصیلی است.

۲- شرح

در محیط عملیاتی موردنظر، دو موجودیت «دانشجو» و «درس» را شناسایی کرده‌ایم. برای هر موجودیت باید کلاسی تعریف کنیم که شامل فیلدها و متدهایی منطقاً مرتبط به معنای آن کلاس باشد. خصوصیات هریک از موجودیت‌ها به شرح زیر است:

دانشجو (شماره دانشجویی، کد ملی، نام، نام خانوادگی، نام پدر، تاریخ تولد، محل تولد، نیمسال ورود)
درس (کد درس، نام درس، تعداد واحد، ماهیت)

هر دانشجو می‌تواند چند درس را اخذ/حذف کند. به‌منظور جلوگیری از بروز افزونگی و آنومالی[1] یک کلاس جدید برای درس‌های اخذ شده‌ی دانشجویان و کلاس دیگری برای درس‌های حذف‌شده‌ی آن‌ها تعریف می‌کنیم. چهار کلاس یادشده منطقاً باید در یک پکیج قرار داشته باشند.

[1] این مفاهیم فراتر از درس برنامه‌سازی پیشرفته هستند و در درس پایگاه‌داده‌ها پوشش داده می‌شوند. در اینجا صرفاً خاطرنشان می‌شود که نگهداری اطلاعات دانشجویان و دروس اخذ/حذف در کنار یکدیگر صحیح نیست. به دلیل مشابه نباید معدل دانشجو را جزو خصوصیات ذخیره‌شده‌ی او برشمرد؛ بلکه هر بار باید آن را با توجه به سایر اطلاعات ذخیره شده محاسبه کرد.

درس‌های اخذ شده‌ی دانشجو (شماره دانشجویی، کد درس، نیمسال تحصیلی، نمره)
درس‌های حذف شده‌ی دانشجو (شماره دانشجویی، کد درس، نیمسال تحصیلی)

۲-۱- خصوصیات موردنظر

۱- واسط کاربری مناسب

برنامه‌ی مورد نظر باید واسط کاربری مناسبی داشته و دریافت و نمایش اطلاعات با نمایش پیغام‌های مناسب همراه باشد. کلیه‌ی ورودی‌ها باید کنترل شود تا در فرمت مناسب باشد و درصورتی که به‌عنوان مثال کاربر به‌جای نام دانشجو یک عدد را وارد کرد، باید پیغام مناسب نشان داده شده و ورودی مجدداً دریافت گردد. برنامه باید در یک حلقه‌ی بی‌نهایت تکرار شود تا کاربر بتواند هر تعداد عملیاتی که در نظر دارد را بدون اجرای مجدد برنامه انجام دهد. گزینه‌ای نیز برای پایان برنامه و خارج شدن از حلقه‌ی بی‌نهایت طراحی کنید.

۲- استفاده از فایل‌ها

هر یک از چهار کلاس فوق باید دارای یک فیلد از نوع فایل باشد که بین تمامی اشیاء آن کلاس مشترک است. اطلاعات هر شیء جدیدی که از هر کلاس ساخته شود، توسط سازنده‌ی آن کلاس درون فایل نوشته می‌شود. می‌توانید یک متد private تعریف کنید که سازنده برای نوشتن در فایل، آن را فراخوانی کند. عمل نوشتن نباید فرمت فایل را تغییر دهد. فرمت مورد نظر برای هر فایل بعداً ذکر خواهد شد.

۳- نگرش شیءگرایی

این برنامه باید با رویکرد شیءگرایی نوشته شود و هریک از قابلیت‌های موردنیاز باید توسط متدهای مناسب از کلاس مناسب فراهم گردد. هر کلاس باید منطق صحیحی داشته باشد و صرف کار کردن برنامه مد نظر نیست.

۲-۲- قابلیت‌های موردنظر

۱- تعریف یک دانشجوی جدید

با انتخاب این گزینه اطلاعات دانشجوی جدید یکی پس از دیگری دریافت شده و به سازنده‌ی کلاس دانشجو ارسال می‌شود. سازنده نیز پس از مقداردهی به فیلدهای کلاس، آن‌ها را در فایل دانشجویان می‌نویسد.

۲- جستجوی یک دانشجو

در حالت کلی جستجو می‌تواند براساس هریک از فیلدهای داده‌ای انجام شود. اما برای سهولت فرض می‌کنیم که تنها بر اساس شماره دانشجویی صورت می‌گیرد. بنابراین با انتخاب این گزینه، شماره‌ی دانشجوی مورد نظر دریافت شده و به متدی از کلاس دانشجو ارسال می‌شود[۲]. این متد فایل دانشجویان را جستجو کرده و در صورتی که دانشجویی با شماره‌ی داده شده یافت شود، با اطلاعاتی که از فایل می‌خواند شیئی از کلاس دانشجو ساخته و آن را به عنوان مقدار بازگشتی خود برمی‌گرداند[۳]. سپس اطلاعات دانشجوی یافت شده یک‌به‌یک از شیء بازگشتی خوانده شده و چاپ می‌شود. اگر دانشجو یافت نشود مقدار بازگشتی null خواهد بود و پیغام مناسب نشان داده می‌شود.

۳- تعریف یک درس جدید

کاملاً مشابه قابلیت اول.

٤- ثبت درس اخذشده‌ی یک دانشجو

برای این کار ابتدا شماره دانشجو دریافت شده و بررسی می‌شود که آیا چنین دانشجویی وجود دارد؟ (قابلیت دوم) سپس کد درس دریافت شده و بررسی می‌شود که آیا چنین درسی وجود دارد؟ درصورتی‌که پاسخ هردو سؤال مثبت بود، شیئی از کلاس «درس‌های اخذ شده‌ی دانشجو» ساخته شده و «شماره دانشجویی، کد درس، نیمسال تحصیلی، نمره» به سازنده‌ی آن ارسال می‌گردد. سازنده نیز بررسی می‌کند که آیا با ثبت این درس، مجموع تعداد واحدهای این دانشجو در نیمسال ذکر شده بیش

---

[۲] طبیعتاً این متد باید استاتیک باشد. استاتیک بودن یا نبودن سایر متدها را خودتان باید تشخیص دهید و برای آن دلیل داشته باشید.

[۳] چرا باید شیء بسازد و چرا اطلاعات را در قالب یک رشته برنگرداند؟

از ۲۰ نخواهد شد؟ در صورتی که پاسخ منفی است درس در فایل مربوطه ثبت می‌شود و در غیر این صورت خیر.

۵- ثبت درس حذف‌شده‌ی یک دانشجو

کاملاً مشابه قابلیت چهارم. بررسی وجود این درس لازم نیست ولی باید بررسی شود که آیا این دانشجو این درس را در نیمسال ذکر شده اخذ کرده است؟ آیا با حذف این درس، مجموع تعداد واحدهای این دانشجو در نیمسال ذکر شده کمتر از ۱۲ نخواهد شد؟

٦- محاسبه‌ی معدل ترمی دانشجو

برای انجام این کار نیز ابتدا شماره دانشجو دریافت شده و بررسی می‌شود که آیا چنین دانشجویی وجود دارد؟ (قابلیت دوم) درصورت وجود، نیمسال مورد نظر از ورودی دریافت و به متد محاسبه‌ی معدل ترمی دانشجو ارسال می‌شود[٤]. ممکن است برای درس نمره‌ای درج نشده باشد.

۷- محاسبه‌ی معدل کل دانشجو

مشابه قابلیت ششم.

۸- مشاهده‌ی اطلاعات تمام دانشجویان.

۹- مشاهده‌ی دروس اخذ شده‌ی یک دانشجو

با استفاده از قابلیت دوم.

[٤] این متد تنها یک عدد اعشاری برمی‌گرداند و منطقاً خودش نباید چیزی چاپ کند.

۲–۳– توضیح بیش‌تر در مورد فیلدهای هر کلاس

برای هر یک از فیلدهای هر کلاس یک getter تعریف کنید. قابلیت دید تمامی فیلدها باید private باشد.
فرمت هریک از فیلدها در فایل مربوطه:

- شماره دانشجویی: رشته‌ای به طول ۷؛
- کد ملی، نام، نام خانوادگی، نام پدر، تاریخ و محل تولد: هرکدام رشته‌ای به طول ۱۰؛
- نیمسال ورود به دانشگاه: رشته‌ای به طول چهار. مثلاً ۳۹۰۱ یعنی نیمسال اول سال ۱۳۹۰.
- کد درس: رشته‌ای به طول ۳؛
- نام درس: رشته‌ای به طول ۱۶؛
- تعداد واحد: رشته‌ای به طول یک.
- ماهیت که یکی از مقادیر «اصلی، عمومی، اختیاری، جبرانی، پایه» است: رشته‌ای به طول ۸؛
- نمره: رشته‌ای به طول ۵.

برخی از فیلدهای ذکر شده طول ثابتی دارند (مانند کد ملی) و برخی دیگر ممکن است از حداکثر طول ذکر شده کوتاه‌تر باشند (مانند نام خانوادگی). چنین فیلدهایی با فضای خالی تکمیل می‌شوند تا فایل شکل منظمی داشته باشد. هر فیلد با یک فضای خالی از فیلد بعدی جدا می‌شود.

مثالی از محتویات چهار فایل مورد نیاز که هریک فیلدی از یک کلاس است، در کنار صورت پروژه ارائه شده است. به فضاهای خالی بین رشته‌ها دقت کنید.

*پیاده‌سازی کامل سیستم موردنظر نیازمند در نظر گرفتن جزئیات بسیاری است که برای سهولت از آن‌ها صرف‌نظر می‌کنیم. مثلاً پیش از ثبت درس باید بررسی شود که آیا دانشجو این درس را قبلاً پاس کرده یا خیر و بسیاری جزئیات دیگر...