Buscador de Juegos

Integrantes :

- Ezequiel Campos Martín
- Carlos Vela Buzón
- Ruben Daniel Ternero Molina
- Sergio Pavón

Índice

Idea: **2**
Código: 2
Caso ordenado por nombre **6**
Caso ordenado por notas **8**
Caso ordenado por año de lanzamiento **9**
Caso ordenado por género **11**
Caso ordenado por Consolas **12**
Caso ordenado por desarrolladora **13**

Idea:

Nuestra idea fue crear un buscador sobre videojuegos con varios criterios de búsqueda. Para la búsqueda de dichos juegos usamos la página web Metacritic, sacando de ahí toda la información disponible. Nuestros videojuegos contarán con los siguientes datos:

Nombre: El nombre del susodicho juego.
Género: El género al que pertenezca el juego. En este caso hemos decidido limitar los géneros a: Acción, Aventura, Deportivo, Plataformas, Puzzle, RPG, Shooter, Sigilo, Simulación, Terror.
Nota: La nota media de todas las reviews, la cual ya está conseguida gracias a la página de Metacritic.
El año de lanzamiento: En este caso solo contamos los años de lanzamiento de cada juego.
Consola: La consola para la que salió a la venta dicho juego. En este caso puede estar formada por varias consolas.
Desarrollador: Cada juego contará con un desarrollador.

En base a estas características de cada juego, empezaremos a explicar el código. Decidimos crear un Array de Juegos con un total de 90 objetos, se podrían agregar los necesarios para el futuro, pero lo hemos visto innecesario para esta prueba el añadir más. Para la búsqueda presentamos un total de siete opciones.

Código:

Empezaremos creando la clase Juego para poder construir los objetos de estos en el main.

```
package pruebas;

public class Juego {
  String nombre = "";
  String genero = "";
  int nota;
  int anyoLanzamiento;
  String consola = "";
  String desarollador = "";

  public String getNombre() {
    return nombre;
  }
   public void setNombre(String nombre) {
    this.nombre = nombre;
  }
```

```
  public String getGenero() {
    return genero;
  }

  public void setGenero(String genero) {
    this.genero = genero;
  }

  public int getNota() {
    return nota;
  }

  public void setNota(int nota) {
    this.nota = nota;
  }

  public int getAnyoLanzamiento() {
    return anyoLanzamiento;
  }

  public void setAnyoLanzamiento(int anyoLanzamiento) {
    this.anyoLanzamiento = anyoLanzamiento;
  }

  public String getConsola() {
    return consola;
  }
  public void setConsola(String consola) {
    this.consola = consola;
  }

  public String getDesarollador() {
    return desarollador;
  }
  public void setDesarollador(String desarollador) {
    this.desarollador = desarollador;
  }

  public Juego(String nombre, String genero, int nota, int anyoLanzamiento, String consola,
String desarollador) {
    this.nombre = nombre;
    this.genero = genero;
    this.nota = nota;
    this.anyoLanzamiento = anyoLanzamiento;
    this.consola = consola;
    this.desarollador = desarollador;
  }
```

```
}
```

En dicha clase tenemos nuestro constructor con los atributos futuros del juego, no necesita mucha mas explicación. A continuación, el main:

```
package pruebas;

import java.util.Arrays;
import java.util.Comparator;
import java.util.Scanner;

public class ProyectoBuscador {

	public static void main(String[] args) {
		Scanner teclado = new Scanner(System.in);


		// Aqui creamos los numeros necesarios para introducir las opciones
		int anyoLanzamiento;
		int numeroOperacion;
		int numeroOperacionAnyo;
		boolean confirmacion = false;
```

Para poder usar opciones futuras en los switch, hemos debido de crear algunas variables primero. Luego podremos crear el array con todos los objetos necesarios.

```
// Aqui hemos creado el array de los juegos con sus caracteristicas, hemos considerado que 90 son mas que necesarios, aunque se podría introducir mas de ser necesario
		Juego[] juegos = new Juego[90];
		juegos[0] = new Juego("ELDEN RING", "ACCIÓN", 96, 2022, "PS, XBOX, PC", "FROMSOFTWARE");
		juegos[1] = new Juego("BLOODBORNE", "ACCIÓN", 92, 2015, "PS", "FROMSOFTWARE");
		juegos[2] = new Juego("SEKIRO: SHADOWS DIE TWICE", "AVENTURA", 90, 2019, "PS, XBOX, PC", "FROMSOFTWARE");
		juegos[3] = new Juego("DARK SOULS", "ACCIÓN", 89, 2011, "PS, XBOX, PC", "PLAYSTATION STUDIOS");
```

(En este ejemplo solo hemos puesto unos cuantos pero en nuestro código llega hasta los 90). Después de que esté todo creado en condiciones, haremos un do-while para que el código repita siempre los mismos pasos a no ser que el usuario indique lo contrario. Especial énfasis en explicar el método Comparator, que nos sirve para comparar las características de un objeto. Al final podemos usar el número de la operación para comenzar el switch con las diferentes opciones.

```
do {
                        do {
                                while (true) {

                                        // Usamos un while para repetir la opcion si se introduce un número incorrecto o una palabra la cual no deberia ser admitida.
                                        // Habrá un total de 7 opciones representadas por sus números correspondientes. Hemos usado switchs para el resto del codigo.
                                        // Usamos el metodo override para sobreescribir el metodo Comparator y cambiarlo por un compare. Lo usaremos durante el resto del codigo
                                        // Gracias al Comparator (ahora nombrado compare) podemos comparar las caracteristicas de un objeto (como ordenarlos de mayor a menor, ordenar alfabeticamente, etc)
                                        System.out.println(
                                                        "Bienvenido al buscador de videojuegos personalizado. Seleccione la opción con la que desea empezar su búsqueda.");
                                        System.out.println("1- Buscar por nombre:");
                                        System.out.println("2- Buscar por nota:");
                                        System.out.println("3- Buscar por año de lanzamiento:");
                                        System.out.println("4- Buscar por género:");
                                        System.out.println("5- Buscar por consola:");
                                        System.out.println("6- Buscar por desarolladora:");
                                        System.out.println("7 - ¿Quieres finalizar la busqueda?");
                                        if (teclado.hasNextInt()) {
                                                numeroOperacion = teclado.nextInt();
                                                break;
                                        } else {
                                                System.out.println("Entrada inválida. Por favor, ingrese un número.");
                                                teclado.next(); //
                                        }
                                }
                                if (numeroOperacion < 1 || numeroOperacion > 7) {
                                        System.out.println("Opción inválida. Intente de nuevo.\n");
                                }
                        } while (numeroOperacion < 1 || numeroOperacion > 7)
```

Caso ordenado por nombre

En el caso de que el usuario decida mostrar los juegos por nombre, este le llevará al **caso 1** del switch en el que se usará el mismo método del **"do while".**

Para que las operaciones de dentro mientras sea nulo, este se repita las veces necesarias.

Para validar, dentro de este bucle nos encontraremos un **"while"** para que el usuario solo pueda introducir un **número del 1 al 3** y si no es así te mostrará un mensaje de error. Y dentro de él mostrará las opciones que hay al usuario y al usuario introducir la operación, para verificar que este contenga un número hicimos un **"if"** para verificar que lo que introduzca el usuario sea un número entero y si no es así te muestre un mensaje de error.

A continuación, como explicado anteriormente, hemos usado un **switch** para que según el número de la operación este filtre según lo que desea el usuario.

Estas son las opciones que permite usar:

1) Para el **primer caso** que es para buscar por nombre específico, hemos usado lo siguiente:

- El usuario introduce el nombre a buscar, se usa el **método trim() y toUpperCase()** para eliminar espacios y convertir el nombre a mayúsculas.

```
nombreJuego = nombreJuego.trim().toUpperCase();
```

- Luego, se recorre el array de juegos y se usa **contains()** para verificar si el nombre del juego contiene la cadena introducida por el usuario.

```
if (juego.getNombre().toUpperCase().contains(nombreJuego)) {
    System.out.println("Juego encontrado: " + juego.getNombre() + " - " + juego.getGenero()
            + " - " + juego.getNota() + " - " + juego.getAnyoLanzamiento() + " - "
            + juego.getConsola() + " - " + juego.getDesarollador());
    encontrado = true;

}
```

Si el nombre no lo ha encontrado usaremos el condicional **if(!encontrado)** para mostrar un mensaje de error.

```
if (!encontrado) {
    System.out.println("No se encontró ningún juego con el nombre: " + nombreJuego);
}
```

2) Para el **segundo caso** que es para **ordenar de "A-Z"** hemos usado lo siguiente:

- Usamos el **Arrays.sort** ordenará los objetos del array. El método **compare(Juego j1, Juego j2)** Comparamos por el atributo nombre, cual es mayor alfabéticamente. El método compare() da negativo cuando el objeto j1 es menor que el objeto j2. Positivo, el objeto j1 es mayor que el objeto j2. Si el valor es cero, los dos objetos son iguales.

```
Arrays.sort(juegos, new Comparator<Juego>() {
    @Override
    public int compare(Juego j1, Juego j2) {
        return j1.getNombre().compareTo(j2.getNombre());
    }
});
```

Y por último mostrará al usuario todo el array modificado usando un **bucle for**:

```
for (Juego juego : juegos) {
    System.out.println(juego.getNombre());
}
```

3) Para el **tercer caso** que es para **ordenar de "Z-A"** hemos usado lo siguiente:

- Es similar al caso anterior, solo que cambia el orden del array, para ello solo debemos de cambiar el orden de los nombres en el return:

```
Arrays.sort(juegos, new Comparator<Juego>() {
    @Override
    public int compare(Juego j1, Juego j2) {
        return j2.getNombre().compareTo(j1.getNombre());
    }
});
```

Y por último mostrará al usuario todo el array modificado usando un **bucle for:**

```
for (Juego juego : juegos) {
    System.out.println(juego.getNombre());
}
```

Con el **“break”** saldría del caso y regresaría al inicio.

Caso ordenado por notas

En el caso de que el usuario decida mostrar los juegos por notas , este le llevara al caso 2 del switch (numeroOperacion = 2) en el que se usará el mismo método del **“ do while “**

Para que el que las operaciones de dentro mientras sea nulo este se repita las veces necesarias Para validar , dentro de este bucle nos encontraremos un “ **while** “ para que el usuario solo pueda introducir un número del 1 al 3 y si no es así te mostrará un mensaje de error. Y dentro de él mostrará las opciones que hay al usuario y al usuario introducir la operación, para verificar que este contenga un número hicimos un **“if”** para verificar que lo que introduzca el usuario sea un número entero y si no es así te muestre un mensaje de error .

A continuación como explicado anteriormente hemos usado un switch para que según el número de la operación este filtre según lo que desea el usuario .

- Para el primer caso que es para ordenar de **" De mayor a menor"** hemos usado lo siguiente :

    1. Usamos el **Array.sort** que hará recorrer y ordenar los objetos del array
    2. El método **compare()** devuelve un valor entero que indica la relación entre los dos objetos. Si el valor es negativo, el objeto j1 es menor que el objeto j2. Si el valor es positivo, el objeto j1 es mayor que el objeto j2. Si el valor es cero, los dos objetos son iguales.

```
Arrays.sort(juegos, new Comparator<Juego>() {
                        @Override
                        public int compare(Juego j1, Juego j2) {
```

```
                            return Integer.compare(j2.getNota(),
j1.getNota());
                        }
                    });
```

3. Y por último mostrará al usuario todo el array modificado usando un bucle for

```
for (Juego juego : juegos) {
                        System.out.println(juego.getNota()+" |
"+juego.getNombre());
```

4. Con el "**break**" saldría del caso y regresaría al inicio

- Para el segundo caso este filtra los juegos de "**Menor a mayor nota** ".

Para este caso es lo mismo que en le caso 1 solo que cambia el orden del array , para ello solo debemos de cambiar el orden de las notas del return

```
return Integer.compare(j1.getNota(), j2.getNota());
```

- Para el último caso es por "**la nota concreta** " **,** Para ello hemos hecho lo siguiente :

1. cerramos un int de la nota al buscar que introducirá el usuario
2. Creamos un bucle "**for** ", que recorre todo el array de juegos
3. Y por último con " **if** " comparamos cada juego si coincide con la nota introducida y si son iguales lo muestra

---

Caso ordenado por año de lanzamiento

Si el usuario escribe la opción para filtrar por año (numeroOperacion = 3) llegará hasta el caso nº 3.

Nada más entrar en el caso aparecerá en la consola otro menú para elegir que tipo de filtro por año de lanzamiento quiere utilizar.

- Pondremos este menú dentro de un **Do-While** para asegurarnos de que el número de la opción que escriba por teclado el usuario esté dentro del rango de opciones proporcionadas por dicho menú

- También pondremos este menú dentro de un **While (true)**, en el que se encontrara un **If (teclado.hasNextInt())** que comprueba que lo que ha escrito el usuario sea un **int** y no otro tipo de dato (**decimal, String, Char**...), si cumple esta condición entrará en el **if** que declara lo que ha escrito el usuario como **numeroOpereacionAnyo** y finaliza el bucle con un **break.**
  Si no cumple la condición existe un **else** que mostrará por pantalla el siguiente mensaje: "Entrada inválida. Por favor, ingrese un número."

Estas son las opciones que permite usar:

1. **"Mostrar todos los juegos a partir del año que introduzcas":**

   Para esta opción primero haremos un bucle **Do-While** para asegurarnos de que el año que introduzca el usuario sea mayor que 2000.
   A continuación crearemos un bucle **For** que recorra todo el array de juegos y muestre todos los juegos en los que el año sea mayor o igual que el año introducido.

2. **"Mostrar todos los juegos del año que introduzcas":**

   Para esta opción primero haremos un bucle **Do-While** para asegurarnos de que el año que introduzca el usuario sea mayor que 2000.
   Seguidamente haremos un bucle **For** que recorra todo el array y muestre únicamente los juegos en los que el año de lanzamiento sea igual que el año introducido por el usuario.

3. **"Mostrar los juegos ordenados por el año de lanzamiento de menor a mayor":**

   Para esta opción, al no depender de un dato introducido por el usuario realizamos directamente las acciones necesarias.
   En primer lugar usaremos un **array.sort** y un **Comparator.comparingInt** que, como su propio nombre indica comparara los años de lanzamiento de los juegos que sumandolo al **array.sort** que hemos creado ordenara el array de menor a mayor año de lanzamiento.
   Después haremos un bucle **For** para que recorra el array reordenado y lo muestre en la consola.

4. **"Mostrar los juegos ordenados por el año de lanzamiento de mayor a menor":**

   Para esta opción (igual que la anterior), al no depender de un dato introducido por el usuario realizamos directamente las acciones necesarias.

En primer lugar usaremos un **array.sort** y un **Comparator.comparingInt** que, como su propio nombre indica comparara los años de lanzamiento de los juegos y para que ordene los juegos de mayor a menor agregamos al final **.reversed()** lo que hace esta función es **invertir** el orden del comparador.
Después haremos un bucle **For** para que recorra el array reordenado y lo muestre en la consola.

Caso ordenado por género

En el caso de que el usuario decida mostrar los juegos por género, este le llevará al **caso 4** del switch en el que se usará el mismo método del **"do while".** Para que las operaciones de dentro mientras sea nulo, este se repita las veces necesarias.

Validar género.

El usuario introduce el género a buscar, se usa el **método trim() y toUpperCase()** para eliminar espacios y convertir el género a mayúsculas.

Tras introducir el texto nos muestra por pantalla el valor:
System.out.println("Juegos del género " + generoBuscar + ":");

```
System.out.println("Generos disponibles:");
System.out.println("Accion");
System.out.println("Aventura");
System.out.println("Deportivo");
System.out.println("Plataformas");
System.out.println("Puzzle");
System.out.println("RPG");
System.out.println("Shooter");
System.out.println("Sigilo");
System.out.println("Simulacion");
System.out.println("Terror");
```

Luego, se recorre el **array de juegos** y se usa **contains()** para verificar si el género del juego contiene la cadena introducida por el usuario.
Si se encuentra, **se muestra el juego con todos sus atributos.**
y cambiará el **booleano encontradoGenero = true.**

```
boolean encontradoGenero = false;
for (Juego juego : juegos) {
    if (juego.getGenero().toUpperCase().contains(generoBuscar)) {
        System.out.println(juego.getNombre() + " - " + juego.getGenero() + " - " + juego.getNota()
                + " - " + juego.getAnyoLanzamiento() + " - " + juego.getConsola() + " - "
                + juego.getDesarolladador());
        encontradoGenero = true;
    }
}

if (!encontradoGenero) {
    System.out.println("No se encontraron juegos para el género: " + generoBuscar);
}
```

En caso que no encuentre el género introducido, usaremos un **if(!encontradoGenero**) para mostrar un mensaje de error. El bucle se repite ya que el booleano = false, por lo que le volverá a pedir que introduzca un género por pantalla.

Caso ordenado por Consolas

Si el usuario escribe la opción para filtrar por consola (numeroOperacion = 5) llegará hasta el caso nº 5.

Nada más entrar en el caso aparecerá en la consola otro menú para elegir que tipo de filtro por año de lanzamiento quiere utilizar.

- Pondremos este menú dentro de un **Do-While** para asegurarnos de que el número de la opción que escriba por teclado el usuario esté dentro del rango de opciones proporcionadas por dicho menú
- También pondremos este menú dentro de un **While (true)**, en el que se encontrara un **If (teclado.hasNextInt())** que comprueba que lo que ha escrito el usuario sea un **int** y no otro tipo de dato (**decimal, String, Char**...), si cumple esta condición entrará en el **if** que declara lo que ha escrito el usuario como **opcionConsola** y finaliza el bucle con un **break.**
  Si no cumple la condición existe un **else** que mostrará por pantalla el siguiente mensaje: "Entrada inválida. Por favor, ingrese un número."

Estas son las opciones que permite usar:

5. **"Filtrar todos los juegos por consola:":**

Para esta opción primera haremos uso del método compare() para las consolas junto con los juegos correspondientes a esta.

6. **"Filtrar todos los juegos por consola específica: (PS: (Consolas playstation), XBOX (Consolas Xbox), PC, Switch, Wii, Nintendo DS)":**

   Para esta segunda opción necesitaremos una consola introducida por el usuario la cual almacenaremos en un String **nombreConsola**, asegurándonos de usar los métodos trim() y uppercase() para que si el usuario introduce espacios o minúsculas puedan compararse bien. Luego usaremos el método contains para compararlo con las consolas de todos los objetos juegos creados, dando un mensaje de error en caso de que no se escriba una de las descritas arriba y volviendo a preguntar.

Caso ordenado por desarrolladora

Si el usuario escribe la opción para filtrar por desarrolladora (numeroOperacion = 6) llegará hasta el caso nº 6.

Nada más entrar en el caso aparecerá en la consola otro menú para elegir que tipo de filtro por año de lanzamiento quiere utilizar.

- Pondremos este menú dentro de un **Do-While** para asegurarnos de que el número de la opción que escriba por teclado el usuario esté dentro del rango de opciones proporcionadas por dicho menú

- También pondremos este menú dentro de un **While (true)**, en el que se encontrara un **If (teclado.hasNextInt())** que comprueba que lo que ha escrito el usuario sea un **int** y no otro tipo de dato (**decimal, String, Char**...), si cumple esta condición entrará en el **if** que declara lo que ha escrito el usuario como **busqDesarrollador** y finaliza el bucle con un **break.**
  Si no cumple la condición existe un **else** que mostrará por pantalla el siguiente mensaje: "Entrada inválida. Por favor, ingrese un número."

Estas son las opciones que permite usar:

1. **"Ordenar alfabéticamente"**

Para esta opción crearemos un **array.sort** con un **Comparator**, en el compararemos el dato **“Desarrollador”** del array de juegos, gracias al **Comparator** el **array.sort** ordena las desarrolladoras por orden alfabético.
Después creamos un bucle **For** que recorra todo el array para mostrar la desarrolladora y el nombre por la consola.

2. **"Por desarrollador específico"**

   Para esta opción se mostrará un listado con las desarrolladoras comprendidas en el array de juegos y se pedirá al usuario que elija la que quiere buscar.
   A continuación crearemos un **Do-While** en el que la condición será que el String que ha introducido el usuario sea igual que alguna de las Desarrolladoras, para esto utilizaremos la función **Equals.**
   Por último crearemos un **For** que recorra todo el array y muestre en la consola el nombre de la desarrolladora y el nombre del juego.