

M4 Programación orientada a objetos FIN A

ACTIVIDAD 2

|  |  |
| --- | --- |
| **Tutor:** | **Victor Madera Hernández** |
| **Estudiante:** | **José Ramón Ibáñez Posadas** |
| **Matricula:** | **BNL098377** |

|  |  |
| --- | --- |
| Monterrey, Nuevo León | Lunes, 15 de Abril de 2024 |

INTRODUCCIÓN

En el desarrollo de software orientado a objetos, el diseño adecuado de las clases y sus interacciones es esencial para construir sistemas robustos y mantenibles. Uno de los pilares fundamentales de este proceso es comprender y manejar adecuadamente aspectos como las relaciones y la colaboración entre clases, la identificación y definición de atributos, así como la representación visual mediante diagramas UML.

DESARROLLO

RESUMEN

**Relaciones y Colaboración entre Clases:**

En el diseño orientado a objetos, las clases son los bloques fundamentales de construcción, representando entidades o conceptos del dominio del problema. La relación entre clases define cómo interactúan estas entidades entre sí para lograr los objetivos del sistema. Hay varios tipos de relaciones, como:

**Asociación:** Es una relación entre dos clases donde un objeto de una clase está relacionado con uno o más objetos de otra clase. Por ejemplo, en un sistema bancario, la clase Cliente puede estar asociada con la clase Cuenta.

**Agregación y Composición:** Ambos son tipos de asociación que representan la relación de "todo-parte". En la agregación, los objetos pueden existir de forma independiente fuera de la clase principal, mientras que en la composición, los objetos solo existen dentro de la clase principal y se crean y destruyen con ella.

**Herencia:** Permite que una clase (subclase) herede atributos y comportamientos de otra clase (superclase), lo que promueve la reutilización del código y la organización jerárquica de las clases.

La colaboración entre clases se refiere a cómo estas entidades cooperan para cumplir sus responsabilidades. Esto puede implicar la delegación de tareas, la comunicación a través de métodos o la transferencia de información mediante el paso de parámetros.

**Diagrama y Esquema Preliminar de Clases:**

El diagrama de clases es una herramienta visual que muestra la estructura estática del sistema, incluyendo las clases, sus atributos y métodos, así como las relaciones entre ellas. Este diagrama proporciona una vista de alto nivel del diseño del sistema y sirve como una guía para el desarrollo posterior.

Antes de crear el diagrama de clases completo, es útil elaborar un esquema preliminar de clases. Esto implica identificar las clases principales del sistema y definir sus responsabilidades y relaciones iniciales. El esquema preliminar sirve como punto de partida para el diseño detallado de clases y puede evolucionar a medida que se comprenden mejor los requisitos del sistema.

**Atributos de Clases:**

Los atributos de una clase representan las características o propiedades que tienen los objetos de esa clase. Estos atributos describen el estado de los objetos y pueden ser de diferentes tipos de datos, como enteros, cadenas de texto, booleanos, entre otros. Es importante identificar y definir adecuadamente los atributos de cada clase para garantizar una representación precisa del sistema.

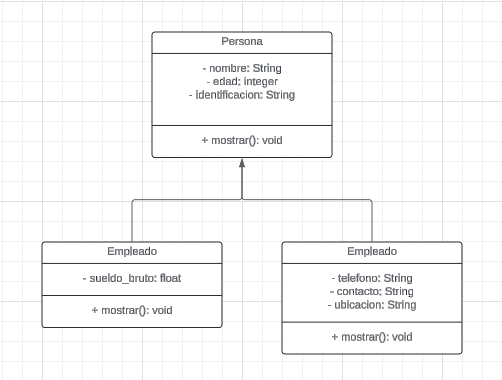
Los atributos pueden tener diferentes niveles de visibilidad, como público, protegido o privado, lo que determina quién puede acceder y modificar estos atributos desde fuera de la clase.

**Relación de Asociación:**

La relación de asociación indica una conexión semántica entre las clases, donde un objeto de una clase está relacionado con uno o varios objetos de otra clase. Esta relación puede ser unidireccional o bidireccional, y puede tener multiplicidad para indicar cuántos objetos están involucrados en la relación.

La asociación puede ser simple, donde las clases están relacionadas directamente, o puede ser una asociación con roles, donde se asigna un papel específico a cada clase en la relación. Por ejemplo, en un sistema de gestión de pedidos en línea, puede haber una asociación entre las clases Cliente y Pedido, donde el cliente realiza uno o varios pedidos.

DIAGRAMA UML



CONCLUSIÓN

En resumen, comprender y manejar aspectos como las relaciones y la colaboración entre clases, la identificación y definición de atributos, y la representación visual mediante diagramas UML son fundamentales en el diseño efectivo de sistemas orientados a objetos. La habilidad para utilizar estos conceptos y herramientas de manera adecuada contribuye significativamente a la creación de sistemas bien estructurados, flexibles y fáciles de mantener. Así mismo logramos adquirir el conocimiento necesario poder crear los diagramas UML de manera fácil.

BIBLIOGRAFÍA