Universidad Nacional de Villa Mercedes

ESCUELA DE INGENIERÍA Y CIENCIAS AMBIENTALES

Ejercicio N°4 – Aplicando los aprendido 3

Carrera: Ingeniería en Sistemas de Información.

Materia: Paradigmas de Programación.

Profesor/res: Walter Molina, Diego Puertas.

Alumno: Germán Adrián Muñoz.

Fecha de entrega: 08/10/2023

Ejercicio N°4

Explica en un texto, con ejemplos y fundamentación qué características de la OOP utilizaste para resolver los programas de los Ejercicios 2 y 3. Si hay alguna que no utilizaste o no implementaste, indica cuál y por qué crees que no fue necesario.

- Calculadora:
    - Clases y Objetos: Aunque no se utilizan clases en el sentido tradicional, se crea un objeto Calculadora que encapsula la funcionalidad de la calculadora. Este objeto contiene métodos como sumar, restar, multiplicar y dividir, que son como "métodos" de una clase de calculadora.
    - Encapsulación: Los métodos de la calculadora (sumar, restar, etc.) encapsulan la lógica de sus respectivas operaciones matemáticas. Esto significa que cada método opera solo en sus argumentos y no afecta directamente otras partes del programa.
    - Abstracción: Aunque no se implementa de manera explícita, los métodos de la calculadora representan una forma de abstracción al ocultar los detalles internos de cómo se realizan las operaciones matemáticas.
- Aplicación de Tareas:
    - Clases y Objetos: En este programa, se utilizan clases y objetos de manera más tradicional. La clase Task se define para representar tareas individuales, y las instancias de esta clase se utilizan para mantener y manipular los datos de las tareas.
    - Encapsulación: La clase Task encapsula los datos de una tarea, como el título, la descripción y el estado. Además, la clase TaskList encapsula la lógica relacionada con la gestión de la lista de tareas.
    - Abstracción: La clase Task representa una abstracción de una tarea, con propiedades que describen sus atributos. La clase TaskList abstracta la lista de tareas y proporciona métodos para interactuar con ella.

- Características no utilizadas:
    - Herencia: En ambos programas, no se utilizó la herencia. No había una necesidad clara de crear una jerarquía de clases en la que una clase hija heredara propiedades o métodos de una clase padre.
    - Polimorfismo: Tampoco se utilizó el polimorfismo en estos programas. No se necesitaba que objetos de diferentes clases compartieran un método con el mismo nombre, pero con implementaciones diferentes.