**Algoritmos 2, Curso Mendez ~ 1er Recuperatorio, 1er Cuatrimestre 2025 ~ 2025-06-12**

Apellido y nombre: _____

Padrón: _____

| | | | | |
|---|---|---|---|---|
| Nota final: | | | | |

1) Explique qué es el **Teorema Maestro**, cómo se aplica y cuáles son las posibles soluciones. Justifique de la manera mas detallada posible la complejidad computacional de **Quicksort**.

**2)** Explique qué es un árbol **AVL** y muestre cómo funcionan los diferentes casos de inserción. Comenzando por un árbol vacío, inserte en orden (y muestre el estado luego de cada inserción) los elementos

**V = [8, 1, 3, 5, 7, 6, 9]**

**3)** Dado el **ABB** implementado en el cuatrimestre, agregue una función a la interfaz que permita iterar los elementos del **ABB** por niveles. Muestre y explique el código. Justifique la complejidad computacional.

**4)** Dados los siguientes recorridos de un mismo **ABB**, defina un algoritmo (no se pide el código) que reconstruya el **ABB** original. Muestre el árbol final y explique el procedimiento justificando el resultado.

**Preorden** = [#,$,@,+,■,●,X,▲] **Inorden** = [$,#,■,+,@,X,▲,●]

**5)** Escriba un algoritmo recursivo (sin utilizar **for**, **while**, **etc**) que recibe un string **texto** y un string **palabra** e imprime por pantalla la posición de cada ocurrencia de palabra dentro de texto. No se pueden utilizar funciones de la bilioteca estandar (excepto printf).

---

**Algoritmos 2, Curso Mendez ~ 1er Recuperatorio, 1er Cuatrimestre 2025 ~ 2025-06-12**

Apellido y nombre: _____

Padrón: _____

| | | | | |
|---|---|---|---|---|
| Nota final: | | | | |

1) Explique qué es el **Teorema Maestro**, cómo se aplica y cuáles son las posibles soluciones. Justifique de la manera mas detallada posible la complejidad computacional de **Quicksort**.

**2)** Explique qué es un árbol **AVL** y muestre cómo funcionan los diferentes casos de inserción. Comenzando por un árbol vacío, inserte en orden (y muestre el estado luego de cada inserción) los elementos

**V = [8, 1, 3, 5, 7, 6, 9]**

**3)** Dado el **ABB** implementado en el cuatrimestre, agregue una función a la interfaz que permita iterar los elementos del **ABB** por niveles. Muestre y explique el código. Justifique la complejidad computacional.

**4)** Dados los siguientes recorridos de un mismo **ABB**, defina un algoritmo (no se pide el código) que reconstruya el **ABB** original. Muestre el árbol final y explique el procedimiento justificando el resultado.

**Preorden** = [#,$,@,+,■,●,X,▲] **Inorden** = [$,#,■,+,@,X,▲,●]

**5)** Escriba un algoritmo recursivo (sin utilizar **for**, **while**, **etc**) que recibe un string **texto** y un string **palabra** e imprime por pantalla la posición de cada ocurrencia de palabra dentro de texto. No se pueden utilizar funciones de la bilioteca estandar (excepto printf).