TaavSysHomeWork-Part3

1. یک کلاس به نام `Person` طراحی کنید که ویژگی‌های `FirstName` و `LastName` را داشته باشد و یک متد به نام `GetFullName` که نام و نام خانوادگی را با هم ادغام کرده و برمی‌گرداند
2. ک سیستم مدیریت کارمندان را پیاده‌سازی کنید. دو کلاس به نام `Employee` و `Manager` داریم. کلاس `Employee` ویژگی‌های `EmployeeID` و `Salary` را دارد. کلاس `Manager` از `Employee` ارث‌بری می‌کند و یک ویژگی اضافی به نام `Department` دارد. هر دو کلاس همچنین یک متد به نام `PrintDetails` دارند که جزئیات کارمند را چاپ می‌کند
3. یک سیستم اجاره خودرو طراحی کنید. از اصول ارث‌بری، کپسوله‌سازی و پلی‌مورفیسم بهره ببرید
4. یک بازی ساده با سه شخصیت "قهرمان"، "دشمن" و "موجود" طراحی کنید. از اصل ارث‌بری برای ایجاد شخصیت‌ها استفاده کنید و هر شخصیت ویژگی‌های خاص خود را مثل قدرت یا سرعت داشته باشد
5. یک سیستم مدیریت دانشجویی را پیاده‌سازی کنید. از ارث‌بری، اینترفیس، و ویژگی‌های مختلف مانند نمره، واحد، و مشخصات فردی دانشجو استفاده کنید.
6. یک برنامه مدیریت کتابخانه بنویسید. از اینترفیس برای تعریف عملیات مشترک مانند اضافه کردن کتاب، حذف کتاب و نمایش جزئیات استفاده کنید. سپس کتابخانه را با کلاس‌های مختلف برای انواع مختلف کتاب‌ها پیاده‌سازی کنید
7. یک سیستم مدیریت پروژه با استفاده از اصول OOP طراحی کنید. از اینترفیس برای تعریف عملیات مشترک بین پروژه‌ها و از ارث‌بری برای ایجاد انواع مختلف پروژه‌ها استفاده کنید)سیستم ثبت ساعات ورود خروج یک شرکت مهندسی و محاسبه اضافه کاری یا کم کاری)
8. یک برنامه مدیریت وظایف (To-Do List) طراحی کنید که از اصول ارث‌بری و اینترفیس برای تعریف وظایف با اولویت‌ها و تاریخ‌های مختلف بهره‌مند باشد
9. یک برنامه مدیریت رستوران طراحی کنید که از اصول ارث‌بری برای انواع غذاها (مانند پیتزا و پاستا) و از اصل کپسوله‌سازی برای مدیریت سفارش‌ها بهره‌مند باشد.
10. توضیح دهید چگونه اصل "کپسوله‌سازی" در برنامه‌نویسی شیءگرا کمک به نگهداری کد می‌کند؟

اصل کپسوله سازی سازی به ما کمک میکند تا از دسترسی به object های ساخته شده از کلاس به جز درمواردی که طراح در نةر گرفته جلوگیری شود که این موضوع از ایجاد خرابی های اتفاقی و عمدی در برنامه جلوگیری میکندو همچنین این اصل سبب کاهش پیچیدگی و متصل شدن اجزای کد به یکدیگر می شود که خود یک اصل حیاتی برای توسعه و نگهداری پروزه محسوب می شود

1. تفاوت اصل "پلی‌مورفیسم" با (Overloading) و (Overriding) چیست ؟

پلی مورفیسم د رحقیقت به خاصیتی گفته میشود که بوسیله آن میتوان یک رفتار را در کلاس هایی که از یک کلاس واحد ارث بری میکنند پیاده سازی کرد, این اصل با استفاده از Override کردن یک متود در (behavior) در کلاس های ارث بری شده از یک کلاس واحد انجام میپذیرد اما مفهوم overloading مقداری متفاوت با overriding است، overloading به معنای این است که از یک متود با یک اسم چندین متود با ورودی های مختلف بسازیم تا تنوع کاربردی آن متود بالا رود

1. تفاوت میان اصطلاحات "Interface" و "Abstract Class" چیست و در چه مواقعی از هرکدام استفاده می‌شود؟ Interface ها یک کلاس نیستند و در درون آن ها متد های بدون بدنه و فیلد تعریف می شود ولی Abstract Class ها کلاس هیتند و میتوانند داری property و متد با بدنه و بدون بدنه باشند که نشان دهنده این است که میتواند یا 100 درصد از مشابه Interface رفتار کند یا نه تنها درصدی از آن و درصدی از مشخصات کلاس را هم داشته باشد، interface ها معمولا برای انجام عملیات خاص نظیر عملیات CRUD یا عملیاتی که در اکثر کلاس ها موجودیت پیدا میکنند (نظیر سرویس ها) استفاده می شوند درحالی که از Abstract Class ها برای ارث بری شدن و ساخت کلاس های زیر دسته استفاده می شود.