Ejercicios Tema 9. Estructuras de datos II. Colecciones

*Todos los ejercicios de este boletín tienen que estar dentro de un **package** llamado **tema9** (siguiendo las reglas que hemos visto en clase, ejemplo: com.germangascon.tema10).*

1. Implementa un método que reciba como parámetro una Array de números enteros y devuelva una lista (ArrayList) en la que primero aparezcan los números pares y después los impares. Posteriormente, genera una lista de 10 números enteros aleatorios entre 0 y 50 y aplica el método anterior. Un ejemplo de ejecución del programa sería:

```
Array generado:
[47, 14, 28, 10, 33, 15, 26, 7, 8, 28]

ArrayList con pares-impares:
[14, 28, 10, 26, 8, 28, 47, 33, 15, 7]
```

2. Crea una clase llamada ArrayListEstadisticas que declare un atributo de tipo ArrayList<Double> y que implemente la siguiente interfaz:

```
public interface IEstadisticas {

        double minimo();
        double maximo();
        double sumatorio();
        double media();
        double moda();
}
```

3. Escribe una clase llamada Pila que implemente la estructura Pila que vimos en el anexo II del tema 7, pero en esta ocasión utilizando estructuras dinámicas (ArrayList). Para llevar a cabo la implementación de esta clase tendrás que definir en primer lugar la siguiente interfaz que declara las operaciones básicas de una Pila:

```
public interface IPila<T> {

        T push(T e);
        T pop();
        int size();
        T top();
        boolean isEmpty();
}
```

   y posteriormente implementar esa interfaz en la clase Pila.

   Después, desde la clase principal crea un objeto de tipo Pila, añade algunos valores aleatorios, borra otros y finalmente muestra el contenido de la Pila.

   Reflexión:

   Examina la clase Stack del paquete java.util, observa sus métodos y compáralos con los que has definido en la clase Pila.

4. Escribe una clase llamada Cola que implemente la estructura Cola que vimos en el anexo II del tema 7, pero en este caso utilizando estructuras dinámicas (ArrayList). Para llevar a cabo la implementación de esta clase tendrás que definir en primer lugar la siguiente interfaz que declara las operaciones básicas de una Cola:

```
public interface ICola<T> {

        boolean add(T e);
        T remove();
        int size();
        T peek();
        boolean isEmpty();
}
```

   y posteriormente implementar esa interfaz en la clase Cola.
   Después, desde la clase principal crea un objeto de tipo Cola, añade algunos valores aleatorios, borra otros y finalmente muestra el contenido de la Cola.
   Reflexión 1:
   Examina la clase LinkedList del paquete java.util, observa sus métodos y compáralos con los que has definido en la clase Cola.
   Reflexión 2:
   Examina la clase ArrayList del paquete java.util, observa sus propiedades y sus métodos y compáralos con los de la clase LinkedList. ¿És ArrayList realmente una estructura dinámica? ¿por qué tiene un constructor donde le podemos pasar una capacidad inicial?

5. Queremos guardar la información de un conjunto de pacientes para posteriormente obtener datos estadísticos. De cada paciente queremos guardar un identificador (autonumérico), nombre, la fecha de nacimiento (dd/mm/yyyy), el sexo (únicamente se admiten los valores 'M' o 'F'; por defecto estará a 'M'), la altura (en metros) y el peso (Kg). Constructor con todos los datos.
   El programa deberá cumplir los siguientes requisitos:
   a) Un método llamado majorMenor, que recibirá una lista de pacientes y devolverá un *array* de 2 números enteros. La primera componente del A*rray* devuelto guardará la posición del paciente más pequeño (en edad) y en la segunda componente se guardará la posición del paciente más mayor (también en edad).
   b) Tendrá un método llamado pacientsPerSexe que, como en el caso anterior, recibirá una lista de pacientes y devolverá un Array de 2 números enteros. La primera componente del Array devuelto guardará la cantidad de hombres y la segunda componente guardará la cantidad de mujeres.
   c) Un método para calcular el índice de masa corporal (IMC). Este índice se calcula a partir de la siguiente fórmula:
      IMC = pes / (altura*altura)
      El método se aplica sobre un paciente y devuelve un *double*.
   d) Un método para calcular la edad en años de los pacientes. Esta edad será la que se mostrará

en el toString del paciente.

e) Un método que, a partir del índice de masa corporal, visualice el mensaje que corresponda según la siguiente tabla:

IMC	**Mensaje**
<18.5	Peso insuficiente
18.5-24.9	Peso normal
25-26.9	Sobrepeso grado I
27-29.9	Sobrepeso grado II
>29.9	Sobrepeso grado III

Desde el programa principal, crea una lista de pacientes. Los pacientes tendrán la siguiente información:

1	12/02/1980	F	57	1,63
2	07/03/1990	F	60,5	1,74
3	20/03/1967	F	50,8	1,62
4	20/04/1972	M	72,5	1,78
5	29/02/1960	M	85,2	1,8

Veamos un ejemplo de ejecución del programa:

```
PACIENTE MAYOR Edad: 58 Sexo: M
PACIENTE MENOR Edad: 28 Sexo: F

Cantidad de pacientes por sexos:
Hombres: 2    Mujeres: 3

Paciente número 1: Peso normal.
Paciente número 2: Peso normal.
Paciente número 3: Peso normal.
Paciente número 4: Peso normal.
Paciente número 5: Sobrepeso grado I.
```

Realizar las llamadas a los métodos que corresponda para que la ejecución sea igual.

6. Queremos hacer una aplicación que guarde parejas de palabras en valenciano-inglés (por ejemplo: llit - bed), para después poder mostrar su traducción. Veamos un ejemplo de ejecución del programa:

```
MENÚ PRINCIPAL
==============
1. Introducir parejas de palabras.
2. Traducir palabras.
0. Salir de la aplicación.

Elige una opción: 1
¿Cuántas parejas deseas introducir? 3
Introduce palabra en inglés: bed
Introduce traducción al valenciano: llit
Introduce palabra en inglés: chair
Introduce traducción al valenciano: cadira
Introduce palabra en inglés: table
Introduce traducción al valenciano: taula
```

```
MENÚ PRINCIPAL
==============
1. Introducir parejas de palabras.
2. Traducir palabras.
0. Salir de la aplicación.

Elige una opción: 2
Palabra a buscar: chair
cadira

MENÚ PRINCIPAL
==============
1. Insertar parejas de palabras.
2. Traducir palabras.
0. Salir de la aplicación.

Elige una opción: 2
Palabra a buscar: green
La palabra no existe.
```

Reflexiona sobre cuál es la mejor estructura para guardar las parejas de palabras.

7. Haz un programa que utilice la interfaz Map para calcular el cambio del euro a las siguientes monedas:
   a) USD (Dólar USA)
   b) GBP (Libra esterlina)
   c) INR (Rupia India)
   d) AUD (Dólar Australiano)
   e) CAD (Dólar Canadiense)
   f) ARS (Peso Argentino)
   g) BOB (Boliviano Boliviano)
   h) CLP (Peso Chileno)
   i) VEZ (Peso Colombiano)
   j) CRC (Colón Costarricense)
   k) CUP (Peso Cubano)
   l) DOP (Peso Dominicano)
   m) MXN (Peso Mexicano)

8. Escribe un programa que permita la gestión de un diccionario. El programa mostrará un menú similar al siguiente:

```
**********************
* GESTIÓN DICCIONARIO *
**********************
1. Añadir palabra
2. Modificar palabra
3. Eliminar palabra
```

```
4. Consultar palabra
5. Mostrar diccionario
---------------------
0. Salir
```

Al seleccionar la opción *Añadir palabra*, solicitará la palabra y su definición y si no existe la añadirá al diccionario. En caso de existir mostrará un mensaje de error.
Al seleccionar la opción *Modificar palabra*, solicitará la palabra a modificar y si existe pedirá su definición y la modificará.
Al seleccionar la opción *Eliminar palabra*, solicitará la palabra a borrar y si existe la borrará.
Al seleccionar la opción *Consultar palabra*, solicitará la palabra a consultar y si existe, mostrará su definición.
Al seleccionar la opción *Mostrar diccionario*, mostrará todas las palabras y definiciones que hay actualmente en el diccionario.

9. A partir del código del ejercicio anterior, crea un juego que muestre al usuario la definición de una palabra del diccionario y el usuario tenga que adivinarla. El menú será similar a:

```
**********************
* JUEGO DICCIONARIO *
**********************
1. Añadir palabra
2. Modificar palabra
3. Eliminar palabra
4. Consultar palabra
5. Mostrar diccionario
6. Jugar
7. Mejores puntuaciones
---------------------
0. Salir
```

Al seleccionar la opción *Jugar*, el programa mostrará la definición de una palabra y el usuario tratará de adivinarla. Cada vez que el usuario acierte una palabra sumará un punto. Este proceso se repetirá hasta que el usuario falle. Una vez acabado el juego, se deberá comprobar si la puntuación del usuario está dentro de las 5 mejores puntuaciones. En caso afirmativo, se le pedirá el nombre para añadirlo a las 5 mejores puntuaciones.
Al seleccionar la opción *Mejores puntuaciones*, se mostrarán el nombre y la puntuación de los 5 usuarios con mejor puntuación.

10. Escribe un programa que permita la gestión de empleados de una empresa. El programa mostrará un menú similar al siguiente:

```
**********************
* GESTIÓN EMPLEADOS *
**********************
1. Nuevo empleado
2. Nuevo hijo
3. Modificar sueldo
4. Borrar empleado
5. Borrar hijo
6. Consultas
----------------------
0. Salir
```

Al seleccionar la opción *Nuevo empleado*, solicitará los datos de un empleado, los validará y si son correctas y el empleado no existe, lo guardará en la lista de empleados. De cada empleado nos interesa el documento de identidad, nombre, apellidos, la fecha de nacimiento, el sueldo y los datos de sus hijos (si tiene) para calcular el IRPF. De cada hijo nos interesa el nombre y la edad.

Al seleccionar la opción *Nuevo hijo*, solicitará el documento de identidad del empleado al cual le queremos añadir un hijo. Si el empleado existe, pedirá los datos del hijo y lo añadirá a la lista de hijos del empleado.

Al seleccionar la opción *Modificar sueldo*, solicitará el documento de identidad del empleado al cual le queremos modificar el sueldo. Si el empleado existe, pedirá el nuevo sueldo y lo modificará.

Al seleccionar la opción *Borrar empleado*, solicitará el documento de identidad del empleado y si existe, lo borrará de la lista de empleados.

En seleccionar la opción Borrar hijo, solicitará en primer lugar documento de identidad del empleado del cual queremos borrar un hijo. Si el empleado existe, pedirá el nombre del hijo y si el empleado tiene un hijo con ese nombre lo borrará.

Al seleccionar la opción *Consultas*, mostrará un submenú similar al siguiente:

```
************************
* CONSULTAS EMPLEADOS *
************************
1. Buscar por NIF
2. Buscar por nombre
3. Buscar por rango de edad.
4. Buscar por rango de sueldo.
5. Buscar por hijos menores de edad
-----------------------------
0. Volver al menú principal
```

Al seleccionar la opción *Buscar por NIF*, solicitará el documento de identidad del empleado y si existe, mostrará sus datos por pantalla.

Al seleccionar la opción *Buscar por nombre*, solicitará el nombre o parte del nombre y buscará todos los empleados que el nombre contenga la cadena de caracteres introducida por el usuario.

Al seleccionar la opción *Buscar por rango de edad*, solicitará una edad mínima y una edad máxima y mostrará los datos de aquellos empleados que tienen una edad comprendida entre la edad mínima y la edad máxima.

Al seleccionar la opción *Buscar por rango de sueldo*, pedirá un sueldo mínimo y un sueldo máximo y mostrará los datos de aquellos empleados que tienen un sueldo comprendido entre el sueldo mínimo y el sueldo máximo.

Al seleccionar la opción *Buscar por hijos menores de edad*, mostrará los datos de aquellos empleados que tienen hijos menores de edad.

En seleccionar la opción *Volver al menú principal,* volverá a mostrar el menú principal del programa.

Como ya hemos comentado en otros ejercicios, crea método que genere empleados aleatorios para facilitar las pruebas (casos de prueba).

11. Escribe un programa para gestionar los alumnos de un centro. Cada alumno tiene un identificador (autonumérico), un nombre, un grupo y una lista de asignaturas en las cuales está matriculado. Cada asignatura tiene un código identificador, un nombre y un profesor que la imparte. Cada grupo tiene un código identificador, un nombre y un aula de referencia. Cada aula tiene un código identificador y unos metros cuadrados. Cada profesor tiene un dni, un nombre y un sueldo.

    En este caso se deja a criterio tuyo cuáles van a ser las opciones del menú, pero como mínimo tiene que cubrir los siguientes puntos:

    a) Alta de alumnos, asignaturas, grupos, aulas y profesores

    b) Mostrar alumnos por grupo, por profesor

    Cualquier ampliación extra será bienvenida.