Examen Práctico
Convocatoria de Junio de 2020

Nombre:_____

DNI:_____ **Grupo:**_____

Preparación del examen

- Crea la carpeta "examen" en el Escritorio.
- Arranca Eclipse y utiliza la carpeta "examen" como espacio de trabajo.
- Crea un proyecto Java que incluya tu grupo, apellidos y nombre, siguiendo la plantilla: **Examen-GXX-*Apellidos-Nombre***. Por ejemplo "Examen-G11-MartinezLopez-Juan".
- Al acabar el examen, comprime la carpeta con el proyecto (no el espacio de trabajo). El nombre del fichero comprimido debe coincidir con el nombre del proyecto con extensión .rar, .zip o la correspondiente al compresor utilizado.
- Sube el fichero a la tarea del **Aula Virtual** "Examen Junio 2020".

Previo. Gestión de fechas.

- Para representar las fechas utilizaremos la clase **`java.time.LocalDate`**, que implementa una fecha (día y hora) sin información de la zona horaria de acuerdo al sistema de calendario ISO-8601, en el formato (por defecto) de `año-mes-día`, `2020-06-16`.
- La clase `LocalDate` no dispone de constructores. Para crear objetos se utiliza el método de clase *`of`*, que recibe como parámetro el año, mes (como un entero o un valor del enumerado `Month`) y día. Por ejemplo:

```
LocalDate fecha = LocalDateTime.of(2020, 06, 16);
LocalDate fecha = LocalDateTime.of(2020, Month.JUNE, 16);
```

- La clase dispone del método de clase `now()` que devuelve la fecha actual.
- Las fechas son **comparables**, aun así, también están disponibles los métodos `isAfter`, `isBefore` e `isEqual`.
- Los objetos `LocalDate` son **inmutables**, por lo que cualquier operación de cambio de fecha devuelve un nuevo objeto con el resultado. Por ejemplo, la clase ofrece el método `plusDays (int numDias)` que devuelve una fecha a la que se le suma el número de días que se pasa como parámetro.

El objetivo del ejercicio es el desarrollo de una aplicación para la gestión de las sesiones de entrenamiento de un gimnasio.

1. (3,5 puntos) Funcionalidad básica

En un gimnasio se organizan sesiones de entrenamiento de diferentes ***actividades*** que pueden ser: yoga, pilates, aeroyoga, spinning, crossfit y zumba.

Los ***usuarios*** que pueden participar en las actividades del gimnasio se caracterizan por:

- Nombre del usuario, que no puede cambiar.
- Niveles. Mapa que registra para cada tipo de actividad el número de sesiones que ha realizado de dicha actividad. Se considera que este número representa el nivel alcanzado por el usuario en cada actividad. Nótese que en la construcción del objeto el mapa debe quedar completo, esto es, debe tener una entrada por cada actividad. Inicialmente todas las actividades tendrán un nivel 0.
- Fecha fin de sanción. Los usuarios que son sancionados tendrán un valor en esta propiedad que establece el fin de la sanción. Esta propiedad se inicializa con la fecha actual y no ofrece un método de modificación.
- Sancionado. Valor booleano que indica si un usuario está sancionado. Un usuario estará sancionado si la fecha actual es anterior a la fecha de fin de la sanción.

La funcionalidad que ofrece la clase que representa a los usuarios, además de la consulta de sus propiedades, es:

- Operaciones para incrementar o decrementar en una unidad el nivel de una actividad, esto es, el número de sesiones realizadas de la actividad que se pasa como parámetro.
- Sancionar. Recibe como parámetro el número de días que va a estar sancionado a partir de la fecha actual.

El constructor recibe como parámetro solo el nombre del usuario.

Los usuarios podrán inscribirse y borrarse de las sesiones que organiza el gimnasio. Las propiedades que caracterizan una ***sesión*** son:

- La actividad que se va a impartir durante la sesión.
- Fecha de la actividad.
- Descripción, como, por ejemplo, “9:00, duración 45 minutos”.
- Cupo: número máximo de usuarios admisibles en la sesión.
- Colección de usuarios inscritos en la sesión.
- Número de usuarios inscritos.
- Completa. Una sesión está completa si el número de usuarios inscritos es igual al cupo máximo establecido.

El constructor recibe como parámetro la actividad, la fecha, el cupo y la descripción. Inicialmente la colección de usuarios está vacía. Una vez construido el objeto solo podrá modificarse la fecha y la colección de usuarios a través de los métodos que gestionan las inscripciones. También pueden construirse las sesiones omitiendo el cupo de usuarios. Se asume en este caso que tendrá un cupo de 15.

La funcionalidad que ofrece el tipo de datos que representa una sesión es:

- Inscribir un usuario. Esta operación tiene como requisitos: 1) que la fecha actual sea al menos un día antes de la fecha de inicio, 2) que no esté completa la sesión y 3) que el usuario no esté sancionado. En el caso de que se cumplan estos requisitos el usuario se añade a la colección de usuarios y se devuelve true. En caso contrario, se devolverá false para indicar que no se ha podido realizar la inscripción.
- Consultar si un usuario está inscrito. Esta operación recibe como parámetro un usuario y retorna un valor booleano que indica si está, o no, inscrito en la sesión.

- Borrar un usuario. Esto es, cancelar la inscripción de un usuario a la sesión. Los requisitos de esta operación son: 1) que el usuario esté inscrito previamente y 2) que la fecha actual sea al menos un día antes de la fecha de inicio. Si se cumplen estas condiciones se eliminará el usuario de los usuarios inscritos y se aplicarán las *acciones* que correspondan por darse de baja. Las acciones dependen de cada tipo de sesión como se verá en los ejercicios siguientes (2 y 3).
- Confirmar. Para confirmar una sesión la fecha actual tendrá que ser igual o posterior a la fecha de inicio. Al confirmar una sesión se asume que la sesión se ha realizado con la participación de todos los usuarios inscritos y, por tanto, se incrementa en una unidad el nivel de la actividad a cada usuario.

2. (1,25 punto) Tipos de sesiones: personal y nivel.

Los tipos de sesiones que pueden organizarse son sesiones de entrenamiento personal, sesiones de nivel y sesiones covid (ejercicio 3).

Una ***sesión personal*** es un tipo de sesión en el que el cupo es 1. La propiedad específica de este tipo de sesiones es el nombre del monitor que la imparte, que puede cambiar. En caso de que un usuario se borre, tendrá una sanción de dos días. La sanción no tendrá ningún efecto en las inscripciones que ya se hayan realizado.

Una ***sesión de nivel*** es un tipo de sesión en la que se establece como requisito haber alcanzado un nivel mínimo en la actividad de la sesión. Por ejemplo, para inscribirse en “aeroyoga” con descripción “9:00. Duración hora y media. Nivel avanzado.” previamente se debe tener un nivel 2 en “aeroyoga”. Por tanto, en la construcción debe establecer el nivel mínimo necesario para poderse inscribir en la sesión. Este valor puede variar.

El usuario que se borre de la sesión de nivel será sancionado descontando un nivel de la actividad de la sesión.

3. (1,5 puntos) Diseño por Contrato. Sesión Covid.

Una ***sesión covid*** es un tipo de sesión que se organiza en grupos pequeños con un número cerrado de usuarios que se establecen en la construcción (argumento variable). El número de usuarios define el cupo, que en ningún caso puede superar 5 usuarios.

Además de los usuarios, en la construcción de una sesión covid se puede establecer como parámetro (es opcional) una sesión covid previa, esto es, una sesión covid con fecha de inicio anterior a la fecha de inicio de la sesión que se está configurando. En el caso de establecerse una sesión previa, es obligatorio que todos los usuarios que se van a inscribir en la sesión actual deben haber participado de la sesión covid previa, de no ser así, la sesión no puede organizarse. De igual forma, si se ha establecido una sesión previa, cuando un usuario se inscribe en la actividad (por la baja de otro participante que deja el hueco) debe haber participado en esta sesión previa. Si no se ha establecido una sesión previa, cualquier usuario podrá inscribirse si se cumplen los requisitos generales establecidos para una sesión.

La cancelación de la inscripción de un usuario no tiene penalización, esto es, no se aplica ninguna acción si un usuario se da de baja en una sesión covid.

Implementa la funcionalidad de la sesión covid aplicando el ***control de precondiciones del Diseño por Contrato*** en la implementación de los constructores.

4. (1,75 puntos) Métodos de la clase Object

Programa el método `clone` en la jerarquía de sesiones siguiendo las recomendaciones vistas en clase. El método ofrecerá una copia que no tendrá usuarios inscritos.

Utiliza el método `clone` para programar un método en la clase `Sesion` (*getSesionesPeriodicas*) que permita configurar una lista de sesiones periódicas. Este método recibe como parámetro el número de días entre sesiones y el número de repeticiones, y devuelve una lista de sesiones. Por ejemplo, si aplicamos el método a una sesión de "pilates" para el 22 de junio (sesión inicial), con 7 días se separación y 8 repeticiones, el método devuelve una lista de 9 sesiones: la sesión inicial (objeto receptor) y las 8 creadas a partir de ella con 7 días de separación entre ellas.

Redefine el método `toString` en todas las clases implementadas para poder ver los resultados del programa.

5. (0,75 puntos) Programa

Implementa un programa con la siguiente funcionalidad:

- Crea los usuarios con nombres "Paqui", "Vicente", "Beatriz", "Fina" y "Juan" y añade todos los usuarios a una lista de nombre `usuarios`.
- Crea una lista de nombre `sesiones` donde vamos a registrar todas las sesiones que se planifiquen.
- Declara una variable `sesion1` y asígnale una *sesión personal* de "yoga" con fecha 16/06/2020, descripción "9:00. Hatha Yoga" y monitor "Manu".
- Genera una lista de sesiones periódicas a partir de la `sesion1` cada 2 días y con 2 repeticiones, y añádelas a la colección de sesiones.
- Declara una variable `sesion2` y asígnale una *sesión de nivel* de "pilates" con fecha 17/06/2020, descripción "18:00. Pilates - Nivel Principiante", cupo = 3 y requisito de nivel = 0.
- Declara una variable `sesion3` y asígnale una *sesión covid* de "zumba" con fecha 17/06/2020, descripción "21:00. Zumba - Grupo Seguro" y con los usuarios Vicente, Beatriz, Fina y Juan.
- Declara una variable `sesion4` y asígnale una *sesión covid* de "zumba" con fecha 18/06/2020, descripción "21:00. Zumba - Grupo Seguro", con los usuarios Vicente, Beatriz y Fina, que tenga como sesión previa la `sesion3`.
- Añade `sesion2`, `sesion3` y `sesion4` a la lista de sesiones.
- Recorre la colección de sesiones e inscribe a todos los usuarios en todas las sesiones. Para cada sesión, muestra en la consola los usuarios que se han inscrito.
- Borra a Fina de la `sesion4` e intenta inscribir a Paqui y a Juan en esta sesión.
- Muestra en la consola los usuarios inscritos en la `sesion4`. Nota: Deben ser {Vicente, Beatriz, Juan}
- Borra a Beatriz de la `sesion2` e intenta inscribir a Fina en esta misma sesión.
- Muestra en la consola los usuarios inscritos en la `sesion2`. Nota: Deben ser {Vicente, Paqui, Fina}
- Recorre la colección de sesiones y si es una sesión personal cambia el monitor por "Tania".

6. (1,25 puntos) Programación funcional y procesamiento basado en el modelo stream.

a) Modifica la clase que representa una sesión de nivel para que disponga de un nuevo constructor que reciba como parámetro, además del nivel mínimo para el tipo de actividad, un ***predicado*** que evalúe los niveles alcanzados en cada actividad por el usuario. Este predicado representa un nuevo requisito para la inscripción de un usuario a una sesión. Por ejemplo, para inscribirse en "aeroyoga" con descripción "nivel avanzado. 9:00. Duración hora y media." previamente se debe tener un nivel 2 en la propia actividad ("aeroyoga") y, además, nivel 1 en "pilates" y nivel 2 de "yoga". Por tanto, a la hora de inscribir un usuario en una sesión habrá que comprobar que cumple el requisito de nivel mínimo en la actividad y también, si se ha establecido el predicado, que cumple los requisitos sobre los niveles alcanzados en otras actividades por el usuario.

Añade al final del programa principal:

- Confirmar la sesión `sesion1`.
- Declara la variable `sesion5` y asígnale una nueva sesión de nivel de la actividad "aeroyoga", para el 17/Junio/2020, descripción "19:00. Nivel principiante", con cupo = 5, con los requisitos de nivel en esta actividad 0 y recibe como parámetro un predicado que evalúa si el usuario tiene al menos un nivel 1 en "yoga".
- Recorre los usuarios e inscribe a todos los usuarios en la `sesion5`.
- Muestra en la consola los usuarios inscritos en la `sesion5`. NOTA: Sólo debe estar Paqui que es la única que cumple los requisitos.

b) Añade al programa principal las siguientes consultas utilizando el *procesamiento basado en streams* sobre la colección de sesiones:

- Muestra en la consola todas las sesiones de "yoga" que todavía no se han realizado ordenadas por fecha.
- Consulta si existe alguna sesión de "pilates" en la que se haya inscrito Juan. Muestra el resultado en la consola.
- Muestra en la consola la descripción de las sesiones programadas para el 18/Junio/2020 ordenadas alfabéticamente (orden natural).