Tema 7. Estructuras de almacenamiento

Vectores

1. Crea un programa *Personas* que almacene en un array los nombres de 20 personas introducidos por teclado. Diseña los siguientes métodos:

   a. imprimePersonas: visualiza por pantalla los elementos del array, una en cada fila.
   b. pares: visualiza por pantalla los elementos del array que ocupan las posiciones pares. Cada elemento debe ir en una fila.

2. Crea un programa *Datos* que almacene en un array 10 números enteros. Imprime por pantalla los elementos que ocupan las posiciones pares y su suma utilizando un método llamado sumaPares.

3. Crea un programa *Datos2* que visualice los elementos pares que ocupan las posiciones impares del array creado en el ejercicio anterior, su cuenta y su suma. Además de imprimir las posiciones que ocupan dichos elementos.

4. Crea un programa *Frases* que almacene en un array unidimensional 5 frases que se introducen por teclado. Diseña los siguientes métodos:

   a. imprimeFrases: imprime por pantalla el contenido del array.
   b. mayorFrase: imprime por pantalla la frase de mayor longitud y la posición que ocupa en el array.
   c. menorFrase: imprime la frase más pequeña y la posición que ocupa.

5. Escribe un programa *ListaAleatoria* que cree e imprima por pantalla un array de 10 elementos con números aleatorios comprendidos entre 1 y 10, de tal forma que no se repita ninguno.

6. Crea un programa *Capicua* que compruebe si un número es capicúa utilizando un array.

Matrices

1. Escribe un programa *Matriz1* que genere un array bidimensional 5x5, de tal forma que sus filas pares sean múltiplos de 2 y las impares sean múltiplos de 3. Además diseña los siguientes métodos:

a. imprimirMatriz: muestra por pantalla la matriz creada.
b. sumaMatriz: muestra la suma de todos sus elementos.
c. diagonal: imprime los elementos de su diagonal principal.

2. Crea un programa *MatrizTraspuesta* que genere una tabla 4x5 cuyos valores sean aleatorios entre 1 y 100. A partir de ella crea su traspuesta. Se deben mostrar por pantalla ambas tablas. Utiliza los métodos necesarios para ello.

NOTA: La traspuesta de una matriz es aquella matriz que se consigue cambiando las filas por columnas o viceversa.

3. Crea un programa *Permutación* que trabaje con una matriz de enteros cuya dimensión será solicitada al usuario, y se rellenará con aleatorios entre 1 y 100. El programa deberá permitir realizar permutaciones a través de los siguientes métodos:

a. permutaFilas: recibe como parámetros un array bidimensional de enteros "m", un entero "fila1" y otro entero "fila2". El método debe intercambiar las filas "fila1" y "fila2".
b. permutaColumnas: de forma análoga intercambia columnas.

4. Crea un programa que recorra una matriz de dimensiones generadas aleatoriamente, por columnas.

Búsqueda y ordenación

1. Escribe un programa *Ordenacion1* que cree un array de palabras introducidas por teclado, el tamaño del array será determinado por el usuario. Seguidamente se imprimirá la lista por pantalla, a continuación, deberás ordenar la lista por longitud de las palabras (tomando como base el método de ordenación de la **burbuja**) y finalmente imprimir de nuevo la lista de palabras ordenada.

2. Escribe un programa *Ordenacion2* que cree un array de palabras introducidas por teclado, el tamaño del array será determinado por el usuario. Imprime la lista., ordénala alfabéticamente e imprímela de nuevo. El programa también deberá permitir buscar el número de palabras del array que empiezan por un carácter que se solicita al usuario.

Colecciones

1. Realiza un programa que permita gestionar una agenda de contactos telefónicos, los cuales se almacenarán en un ArrayList.

Cada contacto de la agenda será un objeto de tipo **Contacto**, con los atributos, id (único), "nombre" y "tf". Ten en cuenta que no se podrán crear contactos sin nombre.

La agenda permitirá realizar las siguientes operaciones:

- Añadir un nuevo contacto. Si ya existe un contacto con ese nombre ó teléfono, se le informará al usuario antes de su almacenamiento.
- Buscar un contacto por nombre ó teléfono. Si existen varios contactos coincidentes, se mostrarán todos.
- Modificar los datos de un contacto (previa búsqueda).
- Eliminar un contacto (previa búsqueda). Antes de su eliminación se pedirá confirmación al usuario.
- Mostrar un listado de todos los contactos almacenados en la agenda, ordenado por nombre o teléfono.
- Vaciar la agenda.

2. Usa una tabla hash (HashMap) para almacenar clientes y realizar las operaciones básicas.

Un Cliente tiene los siguientes atributos:

- id: numérico y único
- dni
- nombre
- saldo

Operaciones a realizar:

- Introducir clientes en el mapa.
- Modificar los datos de un cliente (a excepciÛn de su id).
- Eliminar clientes del mapa.
- Buscar clientes por id y por dni.
- Mostrar todos los clientes del mapa.

3. Realiza un programa que permita gestionar las notas de las diferentes asignaturas de DAM de los alumnos de una clase.

Cada alumno, dispone de 3 notas correspondientes a las 3 evaluaciones.

El programa permitirá añadir la nota a cada asignatura en cada evaluación. También mostrará un listado de las medias de la clase en cada asignatura, y un listado de los alumnos ordenados alfabéticamente por apellidos.

Pilas y Colas

1. Realiza un programa *Parentesis* que a partir de una cadena con un conjunto de paréntesis, "(" o ")", e indique si están bien o mal cerrados.

Ejemplos:

)	mal cerrado
()	bien cerrado
(()	mal cerrado
)(	mal cerrado
(()()(()))	bien cerrado
(()))()	mal cerrado

4. Partiendo de dos pilas de números enteros, la primera ordenada ascendentemente desde la cima hasta el fondo, y la segunda ordenada descendentemente desde la cima hasta el fondo, realiza un programa que fusione ambas pilas en una tercera ordenada descendentemente desde la cima hasta el fondo.

NOTAS:

- Los tamaños de las dos pilas iniciales, se generarán aleatoriamente entre 1 y 20.
- Igualmente, los contenidos de las pilas de partida, se generarán aleatoriamente entre 1 y 100.

Arrays de objetos