PROGRAMACIÓN JAVA EJERCICIOS-1 TEMA 6. Clases y Objetos.

1. Define una clase Cuenta para representar una cuenta bancaria. Los datos de la cuenta son: nombre del cliente (String), número de cuenta (String), tipo de interés (double) y saldo (double).

   La clase contendrá los siguientes métodos:

   - Constructor por defecto
   - Constructor con todos los parámetros
   - Constructor copia.
   - Métodos setters/getters para asignar y obtener los datos de la cuenta.
   - Método ingreso. Un ingreso consiste en aumentar el saldo en la cantidad que se indique. Esa cantidad no puede ser negativa. Devuelve true si se ha podido realizar el ingreso. Devuelve false en caso contrario.
   - Método reintegro. Un reintegro consiste en disminuir el saldo en una cantidad pero antes se debe comprobar que hay saldo suficiente. La cantidad a retirar no puede ser negativa. Devuelve true si se ha podido realizar el ingreso. Devuelve false en caso contrario.

   Prueba el funcionamiento de la clase con este main:

```
public static void main(String[] args) {
   Scanner sc = new Scanner(System.in);
   String nombre, numero;
   double tipo, importe;

   //se crea objeto cuenta1 con el constructor por defecto
   Cuenta cuenta1 = new Cuenta();

   //Se piden los datos por teclado
   System.out.print("Nombre : ");
   nombre = sc.nextLine();
   System.out.print("Número de cuenta : ");
   numero = sc.nextLine();
   System.out.print("Tipo de interes : ");
   tipo = sc.nextDouble();
   System.out.print("Saldo: ");
   importe = sc.nextDouble();

   //Se asignan los valores leídos a cuenta1 mediante los métodos set
   cuenta1.setNombre(nombre);
   cuenta1.setNumeroCuenta(numero);
   cuenta1.setTipoInteres(tipo);
   cuenta1.setSaldo(importe);

   //se crea el objeto cuenta2 con el constructor con parámetros
   Cuenta cuenta2 = new Cuenta("Juan Ferrández Rubio", "12345678901234567890", 1.75, 300);

   //se crea cuenta3 como copia de cuenta1
   //se utiliza el constructor copia
   Cuenta cuenta3 = new Cuenta(cuenta1);

   //Mostrar las tres cuentas
   //Se utilizan los métodos get para acceder a los datos de cada objeto
   System.out.println("Datos de la cuenta 1");
   System.out.println("Nombre del titular: " + cuenta1.getNombre());
   System.out.println("Número de cuenta: " + cuenta1.getNumeroCuenta());
   System.out.println("Tipo de interés: " + cuenta1.getTipoInteres());
   System.out.println("Saldo: " + cuenta1.getSaldo());
   System.out.println();
```

```
    System.out.println("Datos de la cuenta 2");
    System.out.println("Nombre del titular: " + cuenta2.getNombre());
    System.out.println("Número de cuenta: " + cuenta2.getNumeroCuenta());
    System.out.println("Tipo de interés: " + cuenta2.getTipoInteres());
    System.out.println("Saldo: " + cuenta2.getSaldo());
    System.out.println();

    System.out.println("Datos de la cuenta 3");
    System.out.println("Nombre del titular: " + cuenta3.getNombre());
    System.out.println("Número de cuenta: " + cuenta3.getNumeroCuenta());
    System.out.println("Tipo de interés: " + cuenta3.getTipoInteres());
    System.out.println("Saldo: " + cuenta3.getSaldo());
    System.out.println();
}
```

2. Añade al método *main* del ejercicio 1 operaciones de ingreso y reintegro y comprueba que funcionan correctamente.

3. Añade a la clase Cuenta creada en el ejercicio 1, un método *transferencia* que permita pasar dinero de una cuenta a otra siempre que en la cuenta de origen haya saldo para poder hacerla. Un ejemplo de llamada al método transferencia puede ser:

   ```
   cuentaOrigen.transferencia(cuentaDestino, importe);
   ```

   que indica que queremos hacer una transferencia desde cuentaOrigen a cuentaDestino del importe indicado.

4. Crea una clase Contador que contenga un único atributo entero cont.

   La clase tendrá los siguientes constructores:

   - Constructor por defecto
   - Constructor con un parámetro para inicializar el contador con un valor no negativo. Si el valor inicial es negativo el contador tomará el valor cero como valor inicial.
   - Constructor copia.

   La clase contendrá los métodos:

   - Getter/setter
   - incrementar: incrementa el contador en una unidad
   - decrementar: decrementa el contador en una unidad. El contador nunca podrá tener un valor negativo. Si al decrementar se alcanza un valor negativo el contador toma el valor cero.

   Escribe un programa para probar la clase.

5. Crea una clase llamada Libro que guarde la información de cada uno de los libros de una biblioteca. La clase debe guardar el título del libro, autor, número de ejemplares del libro (indica el número total de copias del libro que hay en la biblioteca) y número de ejemplares prestados (indica cuantos ejemplares del libro están prestados). La clase contendrá los siguientes métodos:

   - un constructor por defecto.
   - un constructor con parámetros.
   - Setters/getters
   - método préstamo: incrementa el atributo *número de ejemplares prestados* cada vez que se realice un préstamo. No se pueden prestar libros de los que no quedan ejemplares para prestar. Devuelve true si se ha podido realizar la operación y false en caso contrario.
   - método devolución: decrementa el atributo *número de ejemplares prestados* cuando se produzca la devolución de un libro. No se pueden devolver libros que no se hayan prestado, es decir, no se puede hacer una devolución dce n libro si no hay ningún ejemplar prestado de ese libro. Devuelve true si se ha podido realizar la operación y false en caso contrario.
   - método para mostrar los datos de los libros.

   Escribe un programa para probar la clase.

6. Crea una clase Dados que represente el lanzamiento de dos dados.

   Los atributos de la clase son dos números enteros que representan el valor de los dos dados.

   La clase contendrá un método lanzamiento() para obtener un nuevo valor aleatorio de los dados, dos métodos para ver el valor de cada dado, un método *suma* que devuelva la suma de los valores de los dados. La clase contendrá además un constructor por defecto que asignará valores iniciales aleatorios cada vez que se cree un nuevo objeto de la clase.

7. A partir de la clase Fecha contenida en el tema, añade los siguientes métodos:
   - fechaCorta(): Devuelve un String con la fecha en formato corto: 08-02-2021.
   - fechaLarga(): Devuelve un String con la fecha en formato largo, empezando por el día de la semana: lunes 8 de febrero de 2021.
   - diaSiguiente(): cambia la fecha actual por la del día siguiente. El objeto de la clase Fecha al que se le aplique este método deberá quedar siempre en un estado consistente.

   Escribe un programa para probar estos métodos. El método diaSiguiete() pruébalo dentro de un bucle que imprima la fecha durante cada iteración del bucle.

8. Crea una clase Nif que se usará para manejar DNIs con su correspondiente letra. Los atributos serán el número de DNI y la letra que le corresponde. La clase dispondrá de los siguientes métodos:
   - Constructor que reciba el DNI y establezca la letra que le corresponde.
   - leer(): que pida por teclado el número de DNI y calcule de forma automática la letra
   - Método toString() que nos permita mostrar el NIF (ocho dígitos, un guión y la letra en mayúscula)

   La letra se calculará con un método privado de la siguiente forma:

   se obtiene el resto de la división entera del número de DNI entre 23 y se usa la siguiente tabla para obtener la letra que corresponde:

```
0 - T     1 - R     2 - W    3 - A     4 - G     5 - M    6 - Y     7 - F
8 - P     9 - D    10 - X   11 - B    12 - N    13 - J   14 - Z    15 - S
16 - Q   17 - V    18 - H   19 - L    20 - C    21 - K   22 - E
```

9. Crea una clase Password que tenga como atributo un String que representa una contraseña.

   La clase contendrá los siguientes constructores:
   - Constructor por defecto que crea una contraseña al azar de longitud 8.
   - Constructor con un parámetro de tipo int. Este parámetro indica la longitud de la contraseña que se va a crear.

     Las contraseñas en ambos casos se generan al azar y contendrán letras mayúsculas, letras minúsculas, dígitos y los caracteres . (punto) y _ (guión bajo).

   La clase contendrá un método llamado esFuerte() que comprueba si la contraseña es fuerte o no según estas normas:
   - La longitud es como mínimo de 8 caracteres.
   - Contiene como mínimo dos letras mayúsculas, dos letras minúsculas, un dígito y al menos un punto o un guión bajo.

     El método esFuerte() devuelve un boolean indicando si la contraseña es fuerte o no.

   La clase contendrá también un método modificarContraseña() para cambiar una contraseña ya existente. El método pide por teclado un String que contenga una nueva contraseña. Si la contraseña introducida es fuerte se modifica la contraseña actual. Si no es fuerte se deja la contraseña actual sin modificar. El método devuelve true si se ha modificado la contraseña y false en caso contrario.

   El valor de una contraseña solo se podrá modificar mediante el método modificarContraseña().

   Además de estos métodos puedes añadir a la clase los métodos que consideres necesarios.

   Escribe un programa para probar el funcionamiento de la clase.