Plan Formativo Full Stack Python	
Módulo	Programación Avanzada en Python
Tema	Herencia y polimorfismo
Nivel de Dificultad	Alta
Ejecución	Grupal
Duración	60 Min
Código Ejercicio	E.3.4.G1

Intención del aprendizaje o aprendizaje esperado:

- Creación de Clases con atributos y métodos.
- Herencia de Clases.
- Herencia Múltiple de Clases.
- Composición.
- Encapsulamiento.
- Polimorfismo.
- Creación de objetos de clase. Aplicación de métodos y uso de atributos.

Planteamiento del Problema:

En este ejercicio implementaremos un programa que defina una clase llamada **Condominio**. Ésta deberá poseer:

- **Atributos**
  - direccion, lista_administrador, lista_guardias, num_unidades_habitacionales, lista_unidades, cuenta_corriente

  Agregar 4 atributos adicionales que usted considere apropiados para una clase general de condominio.

- **Métodos**
  - get_direccion, set_direccion, set_administrador, get_administrador, add_guardia, del_guardia, get_guardias, get_unidades

  Agregue 4 métodos adicionales que usted considere apropiados para una clase general de condominio.

Cree las clases **Guardia**, **UnidadHabitacional**, **CuentaCorriente** con al menos 3 atributos y 4 métodos, respectivamente. Utilice composición para hacerlos parte de los atributos de **Condominio**.

Cree una clase llamada **Terreno**, en el cual se pueden emplazar construcciones como edificios o conjuntos de casas. Agregue al menos 6 atributos y 6 métodos apropiados para la clase **Terreno**.

Cree dos clases con herencia múltiple de las clases **Terreno** y **Comunidad**, que se llamen **CondominoVertical** (Que hace alusión a Edificios Habitacionales) y **CondominioHorizontal** (Conjuntos Privados de Casas).

Cree 5 atributos y 6 métodos propios que diferencien a cada una de estas subclases. Dentro de estos métodos demuestre 2 casos de uso de polimorfismo.

Utilice encapsulamiento para proteger variables que usted no desea puedan ser accedidas desde subclases o desde un programa principal. Discuta en grupo cuál es la utilidad de realizar la definición de variables de esta forma. Investigue el concepto de Name Mangling en Python. ¿Es posible acceder un atributo definido como **self.__nombre_atributo** desde una subclase o desde el programa principal? Experimente y muestre ejemplos.

Cree 2 instancias de edificios y 2 instancias de condominios horizontales y demuestre la utilización de sus atributos y métodos.

¿Qué aplicación más completa se le ocurriría implementar con los prototipos de este ejercicio?

Recursos Bibliográficos :

[1] Módulo 3 – Contenido 4: “Codificar un programa utilizando el concepto de herencia para resolver un problema de baja complejidad acorde al lenguaje Python”