**به نام خدا**

تمرینات درس مهندسی نرم‌افزار **:**

**UMLبخش اول: زبان**

**:UML (Unified Modeling Language) 1. تعریف**

UML یک زبان مدلسازی تصویری استاندارد است که برای تجزیه، طراحی، ساخت و مستندسازی سیستم‌های نرم‌افزاری و غیرنرم‌افزاری استفاده می‌شود. این زبان برای مدل‌سازی سیستم‌های شی‌گرا توسعه یافته و از مجموعه‌ای از نمودارها تشکیل شده که جنبه‌های مختلف سیستم را نمایش می‌دهد.

هدف اصلی UML، ساده‌سازی فرآیند طراحی سیستم‌های پیچیده و ایجاد زبان مشترک بین اعضای تیم توسعه نرم‌افزار است.

**2. فهرست نمودارهای UML:**

UML شامل **۱۴ نمودار استاندارد** است که در دو دسته‌ی اصلی تقسیم می‌شوند:

**الف) نمودارهای ساختاری (Structural Diagrams)**

نمایانگر ساختار ایستای سیستم هستند:

1- Class Diagram

2- Object Diagram

3-Component Diagram

4-Deployment Diagram

5-Package Diagram

6-Profile Diagram

**ب) نمودارهای رفتاری (Behavioral Diagrams)**

نمایانگر رفتار و تعاملات سیستم در طول زمان هستند:

1-Use Case Diagram

2-Activity Diagram

3-State (State Machine) Diagram

4-Sequence Diagram

5-Communication Diagram

6-Interaction Overview Diagram

7-Timing Diagram

**.3کاربردهای : UML**

* مستندسازی ساختار و رفتار سیستم‌های نرم‌افزاری
* تسهیل ارتباط بین تیم‌های تحلیل، طراحی و توسعه
* طراحی شی‌گرا قبل از شروع کدنویسی
* تحلیل نیازمندی‌های نرم‌افزار
* تولید خودکار کد در برخی ابزارهای پیشرفته
* شناسایی وابستگی‌ها و جریان‌های داده

**4. مزایای UML:**

* یک استاندارد جهانی برای مدلسازی نرم‌افزار است.
* ابزارهای متعددی برای پشتیبانی از UML وجود دارد.
* به تحلیلگران، طراحان و برنامه‌نویسان زبان مشترک می‌دهد.
* فهم آن حتی برای افراد غیرفنی ساده‌تر از کد است.
* مستندسازی ساختار سیستم برای نگهداری آینده آسان‌تر می‌شود.

**5. معایب UML:**

* برای پروژه‌های کوچک، ممکن است زمان‌بر و غیرضروری باشد.
* درک کامل آن برای مبتدیان ممکن است سخت باشد.
* ممکن است باعث پیچیدگی بیش از حد در مستندسازی شود.
* بعضی از نمودارها، همپوشانی دارند و تفکیک آن‌ها چالش‌برانگیز است.
* در مواردی نیاز به ابزارهای خاص برای ویرایش و تفسیر نمودارهاست.

**6. جنبه‌های ایستا و پویا در UML:**

**ایستا (Static):** نمایش ساختار سیستم بدون در نظر گرفتن زمان و تغییرات)مثلاً Class .(Diagram

**پویا (Dynamic):** نمایش رفتار و تعاملات سیستم در طول زمان )مثلاً Sequence Diagram، .(Activity Diagram

**.7 نمودارهای UML با تعریف، مثال و توضیح:**

**:** **Use Case Diagram.1**

**تعریف:** نمایش‌دهنده‌ی تعاملات کاربران (Actors) با سیستم.

**مثال:** در سیستم فروش آنلاین، Actor: مشتری، Use Cases: مشاهده محصول، افزودن به سبد، پرداخت.

**توضیح:** برای تحلیل نیازمندی‌ها کاربرد دارد و مشخص می‌کند چه کسانی از سیستم چه انتظاری دارند.

**: Activity Diagram.2**

تعریف: نمایش گردش کار (workflow) یک فرآیند یا تابع در سیستم.

مثال : فرآیند سفارش محصول → ورود اطلاعات، تأیید پرداخت، ثبت سفارش.

توضیح: نمایش ترتیبی از فعالیت‌ها با شرایط انشعابی و موازی.

**3. : State (Machine) Diagram**

تعریف: نمایش وضعیت‌های مختلف یک شیء در طول زمان و نحوه‌ی انتقال بین وضعیت‌ها.

مثال: سفارش → وضعیت‌ها: "در انتظار پرداخت"، "پرداخت‌شده"، "ارسال‌شده".

توضیح: مناسب برای مدل‌سازی اشیایی با وضعیت‌های متغیر.

**4. : Communication Diagram**

تعریف: نمایش تعاملات بین اشیاء با تأکید بر ساختار ارتباطات.

مثال: تعامل بین کاربر، سرویس احراز هویت و پایگاه‌داده.

توضیح: مشابه Sequence Diagram ولی بر توپولوژی (ساختار شبکه‌ای) تمرکز دارد.

**5. : Sequence Diagram**

تعریف: نمایش ترتیب ارسال پیام‌ها بین اشیاء در طول زمان.

مثال: فرآیند ورود کاربر → ارسال درخواست → بررسی اعتبارسنجی → تأیید یا خطا.

توضیح: خطوط عمودی زمان و پیام‌ها به ترتیب نمایش داده می‌شود.

**6: Timing Diagram.**

**تعریف:** نمایش تغییر وضعیت اشیاء در طول زمان با تأکید بر زمان‌بندی دقیق.

**مثال:** در سیستم کنترل ترافیک، تغییر وضعیت چراغ‌ها (سبز، زرد، قرمز).

**توضیح:** برای سیستم‌های Real-time کاربرد زیادی دارد.

**7: Interaction Overview Diagram.**

**تعریف:** ترکیبی از Activity و Interaction Diagrams.

**مثال:** پردازش چند مرحله‌ای سفارش شامل تعاملات مختلف.

**توضیح:** فرآیند کلی را نمایش داده و تعاملات مختلف را بخش‌بندی می‌کند.

**8: Profile Diagram.**

**تعریف:** برای گسترش UML و سفارشی‌سازی مفاهیم با استفاده از stereotypes استفاده می‌شود.

**مثال:** تعریف نوع خاصی از کلاس مثل <<WebController>> در یک پروژه خاص.

**توضیح:** در توسعه‌ی metamodelها مفید است.

**9: Composite Structure Diagram.**

**تعریف:** ساختار داخلی یک کلاس یا component شامل پورت‌ها و اجزای داخلی.

**مثال:** کلاس "مدیر سیستم" با اجزای داخلی مانند پایگاه داده و رابط کاربری.

**توضیح:** تعامل اجزای درونی یک کلاس پیچیده را نشان می‌دهد.

**10: Class Diagram.**

**تعریف:** نمایش کلاس‌ها، ویژگی‌ها (attributes)، توابع (methods)، و روابط بین آن‌ها.

**مثال:** کلاس‌های Student، Course، و رابطه Enroll.

**توضیح:** پایه‌ای‌ترین و مهم‌ترین نمودار UML برای طراحی شی‌گرا.

**11: Package Diagram.**

**تعریف:** نمایش گروه‌بندی کلاس‌ها یا اجزای مرتبط در پکیج‌ها.

**مثال:** پکیج‌های UI، Business Logic، Data Access.

**توضیح:** ساختار سیستم در سطح کلان را مشخص می‌کند.

**12: Object Diagram.**

**تعریف:** نمایش نمونه‌های واقعی از کلاس‌ها در یک لحظه خاص.

**مثال:** obj1:Student(name="Ali"), obj2:Course(title="UML").

**توضیح:** وضعیت لحظه‌ای سیستم را نمایش می‌دهد.

**13: Component Diagram.**

**تعریف:** نمایش اجزای نرم‌افزاری و وابستگی‌های بین آن‌ها.

**مثال:** سرویس REST API، پایگاه‌داده، UI.

**توضیح:** برای معماری سیستم‌های بزرگ مفید است.

**14: Deployment Diagram.**

**تعریف:** نمایش استقرار فیزیکی اجزای نرم‌افزار روی سخت‌افزارها.

**مثال:** وب‌سرور، پایگاه‌داده، کلاینت در معماری سه‌لایه.

**توضیح:** سیستم را از منظر زیرساخت نمایش می‌دهد.