Tema 1. Ejercicios de repaso de Java.

0) Ejercicios de Funciones.
1) Ejercicios de Clase String y uso de excepciones.
2) Ejercicios de Matrices.
3) Ejercicios de POO (Objetos, herencia y métodos de ordenación).
4) Ejercicios de Estructuras dinámicas.
5) Explicación Interfaces + Ejercicios.
6) Vídeo de introducción a Spring Framework, Spring MVC y Spring Boot.
7) Ejercicio inventado a entregar.

El problema de las tildes en este IDE se puede arreglar cambiando la codificación del proyecto en: Properties --> Text file encoding --> Other: --> Poner ISO-8859-1

0) Ejercicios de Funciones.

Ejercicio a) Función a la que se le pasan dos enteros y muestra todos los números comprendidos entre ellos, ambos incluidos. (*Ten en cuenta que el usuario puede poner el valor más pequeño en el 1º número o en el 2º número).*

Ejercicio b) Escriba una función que sume los n primeros números impares de un valor elegido por el usuario.

Ejercicio c) Crea una aplicación que nos calcule el área de un circulo, cuadrado o triangulo. Pediremos que figura queremos calcular su área y según lo introducido pedirá los valores necesarios para calcular el área. Crea un método por cada figura para calcular cada área, este devolverá un número real. Muestra el resultado por pantalla.

A continuación puedes ver la fórmula de cada figura:
Circulo: (radio^2)*PI
Triangulo: (base * altura) / 2
Cuadrado: lado * lado

Ejercicio d) Crea una aplicación que muestre en binario un número entre 0 y 255.

1) Ejercicios de Clase String.

Ejercicio a) Crea una función que reciba dos cadenas como parámetro (str1, str2) e imprima otras dos cadenas como salida (out1, out2).

- out1 contendrá todos los caracteres presentes en la str1 pero NO estén presentes en str2.
- out2 contendrá todos los caracteres presentes en la str2 pero NO estén presentes en str1.

Ejemplo: `str1=aeiou r str2=eo z w  out1=aiu out2=zw`

Ejercicio b) Crea una función que calcule y retorne cuántos días hay entre dos cadenas de texto que representen fechas.

- Una cadena de texto que representa una fecha tiene el formato "dd/MM/yyyy".
- La función recibirá dos String y retornará un int.
- El usuario eligirá el orden de las fechas (Cuál de las 2 fechas será la primera y cuál será la segunda). A continuación muestra el orden en el que quedan. Usa el mínimo de variables posible.
- La diferencia en días será absoluta (no importa el orden de las fechas).
- Para facilitar el ejercicio suponemos que todos los años tienen 365 días y cada mes tiene 30 días.
- Si una de las dos cadenas de texto no representa una fecha correcta se lanzará una excepción.
- Apartado extra: Crea una excepción propia que salte cuando sea la fecha de tu nacimiento.

Ejemplo salida por pantalla con formato de fechas correcto:

Bienvenido al restador de fechas. El formato de una fecha es: dd/MM/yyyy
Escriba la primera fecha
12/10/2024
Escriba la segunda fecha
17/06/2024
¿Cuál de las 2 fechas será la primera? (Elija 1 o 2)
2
La primera fecha es entonces: 17/06/2024
La segunda fecha es entonces: 12/10/2024
La diferencia de días es: 115

Ejemplo salida por pantalla con formato de fechas **incorrecto**:

Bienvenido al restador de fechas. El formato de una fecha es: dd/MM/yyyy
Escriba la primera fecha
22/11/2025
Escriba la segunda fecha
no/es/bien
¿Cuál de las 2 fechas será la primera? (Elija 1 o 2)
1
La primera fecha es entonces: 22/11/2025
La segunda fecha es entonces: no/es/bien
Se ha producido el siguiente error: java.lang.NumberFormatException: For input string: "no"
Formato de fechas incorrectas. Cerrando la aplicación.

2) Ejercicios de Matrices.

Ejercicio a) Crea una matriz con 5 filas y n columnas (Valor que debe pedirse al usuario). A continuación, rellénalo con números aleatorios entre 0 y 10. Para ello debes crear una función auxiliar que devuelva el número aleatorio generado y se haga la llamada desde la clase Main.

Ejercicio b) Crea 2 matrices de mxm y suma sus valores. Los resultados deben almacenarse en otra matriz distinta. Los valores y la longitud, seran elegidos por el usuario. Finalmente, muestra por pantalla las matrices originales y el resultado, para ello debes crear una función auxiliar que muestre las matrices y se haga la llamada desde la clase Main.

Ejemplo:
Escribe un tamaño: 2
Escriba el valor para la fila 0 y columna 0 de la matriz 1: 4
Escriba el valor para la fila 0 y columna 0 de la matriz 2: 3
Escriba el valor para la fila 0 y columna 1 de la matriz 1: 2
Escriba el valor para la fila 0 y columna 1 de la matriz 2: 4
Escriba el valor para la fila 1 y columna 0 de la matriz 1: 5
Escriba el valor para la fila 1 y columna 0 de la matriz 2: 6
Escriba el valor para la fila 1 y columna 1 de la matriz 1: 4
Escriba el valor para la fila 1 y columna 1 de la matriz 2: 3

Matriz 1
4 2
5 4

Matriz 2
3 4
6 3

Matriz resultante
7 6
11 7

Ejercicio c) Crea una matriz “marco” de tamaño 8x6: todos sus elementos deben ser 0 salvo los de los bordes que deben ser 1. Muestra el resultado.

Matriz marco:
1 1 1 1 1 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 1 1 1 1 1

Ejercicio d) Tabla de 1 dimensión: Pide 5 números que se introducirán ordenados de forma creciente. Éstos se guardan en una tabla de tamaño 10. A continuación se pide un número N, el cual debe insertarse en el lugar adecuado para que la tabla continúe ordenada.

3) Ejercicios de POO (Objetos, herencia, métodos de ordenación)

Ejercicio a) Crea una clase llamada Cuenta que tendrá los siguientes atributos: titular y cantidad (La cual puede tener decimales).

El titular será obligatorio y la cantidad es opcional. Crea dos constructores que cumpla lo anterior.

Crea sus métodos get, set y toString. Tendrá dos métodos especiales:

- ingresar(double cantidad): se ingresa una cantidad a la cuenta, si la cantidad introducida es negativa, no se hará nada.
- retirar(double cantidad): se retira una cantidad a la cuenta, si restando la cantidad actual a la que nos pasan es negativa, la cantidad de la cuenta pasa a ser 0.

Ejercicio b) Una empresa que se dedica a la venta de desinfectantes necesita un programa para gestionar las facturas. En cada factura figura: el código del artículo, la cantidad vendida en litros y el precio por litro.

Se pide de 5 facturas introducidas: Facturación total, cantidad en litros vendidos del artículo 1 y cuantas facturas se emitieron de más de 600 €.

Ejercicio c) Igual que el ejercicio b) pero suponiendo que no se introduce el precio por litro. Solo existen tres productos con precios: 1- 0,6 €/litro, 2- 3 €/litro y 3- 1,25 €/litro.

Ejercicio d) Crear una clase Libro que contenga los siguientes atributos:

- ISBN
- Titulo
- Autor
- Número de páginas

Crear sus respectivos métodos get y set correspondientes para cada atributo. Crear el método toString() para mostrar la información relativa al libro con el siguiente formato:

«El libro con ISBN creado por el autor tiene páginas»

En el fichero main, crea 2 objetos Libro y muestralos por pantalla.

Por último, indicar cuál de los 2 tiene más páginas.

¿Con qué método de Java se ordenan objetos?

¿Con qué método de Java se comparan objetos?

Método compareTo: Lectura en: https://guru99.es/string-compareto-method-java/

Ejemplos subidos a la plataforma.

Ejercicio e) Ejercicio Ordenación 1.

Apartado 1: Ordena por precios:

Apartado 2: Ordena por precios, a igual precio, ordena por marca alfabéticamente:

iPhone 12 Pro Max	(1.259 euros)
Xiaomi Mi 10 Pro	(999 euros)
Huawei P40 Pro+	(1.399 euros)
Samsung Z Flip 5G	(1.550 euros)
Samsung S20	(1500 euros)
LG V50	(899 euros)
Xiaomi Mi 10 Pro	(999 euros)
Huawei P40 Pro+	(1.399 euros)
Samsung Z Flip 5G	(1.550 euros)
Samsung S30	(1300 euros)
Huawei P50 Pro+	(1.399 euros)
Samsung Z Flip 5G	(1.550 euros)

Ejercicio f) Ejercicio Ordenación 2.

Crea 4 bolígrafos:

Boli 1: Pilot SuperGrip	1€
Boli 2: Pilot G2	1,3€
Boli 3: Bic Cristal	0,5€
Boli 4: Pilot G2	1,3€

Compara el boli 1 con el 2, Compara el 2 con el 4.

Ordena los bolis por marca.

Ejercicio g) Haz una clase llamada Persona que siga las siguientes condiciones:

- Sus atributos son: nombre, edad, DNI, sexo (H hombre, M mujer), peso y altura.

No queremos que se accedan directamente a ellos. Piensa que modificador de acceso es el más adecuado, también su tipo. Si quieres añadir algún atributo puedes hacerlo.

- Por defecto, todos los atributos menos el DNI serán valores por defecto según su tipo (0 números, cadena vacía para String, etc.). Sexo sera hombre por defecto, usa una constante para ello.

- Se implantaran varios constructores:
  - Un constructor por defecto.
  - Un constructor con el nombre, edad y sexo, el resto por defecto.
  - Un constructor con todos los atributos como parámetro.

- Los métodos que se implementaran son:
  - calcularIMC() : calculara si la persona esta en su peso ideal (peso en kg/(altura^2 en m)), si esta fórmula devuelve un valor menor que 20, la función devuelve un -1, si devuelve un

número entre 20 y 25 (incluidos), significa que esta por debajo de su peso ideal la función devuelve un 0 y si devuelve un valor mayor que 25 significa que tiene sobrepeso, la función devuelve un 1.

- ○ esMayorDeEdad() : indica si es mayor de edad, devuelve un booleano.
- ○ comprobarSexo(char sexo) : comprueba que el sexo introducido es correcto. Si no es correcto, sera H. No sera visible al exterior.
- ○ toString() : devuelve toda la información del objeto.
- ○ generaDNI() : genera un número aleatorio de 8 cifras, genera a partir de este su número su letra correspondiente. Este método sera invocado cuando se construya el objeto. Puedes dividir el método para que te sea más fácil. No será visible al exterior.
- ○ Métodos set de cada parámetro, excepto de DNI.

Ahora, crea una clase ejecutable que haga lo siguiente:

- ● Pide por teclado el nombre, la edad, sexo, peso y altura.
- ● Crea 3 objetos de la clase anterior, el primer objeto obtendrá las anteriores variables pedidas por teclado, el segundo objeto obtendrá todos los anteriores menos el peso y la altura y el último por defecto, para este último utiliza los métodos set para darle a los atributos un valor.
- ● Para cada objeto, deberá comprobar si esta en su peso ideal, tiene sobrepeso o por debajo de su peso ideal con un mensaje.
- ● Indicar para cada objeto si es mayor de edad.
- ● Por último, mostrar la información de cada objeto.

4) Ejercicios de Estructuras dinámicas.

Ejercicio a) Busca la diferencia y tipos de las estructuras dinámicas más usadas en Java: Set, List, Map.

Ejercicio b) Escriba un programa Java para vaciar un ArrayList.

Ejercicio c) Escriba un programa Java para probar si un ArrayList está vacío o no.

Ejercicio d) Escriba un programa Java para recortar la capacidad de un ArrayList al tamaño actual de la lista. Usa el método trimToSize, para ello revisa la API de Java de la clase ArrayList: https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/java/util/ArrayList.html

Ejercicio e) Escriba un programa Java para reemplazar el segundo elemento de una ArrayList con el elemento especificado.

Ejercicio f) Escriba un programa Java para imprimir todos los elementos de una ArrayList utilizando la posición de los elementos.

5) Explicación Interfaces

Es importante conocer el término Interface ya que las usaremos con el framework SpringBoot: https://www.freecodecamp.org/espanol/news/interfaces-java-explicadas-con-ejemplos/

Ejercicio Interfaces:

Suponga que debe crear distintas clases Java para describir los productos que vende un supermercado.

Para unificar el código de los distintos programadores del equipo debe crear las siguientes Interfaces Java para describir algunas características de los productos.

Interfaz EsLiquido
Esta interfaz indica que los objetos creados a partir de la clase serán líquidos, y tendrá los siguientes métodos:
public void setVolumen(int v);
public int getVolumen();
public void setTipoEnvase(String env);
public String getTipoEnvase();

Interfaz EsAlimento
Esta interfaz indica que los objetos creados a partir de la clase serán alimentos, y tendrá los siguientes métodos:
public void setCaducidad(LocalDate fc);
public LocalDate getCaducidad();
public int getCalorias();

Interfaz ConDescuento
Esta interfaz indicará que el producto tiene descuento e incluirá los siguientes métodos:
public void setDescuento(double des);
public double getDescuento();
public double getPrecioDescuento();

Creación de clases de productos
Se pide que programe las siguientes clases de productos, implementando las interfaces que sean necesarias.

Clase Detergente
Define una botella de detergente (debe tener en cuenta que este producto puede tener descuento)
Sus propiedades principales serán:
Marca (String) y Precio (double)
Incluya otras propiedades según sea necesario a la hora de implementar las interfaces.

Constructor
Programe un constructor que reciba como parámetros una marca y un precio.
Métodos get y set
Programe métodos set y get para la marca y el precio.

Métodos de las interfaces
Programe los métodos de las interfaces.

Método toString
Programe el método toString con todas las características del producto.

Clase Cereales
Define el producto caja de cereales. (Este producto no tiene descuentos)
Las propiedades del producto serán Marca, Precio y Tipo de cereal (Enum)

Programe un constructor que reciba como parámetros las tres propiedades anteriores.
Programe los métodos set y get para dichas propiedades.
Programe los métodos de las interfaces implementadas. (Si es necesario
añada más propiedades a la clase)

A tener en cuenta: las calorías serán las siguientes: 5 si el cereal es avena, 8 si es maíz, 12 si es trigo, y 15 en cualquier otro caso.
Programe el método toString para devolver una cadena con todas las características del producto.

Clase Vino
Esta clase describirá el producto botella de vino. (Este producto es susceptible de tener descuento)

El producto tendrá como propiedades la marca, el tipo de vino (String), los grados de alcohol y el precio. Programe al igual que los productos anteriores un constructor con estas cuatro propiedades como parámetros. Programe también los métodos set, get, toString y los métodos de las interfaces. Añada nuevas propiedades si es necesario.

A tener en cuenta: las calorías se calcularán multiplicando por 10 la graduación alcohólica.

Programa de prueba

Realice un programa de prueba dónde cree varios productos de cada clase usando un ArrayList de productos alimenticios calculando la suma de sus calorías. También crea un ArrayList de productos líquidos (Detergente). Muestra el importe total de todos los productos.
Por último muestra la suma de los descuentos sobre todos los Detergentes.

La salida por la pantalla es:

```
Listado de alimentos:
Cereales{marca=Oatmeal, tipoCereal=avena, precio=3.0, caducidad=11/03/2025, calorias=5}
Cereales{marca=Fitness fiber, tipoCereal=trigo, precio=2.25, caducidad=20/04/2025,
calorias=12}
Vino{marca=MarcaVino1, tipoVino=tinto, grados=12.0, precio=8.0
volumen=700 ml, tipoEnvase=Botella_de_cristal, caducidad=01/07/2025
descuento=5.0, precio con descuento=7.6, calorias=120}
Vino{marca=MarcaVino2, tipoVino=rosado, grados=6.0, precio=5.3
volumen=330 ml, tipoEnvase=Botella_de_cristal, caducidad=03/03/2025
descuento=5.0, precio con descuento=5.035, calorias=60}
Total de calorias: 197

Listado de líquidos:
Detergente{marca=agerul, precio=6.0, volumen=5000 ml,
tipoEnvase=Botella_de_plastico_grande, descuento=20.0, precio con descuento=4.8}
```

```
Detergente{marca=dixan, precio=4.0, volumen=2500 ml,
tipoEnvase=Botella_de_plastico_mediana, descuento=10.0, precio con descuento=3.6}
Detergente{marca=norit, precio=1.25, volumen=500 ml,
tipoEnvase=Botella_de_plastico_pequeña, descuento=2.0, precio con descuento=1.225}
Total de descuentos: 32%
Dinero ahorrado: 1.625 euros
```

6) Vídeo de introducción a Spring framework, Spring MVC y Spring Boot.

Veamos una introducción con el siguiente vídeo, el cual diferencia JSP y servlets de Spring y SpringBoot:
https://www.youtube.com/watch?v=cTozN8W6FGo&ab_channel=IsmaelRobles

7) Ejercicio inventado.

Ha llegado tu momento de poner en práctica todo lo que has repasado en este boletín. Debes crear una aplicación de escritorio (No web) con una pequeña ***documentación***. Debes usar al menos todos los recursos vistos hasta ahora:

- Funciones.
- Clase String y uso de excepciones.
- Matrices.
- POO (Objetos, herencia / métodos de ordenación).
- Estructuras dinámicas.
- Interfaces.

(Puedes usar también recursos vistos el curso pasado que pueden enriquecer tu aplicación como estructuras estáticas, nºs aleatorios, ficheros de texto…)

Documentación de la aplicación:

- ➔ Título del proyecto.
- ➔ Descripción de la aplicación: ¿Cual será la funcionalidad? ¿Qué información manejará? ¿Cómo se usa la aplicación? (Puede ser una explicación de su uso con capturas de pantalla o información de cada parte de forma detallada)

Criterios de calificación:

- ✔ **Uso de todos los recursos mínimos (30%)**
- ✔ **Interactividad (30%)**
- ✔ **Originalidad / creatividad (15%)**
- ✔ **Presentación y documentación de la aplicación (25%)**