COMPROMISOS DEL RETO

1. Aprovechar el tiempo de manera eficaz. No faltar a las horas del reto para poder avanzar.

2. Comunicación en todo momento. Si algo no es de nuestro agrado, se hablará con el grupo para solucionarlo y evitar confrontaciones.

3. El uso de la inteligencia artificial será permitida siempre y cuando se utilice de manera responsable y se supervise.

4. Aceptar críticas constructivas y aportar ideas para mejorar el trabajo.

5. Respetar las decisiones tomadas por consenso.

6. Revisar el trabajo final antes de entregarlo.

Grupo: Mikel, Nora e Irati.

ENUNCIADO RETO PGR 2

Nora Yakoubi, Irati Martinez y Mikel Martinez

El proyecto consiste en desarrollar un programa para **gestionar la información de festivales de música**, incluyendo los festivales, conciertos, grupos musicales y las personas involucradas (staff y cantantes).
Toda la información de los festivales se guardan en un fichero:
"**festival.obj**" almacena los festivales y sus conciertos. y otro fichero para guardar los grupos musicales con el nombre: "**grupos.obj**".

Por cada **festival** queremos saber:

- **NombreFestival** String
- **Cod** → 3 primeras letras del festival + número del festival que se incremente automáticamente
- **Lista de conciertos (**treeMap**,** ordenado por fecha)

Del **concierto** necesitamos saber:

- **IdentificadorC (**Fecha(LocalDate)+3 primera letras del equipoPrincipal**)**
- **GrupoPrincipal** son referencias a objetos de tipo grupo existentes.
- **GrupoInvitado** son referencias a objetos de tipo grupo existentes.

Del **Grupo** queremos almacenar:

- **CodGrupo** autoincrementa desde 01
- **NombGrupo** String
- **Lista de Personas (**HashMap**,** de Staff y Cantantes)

Persona , es una clase abstracta. Independientemente de que sea Staff o Cantante, para toda **Persona** necesitamos manejar los siguientes datos:

- **Nombre** String
- **Apellido** String
- **DNI (**exception**)** String
- **Email (**excepción**)** String
- Además, la clase Persona define el método abstracto **getDescripcion(),** que será implementado por las clases Staff y Cantante aplicando polimorfismo.

Del **Staff** queremos guardar:

- **Tipo**. Puede ser **Manager**, **técnico** de **sonido** o **médico**. Utilizamos una **enumeración** para manejar este dato, controlando que el usuario no pueda introducir ninguna otra opción.
- **Fecha InicioPuesto.** LocalDate
- La clase Staff implementa el método **getDescripcion()**, mostrando su tipo y los años que lleva en el puesto.
  Validación adicional. Evita fechas futuras

De los **Cantantes** necesitamos saber:

- El **códigoCant** → autogenerado incremental
- La clase Cantante implementa el método **getDescripcion()**, mostrando el género musical. **Género**: Puede ser **pop**, **reggaeton** o **rock**. Utilizamos una enumeración para manejar este dato, controlando que el usuario no pueda introducir ninguna otra opción.Este control se realizará mediante una excepción.

El programa cargará los datos desde los ficheros al iniciar la ejecución.
El programa gestiona un menú repetitivo como se especifica a continuación:

Menú principal

1. **Añadir**
    a. **Submenú para seleccionar qué se añade:**
        1. **Añadir grupo**
            - **Solicitar nombre del grupo**
            - **Generar automáticamente CodGrupo**
            - **Añadir personas al grupo (Staff y Cantantes) con validaciones: DNI, Email, tipo/género, fechas correctas**
            - **Guardar grupo en grupos.obj y en la estructura de datos interna**

        2. **Añadir festival**
            - **Solicitar nombre del festival, si el nombre tiene menos de 3 letras le salta un error y le vuelve a mandar el menú añadir.**
            - **Generar automáticamente CodFestival**
            - **Añadir conciertos al festival, vinculando grupos existentes**
            - **Guardar festival en festival.obj y en la estructura interna**

        3. **Añadir concierto al festival**
            - **Elegir uno de los festivales disponibles**
            - **Solicitar fecha del concierto**
            - **Comprobar que no exista ya un concierto en esa fecha en ese festival**
            - **Mostrar los grupos disponibles**
            - **Seleccionar grupo principal y grupo invitado (distintos)**
            - **Generar automáticamente CodConcierto (Fecha + 3 primeras letras del grupo principal)**
            - **Guardar los cambios en *festival.obj* y en la estructura interna**

2. **Listar Grupos → ordenados por nombre**
3. **Listar Festivales → ordenados por nombre**
4. **Listar Conciertos → ordenados por fecha, pedir al usuario una fecha y sacar los conciertos realizados a partir de esa fecha hasta dia de hoy**
    (un concierto diario)

5. **Eliminar festival**
    a. **Solicitar CodFestival**
    b. **Comprobar existencia y eliminar**
    c. **Mostrar mensaje de error si no existe**
6. **Listar grupos detallado**
    a. **Sacar el código y el nombre del grupo, y mostrar todas las personas asociadas (cantantes y staff), listando cada uno de sus atributos.**
7. **Consultar Staff por DNI**
    a. **Mostrar grupo, puesto y años en el puesto**
    b. **Mostrar mensaje de error si el DNI no existe**
8. **Modificar información**
    a. **Modificar datos de cantantes (nombre, género)**
    b. **Modificar datos de staff (nombre, tipo)**
    c. **Validaciones: revisar existencia y evitar duplicados**
9. **Salir del programa**