Evaluación Diagnóstica de Competencias en Java

Vamos a realizar una breve prueba diagnóstica. Es importante que tengáis en cuenta lo siguiente:

- Objetivo: La prueba es solo para ver el nivel general del grupo en Java.
- Sin nota: No se tiene en cuenta para la calificación de la asignatura.
- Material: Podéis revisar toda la documentación y apuntes subidos en Teams.
- Honestidad: Os pido que seáis justos y no utilicéis IA. El único fin de esto es saber qué nivel tenéis realmente para poder adaptar mejor el contenido.
- Entrega: Cuando finalicéis, enviad vuestros ejercicios por correo electrónico.
- La corrección/solución la tendréis en un video, para que podáis ver un poco como se resuelve.

1. Elaboración de las funciones de dibujado del triángulo rectángulo

Se pide a continuación la siguiente función:

public static void leftTriangle (int nrows, String simbol)

- Se quiere que se represente el siguiente de altura nrows
- Ej: leftTriangle(5, “* “)

```
*
* *
* * *
* * * *
* * * * *
```

2. En este ejercicio, vas a diseñar un "Generador de Identificadores" que cree un identificador a partir del nombre completo de una persona y su fecha de nacimiento

Pasos a seguir:

- El programa pedirá al usuario que ingrese su nombre completo en una sola línea, incluyendo su nombre y dos apellidos
- Luego, el sistema pedirá la fecha de nacimiento en formato DD/MM/AAAA

Reglas para construir el identificador. (Consta de 3 partes separados por "-")

1. Toma la primera letra del nombre y la primera letra de cada apellido en mayúsculas y juntalas (1º parte del id)
2. Suma el dia el mes y el año de la fecha que has indicado (2º parte de id)
3. Agrega los últimos dos dígitos del año de nacimiento y concatena la cantidad total de letras del nombre completo, excluyendo espacios

Ejemplo:

1. **Datos de entrada:**
    - **Nombre completo:** Ana Torres Sánchez
    - **Fecha de nacimiento:** 25/11/1983
2. **Generación del identificador:**
    1. **1ª parte (Iniciales):**
        - **A**na + **T**orres + **S**ánchez = ATS
    2. **2ª parte (Suma de fecha):**
        - Día (25) + Mes (11) + Año (1983)
        - $25 + 11 + 1983 = 2019$
        - Resultado: 2019
    3. **3ª parte (Últimos 2 dígitos del año):**
        - Año: 19**83**
        - Resultado: 83
    4. **4ª parte (Total de letras):**
        - Ana (3) + Torres (6) + Sánchez (7)
        - $3 + 6 + 7 = 16$
        - Resultado: 16

3. Identificador final (separado por guiones): ATS-2019-8316

3. Escribe un programa en Java que realice las siguientes acciones:

a. Crea un array bidimensional de tamaño 3x3 con valores enteros (asignale valores iniciales).
b. Calcula la suma de los elementos de cada fila.
c. Imprime el resultado de la suma de cada fila en el formato:

La suma de la fila 1 es: <suma>
La suma de la fila 2 es: <suma>
La suma de la fila 3 es: <suma>

4. En probabilidad, hay una ley denominada “Ley de los Grandes Números” que indica que si repetimos muchas veces un mismo experimento, la frecuencia de que suceda un cierto evento tiende a ser una constante.

Para comprobar esta ley computacionalmente, se pide que:

A. Se generen 100.000 muestras aleatorias de números enteros entre [0-10), guardados en un Array dinámico.
B. Se utilice un diccionario (HashMap) para albergar una pareja de valores, donde la clave sea los posibles número aleatorios que puedan salir y cuyo valor sea la frecuencia de aparición de esos valores.
C. Como se quiere un porcentaje, sera necesario
    a. Dividir esos valores de frecuencia obtenidas entre la cantidad total de números generados
    b. Multiplicarlos x 100 para tener el %

5. Los arrays de objetos son una estructura muy común dentro de la programación. En Java esto puede traducirse como un ArrayList que contiene HashMaps.

Para este ejercicio, se te va a dar la representación de esa estructura de arrays de objetos aplicado a un sondeo anónimo donde 3 personas han valorado del 1-5 cinco preguntas que se le han hecho.

[

{P1=5,P2=3,P3=4,P4=1,P5=2},

{P1=3,P2=3,P3=3,P4=3,P5=3},

{P1=1,P2=5,P3=5,P4=4,P5=3}

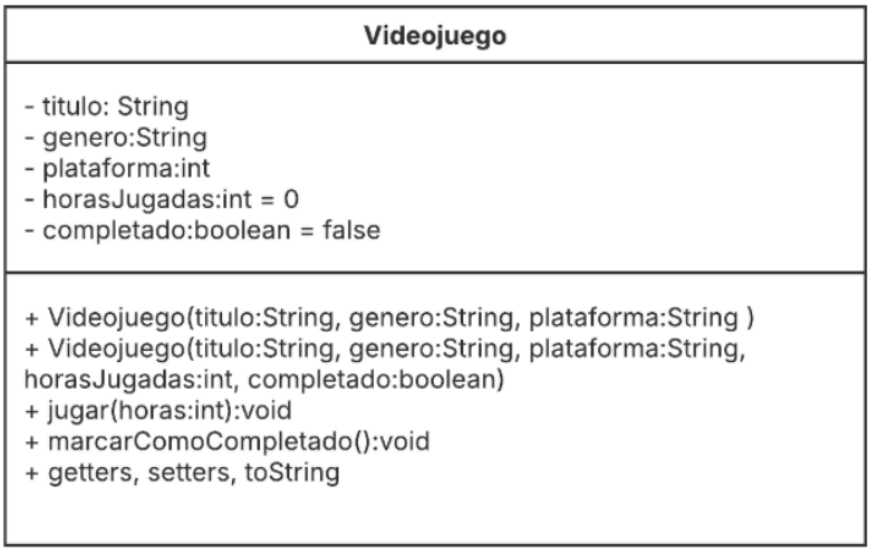
]

Lo que se quiere conocer una vez cargada la estructura en Java es:

A. La valoración máxima y mínima de cada persona.
B. La valoración en orden ascendente de cada persona.
C. La valoración media de las 3 personas (sumar los valores de cada persona y dividirlo entre la cantidad de preguntas)

Nota: Por cada persona se mostrará por pantalla toda la información pedida en el orden indicado.

6. Implementa en Java una clase llamada Videojuego que modele la información básica de un videojuego.

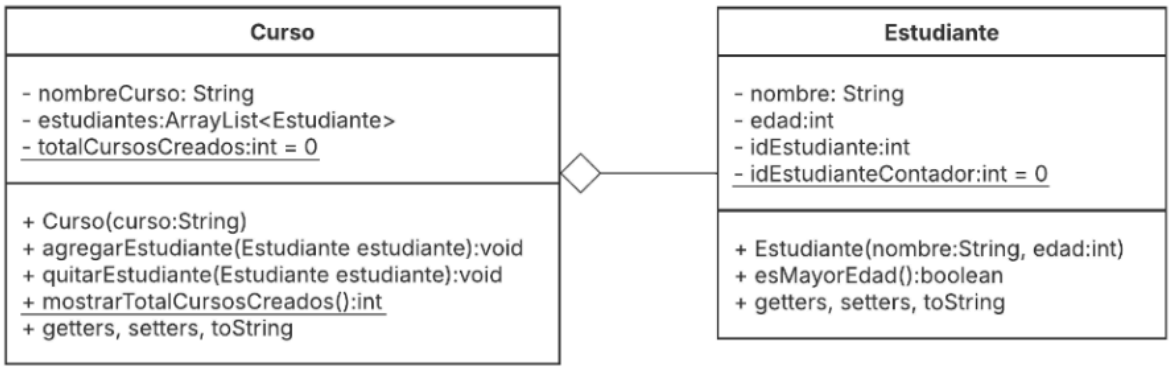


- **jugar(int horas):** Incrementa el número de horas jugadas del videojuego con las horas proporcionadas. Si las horas jugadas superan 100, automáticamente establece completado en true.
- **marcarComoCompletado():** Marca explícitamente el videojuego como completado, independientemente de las horas jugadas.
- **Getters, setters, toString.**

En una clase Main:

- Crea dos objetos Videojuego usando cada constructor.
- Usa los métodos para registrar horas jugadas y marcar alguno como completado.
- Finalmente, muestra toda la información de ambos objetos.

7. Implementa en Java dos clases relacionadas mediante composición: Estudiante y Curso.



Clase Estudiante

Métodos:

- El constructor inicializa el nombre y la edad. El id se asigna automáticamente de forma incremental.
- Un método llamado esMayorEdad() que retorne un booleano indicando si el estudiante es mayor de edad (18 años o más).
- Getters, setters, toString.

Clase Curso

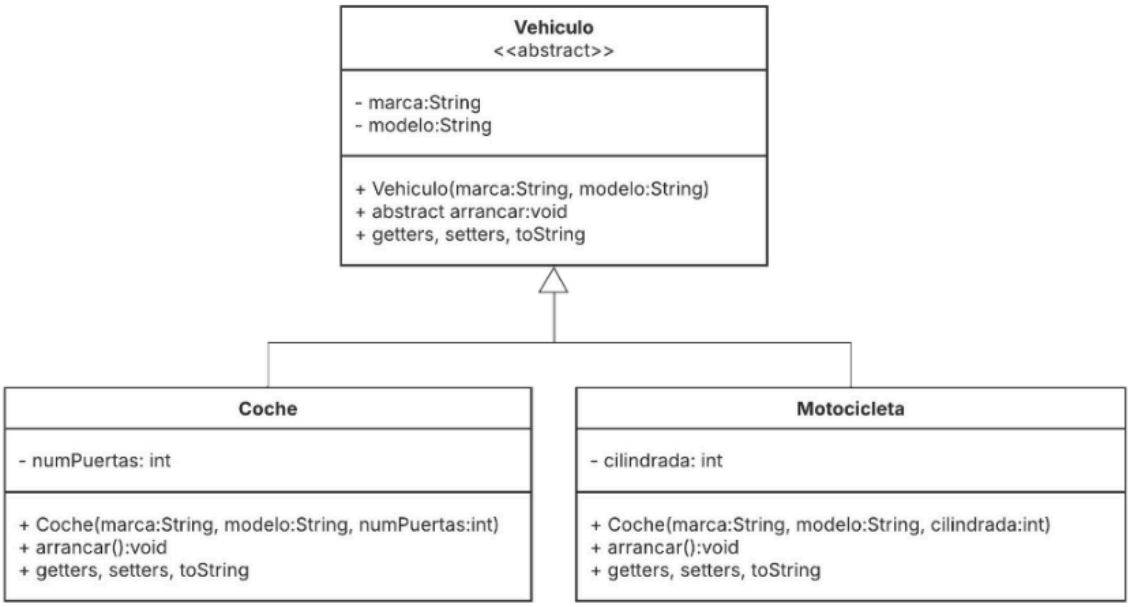
Métodos:

- Un constructor que inicialice el nombre del curso y cree la lista vacía.
- agregarEstudiante(Estudiante estudiante): añade un estudiante al curso.
- quitarEstudiante(int idEstudiante): elimina un estudiante del curso según su idEstudiante.
- mostrarTotalCursosCreados(): muestra la cantidad total de cursos creados.
- Getters, setters, toString.

Funcionamiento:

1. Crea dos cursos diferentes.
2. Añade al menos dos estudiantes a cada curso.
3. Usa el método quitarEstudiante para eliminar a un estudiante del primer curso.
4. Muestra por pantalla los estudiantes de ambos cursos.
5. Usa el método estático para mostrar cuántos cursos se han creado.
6. Finalmente, comprueba con el método adicional esMayorEdad() si los estudiantes son mayores de edad, mostrando claramente el resultado.

8. Implementa en Java una jerarquía de clases con herencia: Vehículo, Coche, Motocicleta



Clase Vehículo

Métodos:

- Constructor con atributos por parámetro.
- Getters y setters, toString
- Método arrancar(): imprime un mensaje indicando que el vehículo está arrancando con sus atributos

Clase Coche

Métodos:

- Constructor con atributos por parámetro.
- Getters, setters, toString
- Método arrancar(): imprime un mensaje indicando que el coche está arrancando con sus atributos.

Clase Motocicleta

Métodos:

(El texto para los métodos de Motocicleta no está completo en la última imagen, pero sigue el mismo patrón que Coche)

- Constructor con atributos por parámetro.
- Getters, setters, toString

- Método arrancar(): imprime un mensaje indicando que el vehículo está arrancando con sus atributos

Clase Main

Realiza las siguientes acciones:

1. Crea una lista (ArrayList<Vehiculo>) y añade a ella al menos dos objetos de tipo Coche y dos objetos de tipo Motocicleta.
2. Recorre la lista y ejecutar el método arrancar() para comprobar el comportamiento polimórfico.
3. Muestra la información completa de cada vehículo usando su método toString().