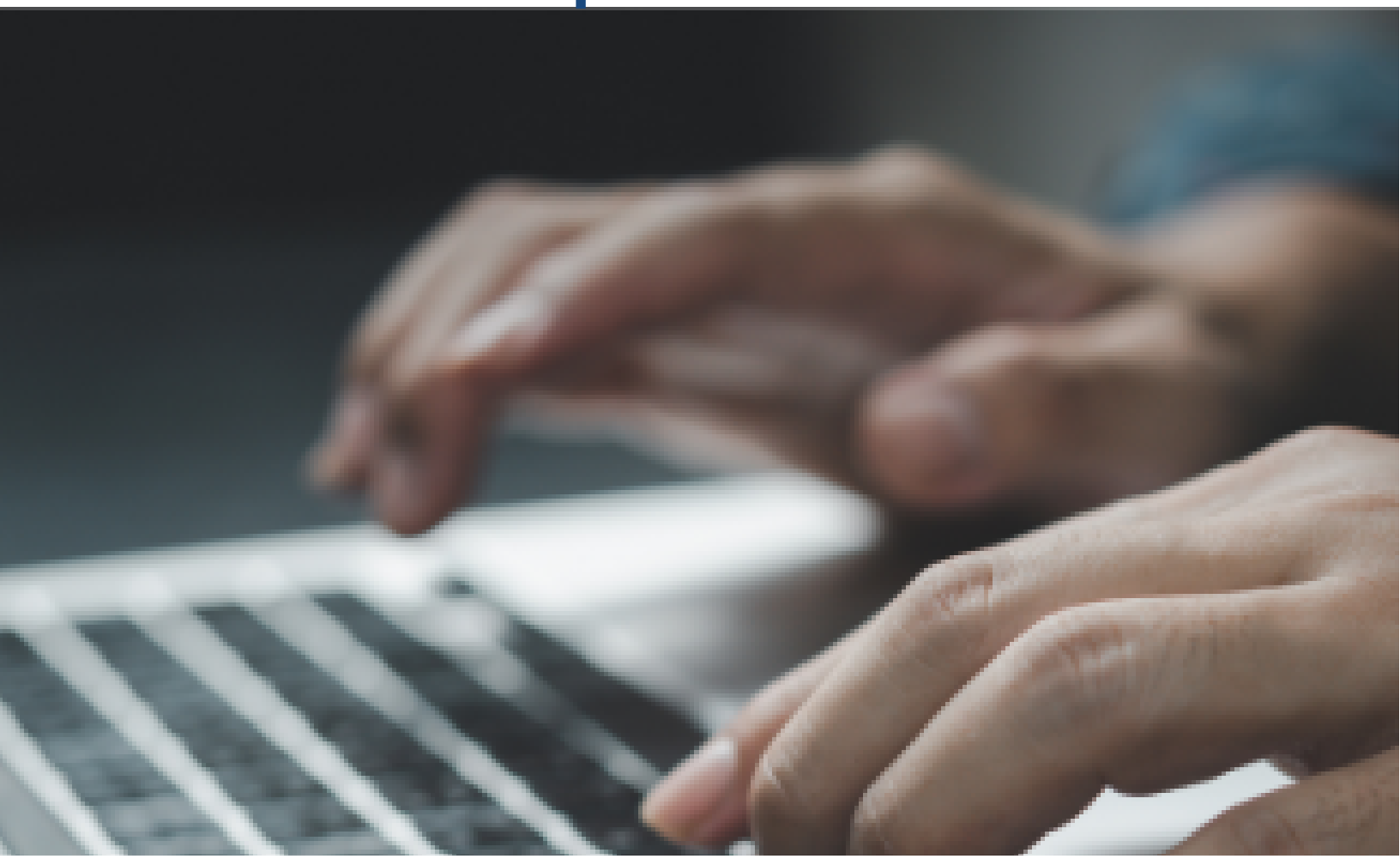
TIC

Proyecto Final:
Sistema de Gestión de Base de Datos de Empleados



En este proyecto final, tendrás la oportunidad de aplicar los conceptos de Programación Orientada a Objetos (POO) utilizando Python para crear un sistema de gestión de la base de datos de empleados de una empresa. Este sistema permitirá administrar la información de los empleados, áreas y jefes, así como realizar operaciones de lectura y escritura en archivos de texto para almacenar y recuperar datos.

Ten en mente que el sistema debe responder preguntas como:

- Quien es el jefe de X empleado?
- En qué área trabaja X empleado?
- Cuántos empleados tienen en el área Y?
- Cuales empleados tienen en el área Y?
- Cuántos empleados tiene a cargo el jefe Z?
- Cuales empleados tiene a cargo el jefe Z?
- Cuál es el salario de X empleado?

Clases a Crear:

1.Clase Empleado:
    - Atributos:
        - Nombre
        - Apellido
        - Edad
        - Salario
        - DNI
        - Fecha de vinculación

Métodos:

- __init__(self, nombre, apellido, edad, salario. dni, fecha_vinculacion): Constructor para inicializar los atributos del empleado.
- obtener_nombre_completo(self): Método para obtener el nombre completo del empleado.
- Otros métodos según sea necesario para la gestión de empleados.

2. Clase Jefe (subclase de Empleado):

- Atributos adicionales:
  - Empleados_a_cargo (lista de empleados bajo su supervisión)
- Métodos:
  - agregar_empleado(self, empleado): Agrega un empleado a la lista de empleados a cargo del jefe.
  - Otros métodos según sea necesario para la gestión de jefes y sus subordinados.

3. Clase Área:

- Atributos:
  - Nombre
  - Descripción
  - Lista de empleados en el área
- Métodos:
  - __init__(self, nombre, descripcion): Constructor para inicializar los atributos del área.
  - agregar_empleado(self, empleado): Agrega un empleado a la lista de empleados en el área.
  - Otros métodos según sea necesario para la gestión de áreas y sus empleados.

Funcionalidades a Implementar:

- **Lectura y Escritura en Archivos de Texto:**
  - Implementa métodos para guardar y cargar la información de empleados, jefes y áreas desde y hacia archivos de texto. Esto permitirá que la información persista entre sesiones del programa.

- **Gestión de Empleados:**
  - Permite agregar, eliminar, modificar y mostrar la información de los empleados en el sistema.

- **Gestión de Jefes:**
  - Permite asignar empleados a jefes, mostrar la lista de empleados a cargo de cada jefe y otras operaciones relacionadas.

- **Gestión de Áreas:**
  - Permite crear, eliminar y modificar áreas, así como asignar empleados a estas áreas.

Requisitos Adicionales:

- Utiliza principios de herencia para compartir atributos y métodos comunes entre las clases.
- Implementa encapsulamiento para proteger los datos y garantizar su integridad.
- Utiliza adecuadamente los métodos especiales de Python (por ejemplo, __str__) para representar objetos como cadenas legibles.
- Asegúrate de manejar posibles errores y excepciones de manera adecuada en tu código.

Nota: Los atributos y métodos listados son los mínimos requeridos para la gestión del proyecto, sin embargo, la dinámica del desarrollo puede indicarnos que hay elementos adicionales que se deben considerar para agregar. Un ejemplo es crear una clase superior que se llame Persona donde se definan todos los atributos generales de los empleados y jefes y que éstos hereden de allí.