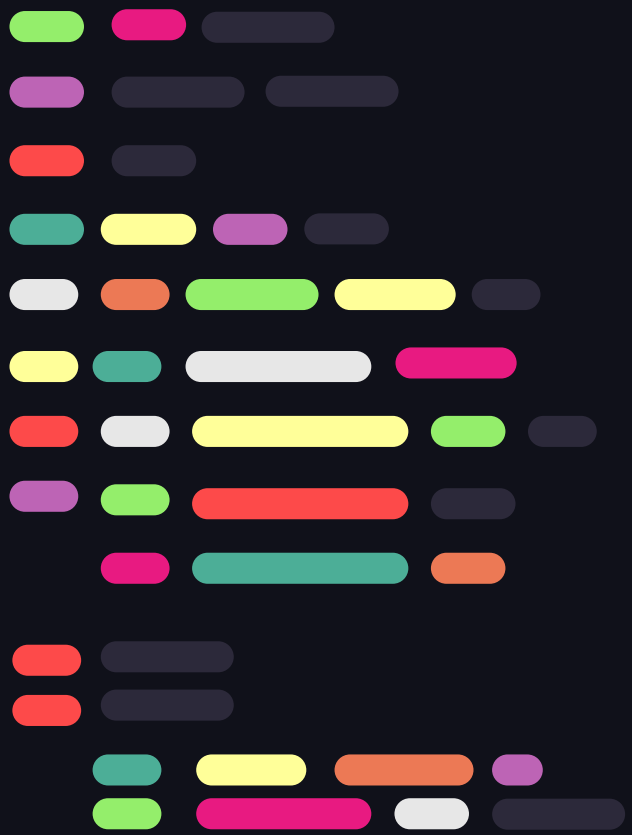
ESCUELA DE INGENIERÍA EN CIENCIAS Y SISTEMAS
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

ECYS | DIDT
DESARROLLO DE TRANSFERENCIA TECNOLÓGICA

Día, Fecha:	Lunes, 10/06/2024
Hora de inicio:	13:00

Introducción a la Programación y Computación 1 [A]

Josué Rodolfo Morales Castillo



Clase 6- Agenda

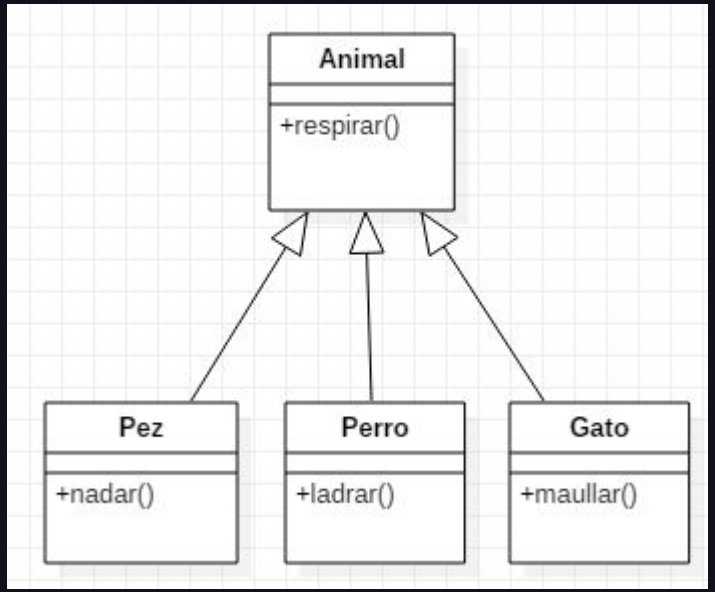
- Foro No. 2
- Preguntas práctica 1
- Principios básicos de UML (Diagrama de clases)

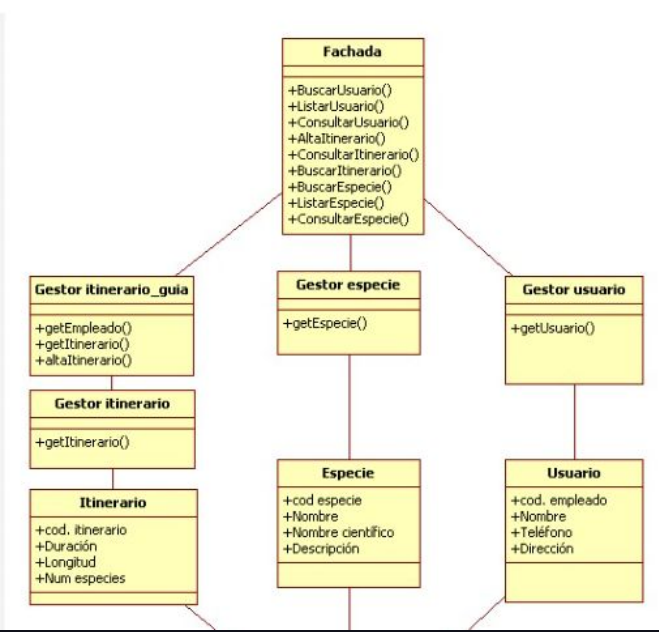
Principios básicos de UML (Diagrama de clases)

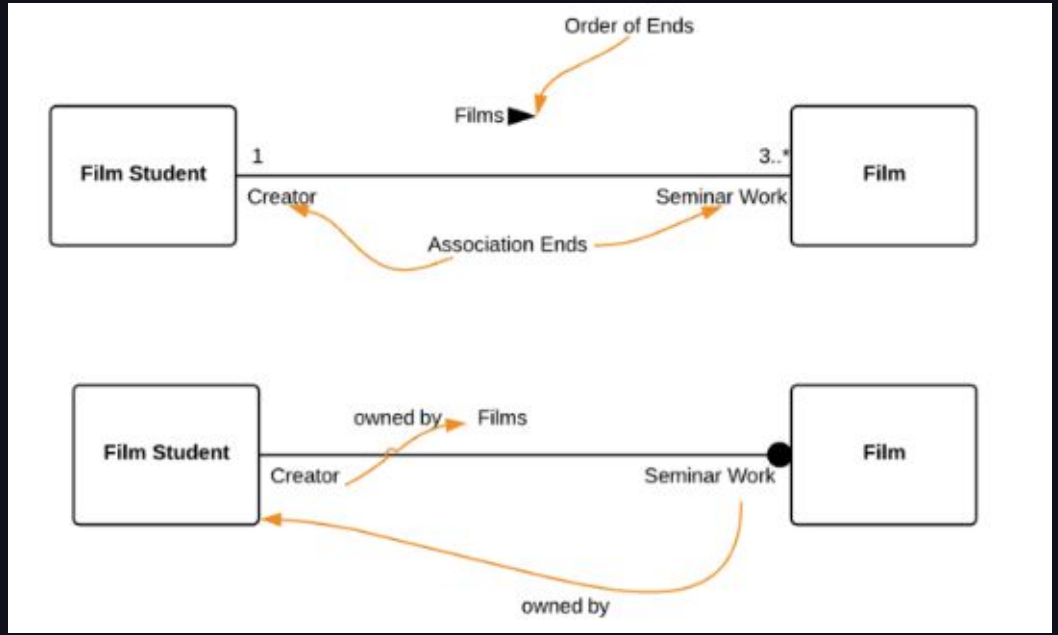
https://youtu.be/Z0yLerU0g-Q?si=to4mg0Qp7Bn5JYCM

¿Qué es un diagrama de clases?

Es una representación gráfica que describe la estructura de un sistema orientado a objetos, delineando las clases, sus atributos y métodos, así como las relaciones entre ellas, como asociaciones, herencias y composiciones. Es una herramienta de análisis y diseño de sistemas de software, facilitando la comprensión de la arquitectura del sistema y mejorando la comunicación entre los miembros del equipo de desarrollo al ofrecer una vista organizada y precisa de las entidades y sus interacciones.







Los diagramas de clases son uno de los tipos de diagramas más útiles en UML, ya que trazan claramente la estructura de un sistema concreto al modelar sus clases, atributos, operaciones y relaciones entre objetos.

Definición de Clases

Clase: Representa un conjunto de objetos que comparten características comunes y comportamientos. Se representa en el diagrama con un rectángulo dividido en tres secciones: la superior contiene el nombre de la clase, la del medio incluye los atributos, y la inferior muestra los métodos.

Atributos: Características o propiedades de la clase. Se muestran en la sección de atributos y pueden tener diferentes niveles de visibilidad (públicos, privados, protegidos). Ejemplo:
+nombre: String indica un atributo público llamado nombre de tipo String.

Métodos: Comportamientos que la clase puede realizar. Se representan en la sección de métodos y también tienen niveles de visibilidad. Ejemplo:
+calcularPrecio(): double método público llamado calcularPrecio que devuelve un valor de tipo double.

Relaciones

Asociacion

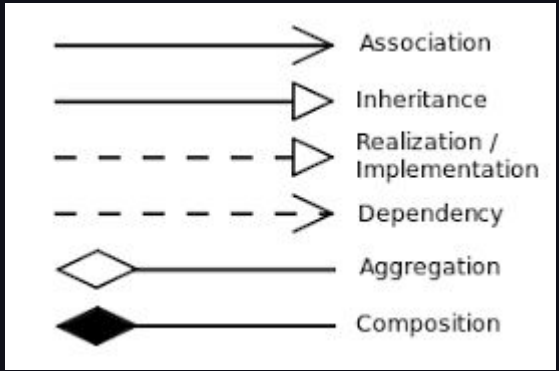
Representa una relación entre dos clases. Puede ser unidireccional o bidireccional. Se muestra con una línea y puede tener una multiplicidad que indica cuántos objetos de cada clase participan en la relación.

Herencia

Representa una relación "es-un" entre clases. La subclase hereda atributos y métodos de la superclase. Se muestra con una línea y una flecha apuntando hacia la superclase.

Agregacion

Indica una relación "todo-parte" entre clases. Se representa con un rombo en la clase que contiene los objetos y una línea hacia la clase contenida.



Composicion

Es una forma más fuerte de agregación, indicando que una clase es parte de otra y no puede existir independientemente. Se representa con un rombo relleno.

Dependencia

Indica que un cambio en una clase puede afectar a otra. Se representa con una línea punteada.

Ámbito de las propiedades

Público (+)

Accesible desde cualquier parte del sistema.

Privado (-)

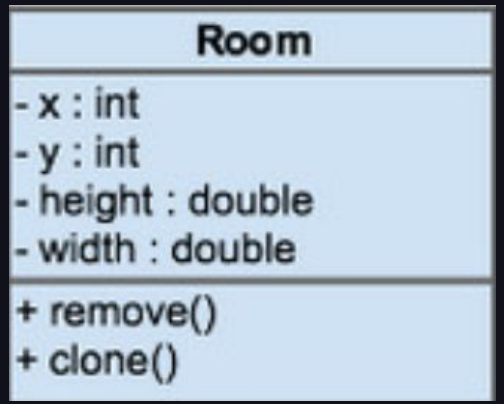
Solo accesible dentro de la propia clase

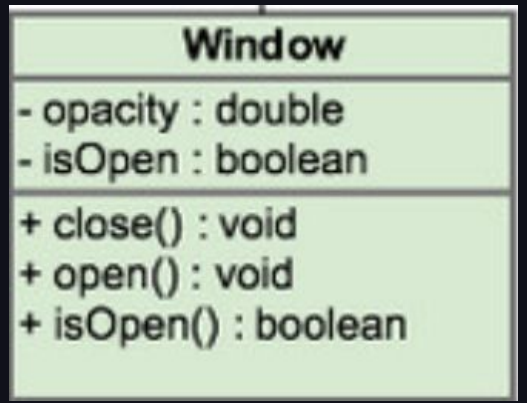
Protegido (#)

Accesible dentro de la propia clase y sus subclases.

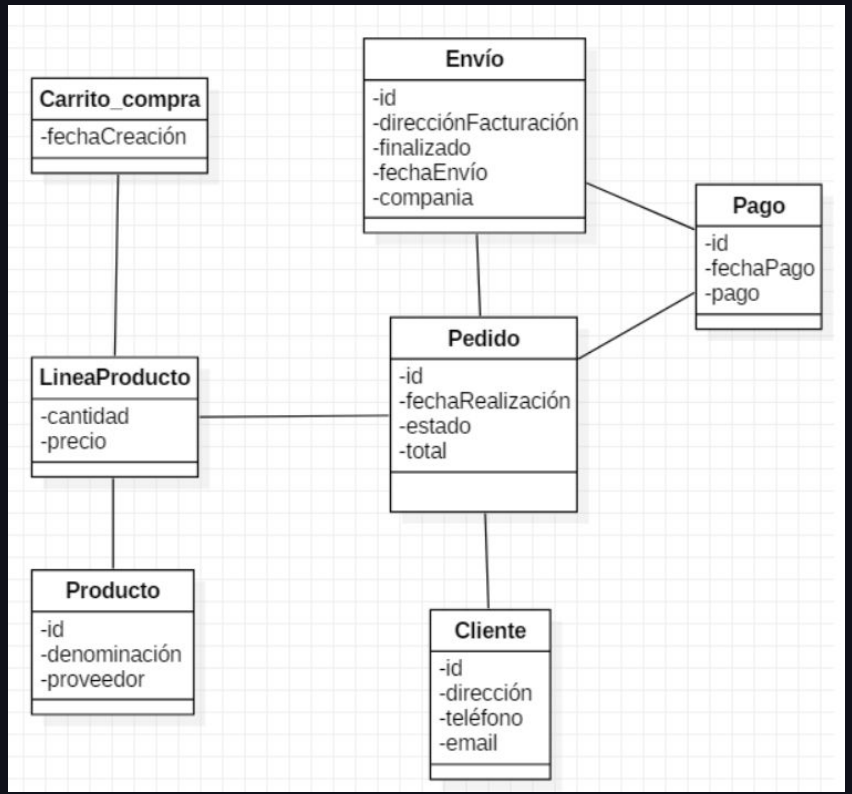
Paquete (~)

Accesible dentro del paquete que contiene la clase





Diseño de programas



Se utiliza para el diseño de programas orientados a objetos. Permite visualizar la estructura de un sistema, sus componentes y cómo interactúan entre sí.

Clases de asociaciones

- Asociación Simple: Es la relación más básica entre dos clases. Representa una conexión general sin distinciones especiales. Una asociación simple podría ser entre las clases "Estudiante" y "Curso", donde un estudiante está relacionado con uno o más cursos.

- Agregación: Representa una relación de "todo-parte", donde una clase (el todo) contiene o está compuesta por otras clases (las partes).
  Una agregación podría ser entre "Departamento" y "Empleado", donde un departamento contiene empleados, pero los empleados pueden existir independientemente del departamento.

- Composición: Es una forma más fuerte de agregación, donde la parte no puede existir independientemente del todo. Si el todo se destruye, las partes también se destruyen.
  Un ejemplo sería la relación entre "Carro" y "Rueda", ya que si el carro se destruye, las ruedas también se destruyen.

Multiplicidad y dependencia

Multiplicidad

La multiplicidad especifica cuántas instancias de una clase pueden estar asociadas con una instancia de la otra clase.
Por ejemplo una multiplicidad de "0..*" para la relación entre "Cliente" y "Cuenta Bancaria" indica que un cliente puede tener ninguna o varias cuentas bancarias.

Dependencia

Indica que un cambio en una clase puede afectar a otra clase. Puede ser de uso (una clase utiliza la otra), realización (una clase implementa una interfaz) o derivación (una clase hereda de otra).
Por ejemplo si la clase "Factura" utiliza la clase "Producto", hay una dependencia entre ellas.

Relaciones múltiples (asociativas) y reflexivas

Relaciones múltiples

Las relaciones múltiples, también conocidas como asociativas, son aquellas en las que más de dos clases están conectadas. Estas relaciones se utilizan cuando hay una entidad adicional que describe la relación entre las clases originales.

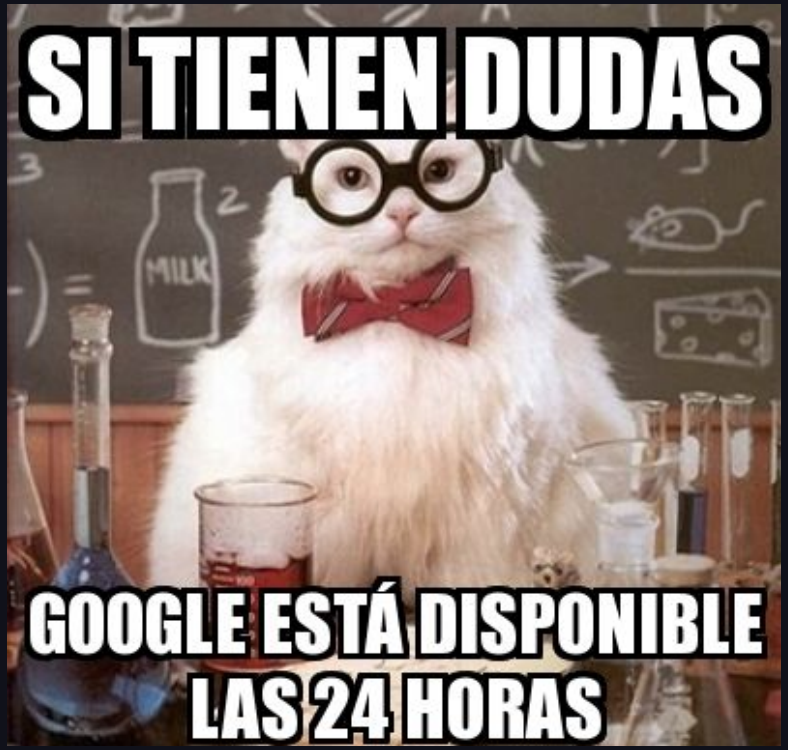
Por ejemplo supongamos que se tienen las clases "Estudiante" y "Curso". Se puede tener una relación asociativa llamada "Inscripción", que conecta a un estudiante con un curso, y además, puede tener atributos adicionales, como la fecha de inscripción.

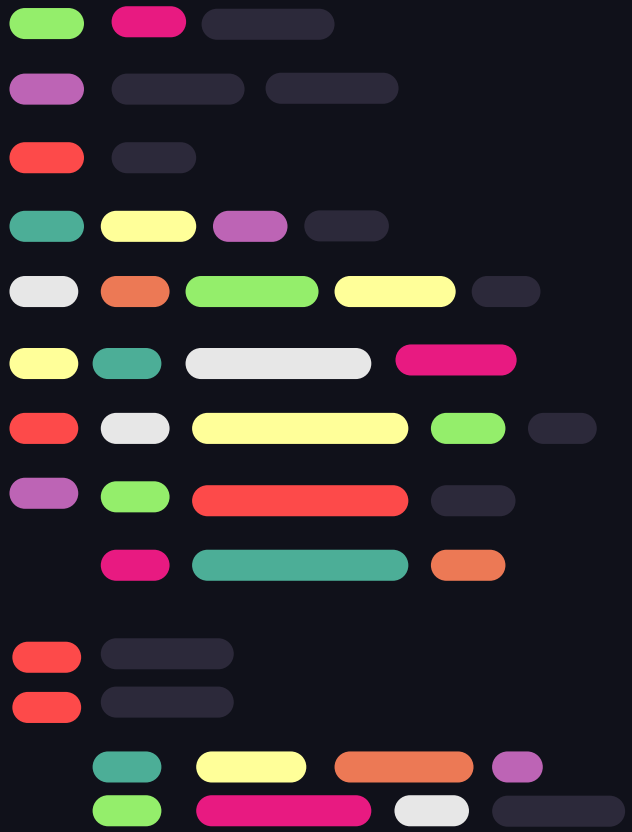
Relaciones reflexivas

Una relación reflexiva es aquella en la que una clase está relacionada consigo misma. Esto es útil cuando una instancia de la clase tiene una relación con otra instancia de la misma clase.

Por ejemplo supongamos que se tiene la clase "Persona" y se quiere representar la relación de parentesco entre personas. Se puede tener una relación reflexiva llamada "Parentesco" que conecta a una persona con otra y tiene un atributo para indicar el tipo de relación (padre, madre, hermano, etc.).

¿Dudas?





Ejemplo

Descargar JFreeChart:

https://sourceforge.net/projects/jfreechart/files/