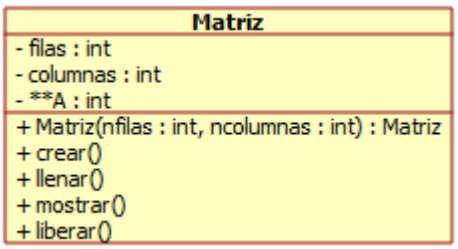
Trabajo práctico N° 1

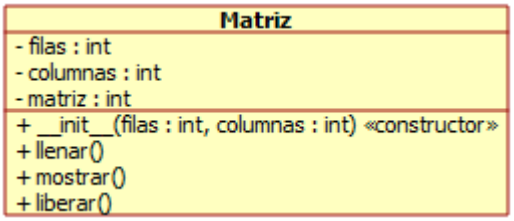
Ejercicio N° 1



La clase Matriz está diseñada para representar matrices bidimensionales de enteros.

- Atributos privados:
  - filas: Número de filas de la matriz.
  - columnas: Número de columnas de la matriz.
  - A: Puntero a un puntero a enteros, que representa la matriz bidimensional.
- Métodos públicos:
  - Matriz(int nfilas, int ncolumnas): Constructor que inicializa la matriz con las dimensiones especificadas.
  - crear(): Crea la matriz dinámica.
  - llenar(): Llena la matriz con números aleatorios entre 0 y 9.
  - mostrar(): Imprime la matriz en la consola.
  - liberar(): Libera la memoria utilizada por la matriz.

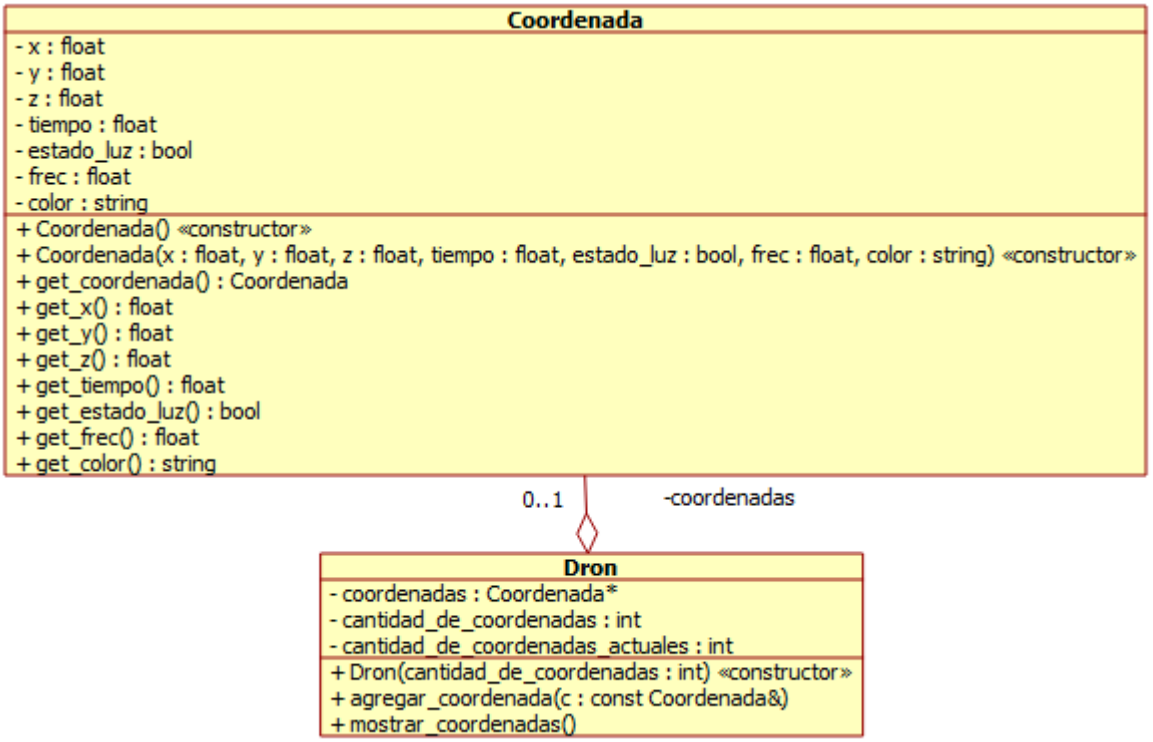
Ejercicio N° 2



La clase Matriz en Python proporciona una implementación de matrices bidimensionales.

- Atributos privados:
  - _filas: Número de filas de la matriz.
  - _columnas: Número de columnas de la matriz.
  - _matriz: Lista de listas que representa la matriz bidimensional.
- Métodos públicos:
  - __init__(filas, columnas): Constructor que inicializa la matriz con las dimensiones especificadas.
  - llenar(): Llena la matriz con números aleatorios entre 0 y 9.
  - mostrar(): Imprime la matriz en la consola.
  - liberar(): Libera la memoria utilizada por la matriz (en Python, esto se logra simplemente eliminando la referencia a la matriz).

Ejercicio N° 3



El código define dos clases principales: Coordenada y Dron.

- Coordenada: Representa un punto en el espacio con atributos como posición (x, y, z), tiempo, estado de la luz, frecuencia y color.
- Dron: Representa un dron que puede almacenar múltiples coordenadas a lo largo de su trayectoria.

El programa principal lee un archivo de texto con datos de coordenadas y crea un objeto Dron para almacenar y mostrar esta información.

Clase Coordenada

Atributos: Almacena información detallada sobre una coordenada, incluyendo atributos físicos, temporales y de color.

Métodos: Proporciona métodos para obtener el valor de cada atributo individualmente.

Clase Dron

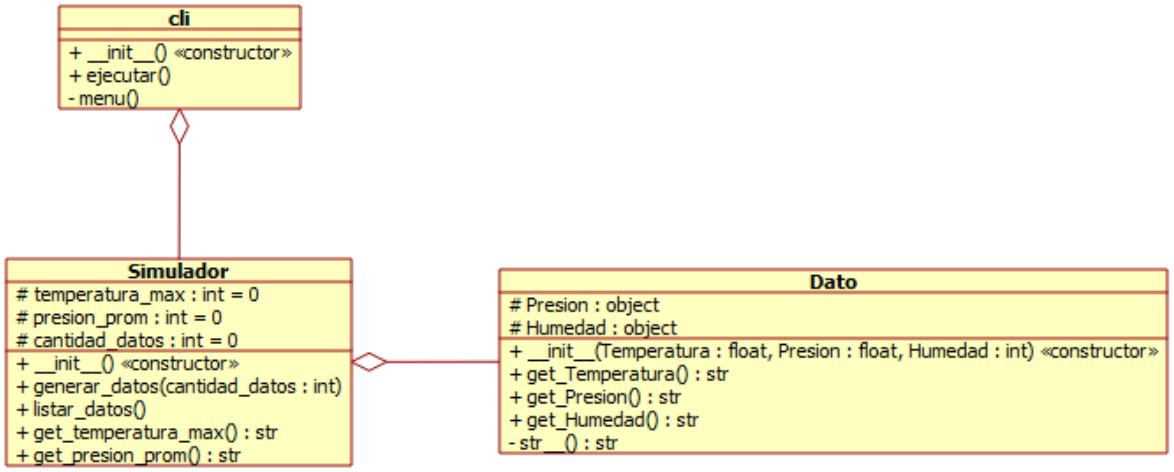
Atributos:

- coordenadas: Un puntero a un arreglo dinámico de objetos Coordenada para almacenar la trayectoria del dron.
- cantidad_de_coordenadas: La capacidad máxima de coordenadas que puede almacenar el dron.
- cantidad_de_coordenadas_actuales: El número actual de coordenadas almacenadas.

Métodos:

- Dron(int cantidad_de_coordenadas): Constructor que inicializa el dron con una capacidad máxima de coordenadas.
- agregar_coordenada(const Coordenada& c): Agrega una nueva coordenada a la trayectoria del dron.
- mostrar_coordenadas(): Imprime en la consola todas las coordenadas del dron.

Ejercicio N° 4



- cli.py: Define la clase cli que administra la interfaz de línea de comandos.
- Dato.py: Define la clase Dato para representar un registro climático con temperatura, presión y humedad.
- Simulador.py: Define la clase Simulador que simula y almacena los datos climáticos.

Clase cli

Atributos:

- _simulador: Una instancia de la clase Simulador que se utiliza para realizar las operaciones relacionadas con los datos climáticos.

Métodos:

- __init__(): Constructor que inicializa un objeto Simulador.
- ejecutar(): Método principal que ejecuta un bucle infinito presentando un menú al usuario y procesando sus opciones.
- menu(): Imprime el menú de opciones disponibles para el usuario.

Clase Dato

Atributos:

- _Temperatura: Temperatura en grados.
- _Presion: Presión atmosférica.
- _Humedad: Humedad relativa.

Métodos:

__init__(): Constructor que inicializa un objeto Dato con los valores de temperatura, presión y humedad.
get_Temperatura(), get_Presion(), get_Humedad(): Métodos para obtener los valores de los atributos.
__str__(): Método especial que define cómo se representa un objeto Dato como una cadena de texto (por ejemplo, al imprimirlo).

Clase Simulador

Atributos:

- _datos: Una lista para almacenar los objetos Dato generados.
- _temperatura_max: La temperatura máxima registrada.
- _presion_prom: La presión promedio.
- _cantidad_datos: La cantidad total de datos generados.

Métodos:

- __init__(): Constructor que inicializa los atributos.
- generar_datos(cantidad_datos): Genera cantidad_datos registros climáticos aleatorios y los agrega a la lista _datos.
- listar_datos(): Imprime en la consola todos los datos almacenados.
- get_temperatura_max(): Calcula y devuelve la temperatura máxima registrada.
- get_presion_prom(): Calcula y devuelve la presión promedio.