SEGUIMIENTO 3

Punto 3:

1. *Responsabilidad Única:* Como tal, todas las clases cumplen un este principio. Por ejemplo: La clase Empleado se encarga únicamente de manejar la información básica de un empleado. Lo mismo sucede, por ejemplo, con la clase Proyecto, que únicamente gestiona lo relacionado con el mismo proyecto y los empleados que hacen parte del mismo, todo se relaciona.

2. *Abierto/Cerrado:* Un ejemplo claro en nuestro sistema es la interfaz de Contribuyente, porque al ser precisamente una interfaz puede permitir añadir mas clases que lo implementen, por lo cual, el código se puede extender.

3. *Sustitución de Liskov:* Un ejemplo son las herencias Gerente/Técnico puesto que, al heredar de la clase Empleado, se pueden utilizar en cualquier contexto que se requiera un empleado.

4. *Segregación de interfaces:* Un ejemplo, y el único, es la interfaz Contribuyente, que solo tiene un método contribuir () y nada más.

5. *Inversión de dependencia:* Utilizando la interfaz que ninguna clase superior dependa de una inferior, puesto que, por ejemplo, alguna contribución relacionada con algún proyecto no dependa de empleado, si no de Contribuyente.

Punto 4:

1. ***Funcionalidad 1*** *Creación de Empleado y Asignación de rol:* Esta funcionalidad permite crear un nuevo y asignarle su rol (Gerente o Técnico). Dependiendo del rol, se llama al método correspondiente para la contribución única. Se involucran las clases Empleado, Gerente y Técnico, Contribuyente.

2. ***Funcionalidad 2*** *Evaluación del desempeño del empleado:* Esta funcionalidad permite evaluar el desempeño de un empleado basado en su rol y sus contribuciones. Se involucran las clases Empleado, Gerente y Técnico, Proyecto.

3. ***Funcionalidad 3*** *Asignación de empleados a un proyecto con distribución de tareas:* Esta funcionalidad permite asignar empleados a un proyecto y distribuir tareas específicas según su rol. Se involucran las clases Empleado, Gerente y Técnico, Proyecto.