Ingeniería Civil en Computación e Informática
Ingeniería Civil Industrial
Ingeniería en Tecnologías de Información

Problema 1 (10 puntos)

Haz un programa en Java que lea un **número entero positivo** desde teclado y determine, usando **recursión**, si el número es **capicúa**.
(Un número capicúa es el que se lee igual de izquierda a derecha que de derecha a izquierda).

Ejemplos:

121 → es capicúa

3443 → es capicúa

123 → no es capicúa

No puede transformar el valor **entero** a otro tipo de variable. Implementar la función principal y la función recursiva.

Nombre de los métodos a implementar: **public static void main()**

public int capicua()

Problema 2 (15 puntos)

Utilizando **dividir para conquistar**, Dada una **matriz cuadrada de números enteros**, implemente un algoritmo recursivo que calcule el **producto de los elementos de la diagonal secundaria** que sean **divisibles por 3**.

Si **ningún elemento** cumple la condición, debe imprimir 1 como resultado.

- Tomar como guía las funciones ya creadas en la siguiente hoja.

Implemente la función correspondiente al enunciado-

Nombre de la función a implementar a implementar:

public int multiplicarDivisiblesPor3()

Diagonal Secundaria

$$\begin{pmatrix} 1 & 2 & 0 \\ 3 & 1 & 4 \\ 3 & 0 & 1 \end{pmatrix}$$

Problema 3 (15 puntos)

Considere la organización de datos de personas en un **árbol binario de búsqueda**, ordenado por **ID**.
Cada nodo contiene un ID y rendimiento.

Se solicita que implemente las funciones utilizando **recursión**:

- **Se pide implementar un método recursivo en Java que, dados dos IDs, encuentre su ancestro común más cercano, y luego calcule la suma de rendimiento de todos los nodos que pertenecen al subárbol enraizado en dicho ancestro.**

Nombre de las funciones a implementar: **public int sumaRendimientoDesdeAncestro(String id1, String id2)**

Tomar como guía las funciones ya creadas en la siguiente hoja.

Problema 4 (10 puntos) Rutee claramente el siguiente código Java, e indique lo que imprime.

```
public class Ruteo {
    public static void main(String[] args) {
        int k = 6;
        int valor = metodo(k, 1);
        System.out.println("valor " + valor);
    }

    public static int metodo(int m, int n) {
        if (m <= 0) {
            return 3 * (m + n);
        } else {
```

```
            return (3 + m) * metodo(m - 1, n);
        }
    }
}
```

Problema 2, código ya creado.

```
public static void main(String[] args) {

    int k = 3;

    int N = (int) Math.pow(2, k);

    int[][] matriz = generarMatriz(N);


    System.out.println("Matriz:");

    int resultado = dividirParaConquistar(matriz, 0, N - 1);

    System.out.println("Producto de los elementos divisibles por 3 en la diagonal
secundaria: " + resultado);

  }
```

Problema 3, código ya creado.

```
// Clase NodoEmpleado
class NodoEmpleado {
   String id;
   int rendimiento;
   NodoEmpleado left, right;


  public NodoEmpleado(String id, int rendimiento) {
      this.id = id;
      this.rendimiento = rendimiento;
      this.left = null;
      this.right = null;
   }
}
```

```
class ArbolEmpleado {
   NodoEmpleado root;


   public ArbolEmpleado() {
      root = null;
   }


   public void insertar(String id, int
rendimiento) {
      root = insertarRec(root, id,
rendimiento);
   }
```