

## Prueba Sumativa N.º 1

**Técnicas y Metodologías de Programación Avanzada– 30-04-2025**

Nombre			RUT	
--------	--	--	-----	--

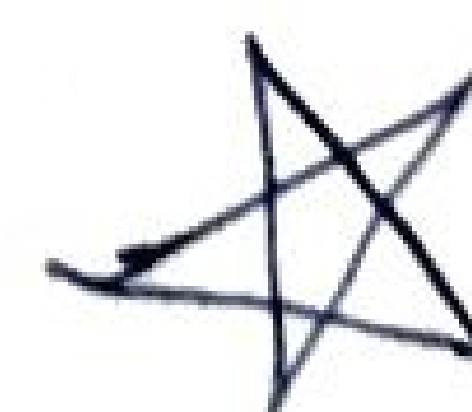
### Antecedentes generales:

Puntaje total de la prueba/Puntos para nota aprobatoria (4,0)	55 puntos 33 puntos	Puntaje Obtenido	
Duración de la prueba	1,5 horas	Nota final	
Resultados de Aprendizaje a evaluar	Aplicar estrategias avanzadas recursivas y de fuerza bruta para resolver problemas de ingeniería		
Fecha de entrega de resultados			

### Instrucciones:

1. Esta evaluación tiene 3 páginas (incluyendo la portada) y 4 preguntas. Compruebe que dispone de todas las páginas. Responda a las preguntas en los espacios previstos para ello en las hojas de preguntas. Asegúrese de marcar apropiadamente sus respuestas.
2. Durante la prueba no se puede utilizar: teléfono móvil, calculadora, apuntes y/o computador. Está prohibido intentar conectarse a internet de cualquier manera. Si es sorprendido obtendrá la calificación mínima. Tampoco puede utilizar dispositivos de almacenamiento externos o cualquier otro dispositivo relojes inteligentes, etc.
3. Lea la prueba completamente DOS veces antes de hacer cualquier pregunta
4. Una prueba respondida correctamente en un 60%, de acuerdo con las ponderaciones asignadas, corresponde a una nota 4,0.
5. Solamente se pueden realizar preguntas durante los primeros 10 minutos de la prueba. Solo se responderán preguntas respecto a los enunciados a viva voz.
6. La prueba es individual, cualquier sospecha de copia será calificada con la nota mínima y el caso será remitido al comité de ética.
7. En su espacio personal no debe haber nada más que hojas de papel en blanco, lápiz, goma y/o calculadora.
8. El resto de sus implementos debe guardarlos dentro de su mochila/bolso y ésta debe posicionarse al frente debajo de la pizarra. Si leyó hasta este punto, felicidades, para saber que lo hizo dibuje una estrella al final de esta página.

Acepto las condiciones firmando: \_\_\_\_\_



### Problema 1 (15 puntos)

Haz un programa en Java que lea un **número entero positivo** por pantalla y, usando recursión, calcule la **suma de sus dígitos**.

Ejemplos:

- $123 \rightarrow 1 + 2 + 3 = 6$
- $502 \rightarrow 5 + 0 + 2 = 7$
- $9 \rightarrow 9$

$$\begin{array}{r} 12 \\ 123 : 10 = 3 \\ 12 : 10 = 2 \\ 1 : 10 = \frac{1}{10} \end{array}$$

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: `main()`

`sumaDigitos()`

### Problema 2 (15 puntos)

Utilizando **dividir para conquistar**, se pide **multiplicar todos los números impares** de una lista. Se debe imprimir el resultado. Considere:

- Si no hay números impares en la lista, 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: `multiplicarImpares()`

### Problema 3 (15 puntos)

Considere la organización de datos de personas en un **árbol binario de búsqueda**, ordenado por **RUT**. Cada nodo contiene una persona con RUT, nombre y edad.

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

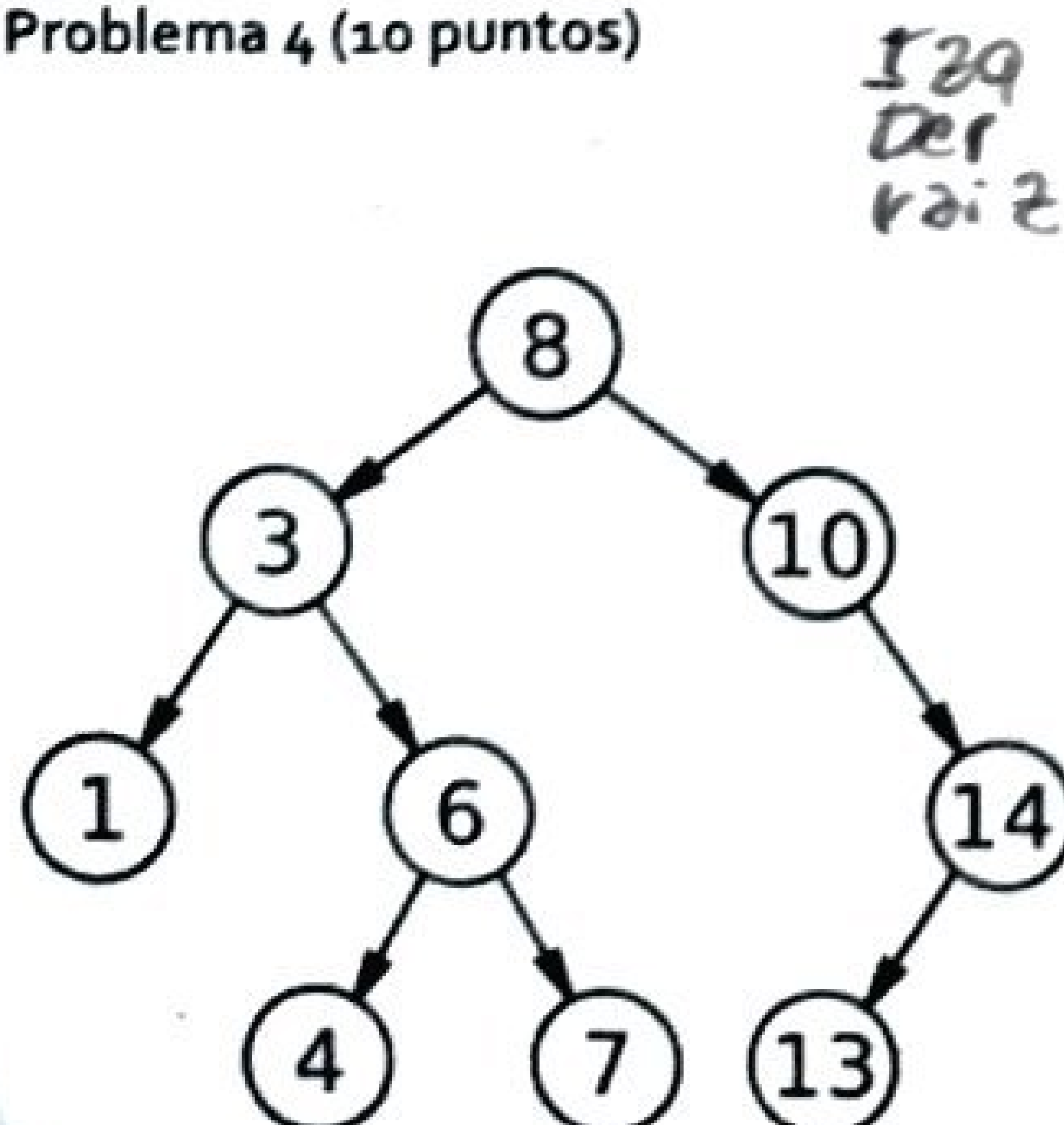
- Imprima la suma total de las edades de las personas que tienen exactamente un solo hijo (es decir, **nodos con un solo hijo izquierdo o derecho**).
- Inserte recursivamente los valores en el árbol.

Nombre de las funciones a implementar: `sumaEdadesUnSoloHijoRec(): int`

`insertarRec(): Nodo`

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

### Problema 4 (10 puntos)



Teniendo el siguiente árbol binario.  
Rutear el recorrido pos-orden, y mostrar el resultado de este.

Problema 2, código ya creado.

```
public static void main(String[] args) {  
    Random random = new Random();  
    int i = random.nextInt(9); // i entre 0 y 8  
    int n = (int) Math.pow(2, i);  
    int[] arreglo = new int[n];  
    // Llamar a la función recursiva  
    int resultado = multiplicarImpares(arreglo, 0, arreglo.length - 1);  
    System.out.println("Producto de los números impares: " + resultado);  
}
```

Problema 3, código ya creado.

```
// Clase Persona  
class Persona {  
    String rut;  
    String nombre;  
    int edad;  
  
    public Persona(String rut, String nombre, int edad)  
    {  
        this.rut = rut;  
        this.nombre = nombre;  
        this.edad = edad;  
    }  
}
```

```
class ArbolPersonas {  
    Nodo raiz;  
  
    public ArbolPersonas() {  
        raiz = null;  
    }  
}
```

```
// Clase Nodo  
class Nodo {  
    Persona persona;  
    Nodo izquierdo, derecho;  
  
    public Nodo(Persona persona) {  
        this.persona = persona;  
        izquierdo = derecho = null;  
    }  
}
```

```
// Insertar persona en el árbol  
public void insertar(Persona persona) {  
    raiz = insertarRec(raiz, persona);  
}  
  
// Método público para calcular suma de edades de  
nodos con un solo hijo  
public int sumaEdadesUnSoloHijo() {  
    return sumaEdadesUnSoloHijoRec(raiz);  
}  
  
//Las funciones pertenecen a la clase ArbolPersonas
```