

Examen 1ª evaluación

La empresa *KonohaTech* te ha contratado para desarrollar un software de gestión de equipos Ninjas. Se trata de una aplicación Java formada por una clase principal **Main**, otras llamadas **Aldea**, **Equipo**, **Sensei**, **Ninja** y **Persona**. Además de dos Enum **Rango** y **Chakra**.

Clase Main

Clase principal con función main. Encargada de interactuar con el usuario, mostrar el menú principal, dar feedback y/o mensajes de error, etc. Puedes implementar las funciones que consideres oportunas en cualquiera de las clases.

El programa **pedirá los datos necesarios para crear una Aldea**. Si son válidos, creará la aldea y mostrará el menú principal para permitir actuar sobre la aldea. Tras cada acción se volverá a mostrar el menú.

1. Crear Equipo.
2. Registrar Ninja en el equipo.
3. Mostrar el número de ninjas totales en la Aldea.
4. Mostrar información de un equipo.
5. Mostrar información de todos los equipos.
6. Mostrar información de toda la Aldea.
7. Calcular estadísticas de toda la Aldea.
8. Cambiar ninja de un equipo.
9. Eliminar ninja de la aldea.
10. Salir de la aplicación.

Clase Aldea

La clase Aldea tiene los siguientes atributos

- Nombre de la Aldea
- Código: (Cinco letras mayúsculas y 5 dígitos)
- Fecha de creación (LocalDate)
- Kage (tipo sensei con rango Kage)
- Máximo de equipos (int)

- Equipos: Array de tipo Equipo

Cuando se crea una aldea es **obligatorio que tenga un código en el formato correcto, un nombre, una fecha de fundación y un número máximo de equipos y un Kage**. Se deben de crear los **constructores y propiedades necesarias** con la visibilidad adecuada.

Además de los constructores necesarios, deberá modificar el **toString()** para mostrar toda la información de esa persona. Las **propiedades deben de tener la visibilidad adecuada**.

Se deben de crear al menos los siguientes métodos:

- **registrarEquipo**: el sensei no puede estar en otro equipo y no se puede exceder el máximo de equipos de la aldea.
- **buscaEquipoPorCodigo(String codigo)**: obtener un equipo por el código del equipo
- **mostrarInformacionTodosEquipos**: mostrar toda la información de todos los equipos.
- **calcularNinjasAldea**: calcula el número total de ninjas en la aldea, incluyendo ninjas, senseis y kage.
- **calcularEstadisticasAldea**: mostrar la información promedio de Estrategia, liderazgo, ataque y defensa de todos los equipos.

Se admiten y se evaluarán mejoras sustanciales en el programa o en la complejidad. Así como comprobaciones o validaciones.

Clase Equipo

La clase equipo tiene como datos asociados:

- Nombre del equipo
- Código: (Dos letras mayúsculas y 7 dígitos)
- Fecha de creación (LocalDate)
- Sensei (tipo Sensei)
- Máximo de ninjas (int)
- Array de Ninjas (array de tipo Ninja)

Cuando se crea un equipo es **obligatorio que tenga un código en el formato correcto, un nombre, una fecha de fundación y un número máximo de ninjas sin contar al sensei** (que no podrán cambiar nunca). Se deben de crear los **constructores y propiedades necesarias** con la visibilidad adecuada. Se deberá modificar el **toString()** para mostrar toda la información de ese equipo. Las **propiedades deben de tener la visibilidad adecuada**.

Se deben de crear al menos los siguientes métodos:

- **registrarNinja**: comprobación de que no esté ya registrado por el DNI y codigoNinja, se debe de comprobar que no exceda del máximo de ninjas y que sea mayor de edad, además asignar el N° de contrato al ninja (propiedad de clase Persona) con el siguiente formato CONT-CodigoEquipo.
- **eliminarNinjaEquipo**: debe eliminar el ninja del equipo, y borrar el N° de contrato al ninja. Para buscar al ninja se debe de hacer por DNI. Este Ninja quedara en una colección de ninjas sin equipo.
- **mostrarInformacionNinjas**: mostrar toda la información de todos los ninjas registrados en el equipo.
- **calcularEstadisticasEquipo**: mostrar la información promedio de Estrategia, liderazgo, ataque y defensa del equipo.

Clase Sensei

La clase Sensei **hereda de Ninja** y deberá tener los siguientes atributos:

1. codSensei (String, generación aleatoria de un código)
2. Estrategia (double entre 0 -100)
3. Liderazgo (double 0 -100)

Además de los constructores necesarios, deberá modificar el **toString()** para mostrar toda la información de ese Sensei. Las **propiedades deben de tener la visibilidad adecuada**.

Se admiten y se evaluarán mejoras sustanciales en el programa o en la complejidad. Así como comprobaciones o validaciones.

Clase Ninja

La clase Ninja **hereda de Persona** y deberá tener los siguientes atributos:

1. Chakra (Enum)
2. Rango (Enum)
3. idNinja (int autoincremental)
4. TecnicaSecreta (String)
5. FechaAlta (LocalDate)
6. Ataque (double 0 -100)
7. Defensa (double 0 -100)

Además de los constructores necesarios, deberá modificar el **toString()** para mostrar toda la información del ninja. Las **propiedades deben de tener la visibilidad adecuada**.

Se admiten y se evaluarán mejoras sustanciales en el programa o en la complejidad. Así como comprobaciones o validaciones.

Enum Rango

Se debe de crear un ENUM con los rangos de los ninjas:

- GENIN
- CHUNIN
- JONIN
- AMBU
- KAGE

Enum Chakra

Se debe de crear un ENUM con los rangos de los ninjas:

- TIERRA
- FUEGO
- AIRE
- AGUA
- RAYO

Clase Abstract Persona

La clase Persona deberá tener los siguientes atributos:

1. Nombre (String)
2. FechaNacimiento (LocalDate)
3. DNI (debe de tener el formato correcto 8 dígitos y una letra mayúscula) (String)
4. Dirección (String)
5. Nº de contrato (String)

Se admiten y se evaluarán mejoras sustanciales en el programa o en la complejidad. Así como comprobaciones o validaciones.