Ejercicios con ArrayList.

1. Diseña un programa que sirva para trabajar con datos de una estación meteorológica. Para ello crea la clase DatosMeteo que guarda datos de temperatura, precipitaciones , etc. de una fecha concreta. Características:
    a. Atributos
        i. fecha, clase LocalDate (java.time).
        ii. temperaturas máxima y mínima, double.
        iii. humedad máxima y mínima, double.
        iv. precipitaciones, double
    b. Métodos.
        i. Constructor. Todos los atributos.
        ii. Constructor. Todos los atributos menos la fecha. Tomamos la fecha del sistema.
        iii. Getters, setters y redefinir toString.

    Crea una clase llamada GestionDatosMeteo. Tendrá como atributo un arrayList de la clase DatosMeteo y los métodos:
    - anadirDatos. Recibe un objeto de la clase DatosMeteo y lo añade al arrayList.
    - mediaTempMax. Devuelve un double.
    - mediaTempMax. Media de las máximas de un mes y año pasados como parámetros (enteros). Devuelve double.
    - numeroRegistros. Devuelve el número de elementos del arrayList.

2. Crea un programa en Java que trabaje con un ArrayList de la clase Producto utilizada en los ejercicio7/8 de la unidad 5 (copia las clases en este nuevo proyecto). Diseña una nueva clase llamada GestionProductos que contendrá como atributo el ArrayList de productos y los métodos:
    - nuevo producto. El producto podrá ser congelado, refrigerado o fresco.
    - eliminar producto. Localizar mediante su ID. Devolver true/false según se haya podido eliminar o no.
    - devolver productos. Devolver todo el ArrayList.
    - Devolver los productos del tipo…Filtrar según tipo de producto (congelado, fresco o refrigerado). Devolver un ArrayList con los productos del tipo seleccionado.
    - eliminar todos los productos congelados. Retornar el número de productos eliminados.

3. Crea un programa Java que trabaje con la clase Evento. Esta clase tiene las siguientes características:
    a. Atributos:
        i. Fecha evento, clase DateTime/LocalDate.
        ii. Descripción, String
        iii. Precio, double
        iv. Lugar, String.
    b. Métodos.
        i. Constructor
        ii. Getters y setters

iii. Redefine toString.

El programa trabajará también con la clase GestionEventos que contiene como atributo un arrayList de eventos y los métodos.

- nuevoElemento, recibe un evento y lo guarda en el arrayList.
- listaEventos. Método sobrecargado que devuelva
  - todos los eventos
  - los de un año pasado como parámetro
  - los de un año y un mes pasados como parámetros
  - los de un año, mes y día pasados como parámetros
  - los de una fecha (clase DateTime/LocalDate) pasada como parámetro
- mostrar eventos ordenados por fecha
- eliminar los eventos de un año y mes pasados como parámetros.

4. En un proyecto Java construye un arrayList que almacene ciudades. Una ciudad tiene como atributos nombre, país y habitantes. Inserta datos en el array y muestra estos ordenados por
   a. número de habitantes.
   b. país.
   c. nombre de ciudad, ascendente y descendente, según se elija.

HashMap.

5. Diseña un proyecto Java que permita añadir, borrar, buscar y listar un HashMap. Las parejas del HashMap están formadas por DNI y nombre de personas (String).

6. Crea una clase llamada Publicacion destinada a guardar datos de periódicos y revistas. Sus atributos son: título, tipo (deportes, prensa general, motor,...), editorial, periodicidad, número de ejemplares. Crea su constructor, getters y setters. Diseña una aplicación Java que trabaje con la clase GestionPublicaciones que contendrá un HashMap que pueda almacenar publicaciones utilizando como clave el título de la publicación. En esta clase programa los métodos añadir publicación, eliminar publicación y lista publicaciones.

7. Crea una clase llamada Diccionario. Un diccionario debe tener como atributo un idioma y una relación de términos o palabras. Has de tener en cuenta que un término puede tener uno o varios significados. Crea métodos para añadir términos y significados y para buscar los significados de un término.
   Ejemplos de métodos a incluir en Diccionario:

- public void addPalabra(String palabra, String significado)
- public void addPalabra(String palabra, List<String> significados)
- public void addPalabra(String palabra, String [] significados)
- public void addPalabra(String palabra, String significados, String separador)
- public List<String> getSignificadosDe(String palabra)

8. Diseña un programa Java que cree un ArrayList y lo llene con 10 valores enteros elegidos al azar y comprendidos entre 1 y 10. El programa mostrará en pantalla los valores del arrayList en orden ascendente y las veces que se repite cada uno. Utiliza un HashMap para realizar las cuentas.

9. Partimos de una lista con objetos de la clase Multa (matrícula, fecha de la multa y el importe de la misma) y queremos obtener:
    a. Relación de las matrículas con sus multas. Ordenar por matrícula y por fecha.
    b. Relación de matrículas con el número de multas y el importe total de esas multas. Ordenar también por matrícula.

    Descarga el proyecto desde aquí.

10. Diseña una aplicación que sirva para gestionar datos de estudiantes. Para ello debes crear las clases que siguen.
    Esta es la descripción base de las clases, puedes añadir métodos que consideres necesarios.
    - Estudiante
        - Atributos: id, nombre, curso, HashMap de notas (pareja asignatura-nota)
        - Métodos: Constructor (solo id, nombre y curso), getters, setters, toString, addNota.
    - GestionEstudiante.
        - Atributos: ArrayList de Estudiantes
        - Métodos:
            - addEstudiante. Recibe un estudiante y lo añade a la colección.
            - getEstudiante. Recibe un id y devuelve el estudiante con esa id o Optional<Estudiante> si no lo encuentra.
            - getNotasEstudiante. Recibe un id de estudiante y devuelve sus notas (retornar el mapa).
            - getAprobados. Devuelve un hashMap con los estudiantes que tienen todas las asignaturas aprobadas. El hashMap tendrá el id y nombre del estudiante.
            - getAprobadosXAsignatura. Devuelve un listado de los nombres de los alumnos aprobados en una asignatura que se pasará como parámetro.

11. Diseña una aplicación Java que permita gestionar una agenda. Cada contacto de la agenda se identifica por su nombre y de él podrán guardarse varios teléfonos y/o correos electrónicos. Crea métodos que permitan insertar nuevos contactos, agregar teléfonos/correos a un contacto existente, buscar teléfonos/correos…
    a. Programar la agenda con HashMap
    b. Programar la agenda con Set (sin repetir nombre de contacto).

    Crea un interfaz que deban implementar ambos tipo de agenda.

12. Diseña una clase llamada GestionNumeros que tenga como atributo un arrayList que permita trabajar con enteros (Integer). Las opciones(métodos) a programar en esta clase serán:
    a. en el constructor, añade varios valores al arrayList.
    b. añadir un valor al arrayList.
    c. añadir una colección completa al arrayList.
    d. mostrar el arraList en formato toString.

e. ordenar el arraylist de forma ascendente y descendente (indicarlo mediante un parámetro).
f. retornar el tamaño del arrayList.
g. retornar los valores pares del arrayList en formato List<Integer>.
h. retornar la suma de los valores del arrayList
i. eliminar un entero pasado como parámetro. Si puede eliminarse, retorna true. En caso contrario, retorna false.
j. eliminar del arrayList los múltiplos de “n”. Siendo “n” un valor pasado como parámetro. Retornar cuántos elementos se han eliminado.
k. retornar un array que contenga todos los valores pares del arrayList.

13. Diseña una aplicación java que trabaje con un arrayList de cadenas (String). Para ello debes programar una clase que tenga como atributo un arrayList de String y los métodos.
    a. addCadena. Recibe una cadena de caracteres con formato “valor1;valor2;valor3;...;valorN;” e inserta individualmente todos esos valores en el arrayList.
    b. buscarCadena. Retorna true si la cadena pasada como parámetro está contenida en el arrayList, false en caso contrario.
    c. getListaPatron. Retorna una lista (List) con todas las cadenas del arrayList que contienen el patrón pasado como parámetro.
    d. ordenar. Ordena el contenido del arrayList de forma ascendente o descendente, según indique el parámetro pasado.

14. Diseña una aplicación Java que permita:
    a. Guardar una matriz cuadrada de datos enteros en un arrayList.
    b. Pedir al usuario el número de filas y generar los valores enteros de forma aleatoria. Los valores generados deben estar comprendidos entre 0 y 9.
    c. Mostrar la matriz en pantalla.
    d. Calcular y mostrar la suma de las dos diagonales.

15. Diseña un programa Java que lleve a cabo la gestión de una empresa de paquetería. Para ello debes crear las siguientes clases:

- Paquete.
    - Atributos.
        - id, String.
        - fecha de envío y fecha de recepción, DateTime o LocalDate.
        - estado, String. El estado sólo debe permitir los valores: ALMACEN, REPARTO y ENTREGADO.
        - cliente origen y cliente destino, clase Cliente.
    - Métodos.
        - constructor, toString, getters y setters. El constructor asignará a la fecha de recepción null y al estado el valor ALMACEN.
        - Si se cambia el estado del paquete a ENTREGADO, debe ponerse como fecha de recepción la fecha del sistema. Si el cambio es a los

estados ALMACEN o REPARTO, la fecha de recepción se pondrá de nuevo a null.

- Cliente.
  - Atributos.
    - DNI, nombre, dirección, teléfono y email. Todos String.
  - Métodos.
    - constructor, getters, setters y toString.
- GestionPaquetes.
  - Atributos.
    - ArrayList, clase Paquete.
  - Métodos.
    - nuevoPaquete. Recibe un paquete y lo almacena en el ArrayList.
    - cambiarEstado. Parámetros: id de paquete y estado. Cambia el estado del paquete con el id indicado al nuevo estado pasado pasado como parámetro. Devuelve true si pudo hacerse el cambio, false en caso contrario.
    - getTodos. Devuelve todo el ArrayList.
    - consultarEstadoPaquete. Parámetro: id de paquete. Devuelve un String con el valor del estado de ese paquete, y null en caso de que ese id no exista en la colección.
    - getPaquetesEstado. Parámetro: un valor de estado. Devuelve un HashMap con los paquetes en los que el estado coincide con el pasado como parámetro. El HashMap estará formado por la pareja: id, Paquete.

Una vez diseñadas estas clases, pruébalas en el main de un proyecto. Crea un objeto de la clase GestionPaquetes y prueba sus métodos.

- La creación de un nuevo paquete se hará sin que intervenga el usuario (valores al azar).
- Para probar el método getTodos, recorre el ArrayList devuelto y llama a toString de cada objeto.
- Para probar el método getPaquetesEstado, recorre el HashMap devuelto y muestra los datos en pantalla mediante toString.