Práctico 11 – Recursividad (parte 1)

OBJETIVOS:

- Profundizar en el diseño e implementación de algoritmos recursivos aplicados a listas y matrices.
- Consolidar el uso de recursividad como técnica de resolución de problemas en estructuras de datos.

Contexto

Se dispone de una lista simplemente enlazada implementada con una clase Lista que contiene un puntero al primer nodo (inicio) y un entero que representa la cantidad de elementos (cantidad):

```
public class Lista {
        private Nodo inicio;
        private int cantidad;
        //…….
}
```

```
public class Nodo {
        private int dato;
        private Nodo sig;

//Métodos de acceso y modificación disponibles
}
```

Se requiere implementar los siguientes métodos de instancia de forma **recursiva**:

Ejercicio 1

Implementar un método recursivo que muestre por consola **únicamente los números pares** de la lista, en **orden inverso** al de aparición.

Ej: para 34 - 5 - 44 - 4 - 17 debería mostrar: 4 - 44 - 34
Firma principal: **public void mostrarInverso()**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Ejercicio 2

Implementar un método recursivo que devuelva la suma de todos los elementos **pares** de la lista.
Firma principal: **public int sumaPares()**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Ejercicio 3

Implementar un método recursivo que determine si un valor dado se encuentra en la lista.
Firma principal: **public boolean estaValor(int valor)**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Ejercicio 4

Implementar un método recursivo que, dada otra lista representada por su primer nodo, determine si ambas listas son **idénticas** en tamaño y contenido (orden incluido).
Firma principal: **public boolean sonIguales(Nodo nodo)**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Ejercicio 5

Implementar un método recursivo que muestre los **primeros n elementos** de la lista en orden de aparición.
//pre: n <= cantidad
Firma principal: **public void mostrar(int n)**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Ejercicio 6

Implementar un método recursivo que determine si la lista se encuentra ordenada en forma **ascendente**.
Firma principal: **public boolean estaOrdenada()**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Contexto

Se propone resolver los siguientes ejercicios empleando recursividad. Se asume que las matrices están correctamente definidas y son cuadradas donde corresponda.

Ejercicio 7

Implementar un método recursivo que imprima los elementos de la **diagonal principal** de una matriz cuadrada.
Firma principal: **public void diagonalPrincipal(int mat[][])**
Nota: Defina una función recursiva auxiliar que sea invocada desde este método.

Ejercicio 8

Implementar un método recursivo que imprima los elementos de una columna determinada, comenzando por la **fila 0**.
Firma principal: **public void mostrarColumna(int mat[][], int col)**
Nota: Defina la función recursiva que se invocará

Ejercicio 9

Implementar un método recursivo que devuelva la suma de los valores contenidos en una fila determinada.
Firma principal: **public int sumaDeFila(int mat[][], int fila)**
Nota: Defina la función recursiva que se invocará

Ejercicio 10

Implementar un método recursivo que indique si **dos filas de una matriz** son iguales. Dos filas se consideran iguales si contienen los mismos elementos en las mismas posiciones.
Firma principal: **public boolean mismasFilas(int mat[][], int fila1, int fila2)**
Nota: Defina la función recursiva que se invocará