Programación II, Algoritmos y Estructuras de Datos II

Examen Final Adelantado - 07/07/2025

Docente: Monzón, Nicolás Alberto .....

Apellido y Nombre: .....
L.U.: ..... **Firma:** .....

La firma de este documento implica la aceptación de las condiciones de aprobación abajo numeradas. Favor de verificar que cumpla con todas, en particular con los nombres de carpetas.

Requisitos para aprobar el examen:

1. Por actos de deshonestidad académica será sancionado.
2. Para poder aprobar este examen Ud. deberá estar administrativamente en condiciones de poder rendir el examen. En caso de no estarlo y rendir el examen, este no va a ser corregido y quedará anulado.
3. Deberá comprimir la carpeta `src` de su proyecto. El archivo comprimido deberá tener como nombre su APELLIDO y NOMBRE en ese orden. Deberá mandar el comprimido en formato `zip` al mail `nimonzon@uade.edu.ar`.
4. Deberá desarrollar un set de prueba para demostrar que funciona su código para los ejercicios 1 y 2. No es necesario escribir test unitarios, pero sí alguna prueba en el método `main` del proyecto.
5. Deberá escribir la estrategia de cada ejercicio y métodos que haga, y colocarla dentro de la carpeta `strategies` en formato `txt`, `MD` o `pdf`, cualquier otro formato no va a ser corregido.
6. En la firma de cada método escribir las precondiciones.
7. Solo podrá utilizar técnicas vistas en este curso. Esta prohibido el uso de librerías y genéricos, incluyendo estructuras que vienen con la JRE. No respetar este punto es suficiente para desaprobar el examen.
8. Este examen consta de 4 paginas.

9. El examen deberá ser entregado en el horario estipulado.

10. CONDICIONES PARA APROBAR: obtener al menos 60 puntos de los 100 puntos y cumplir con todos los ítems anteriores. En caso de no cumplir con alguno de los ítems anteriores, el examen queda desaprobado sin excepción.

Si el examen utiliza predicados, use la siguiente definición:

$$\mathcal{P}(n) = \text{Su LU, módulo 3, es } n$$

1. Definición, implementación y costos de los TDAs (%40)

Si usted cumple $\mathcal{P}(n)$. Modifique la implementación estática del TDA `Dictionary`, para que una clave pueda ahora tener hasta $n+1$ valores asociados. Si dada una clave se pide su valor, y el valor no es único, entonces devolver de forma aleatoria uno de los dos valores.

2. Utilización de los TDAs (%40)

Escribir un método que pida recibir un árbol binario con forma de matriz rotada. Verificar que el árbol representa una matriz identidad. Para hacerlo, recorrer el árbol binario de forma iterativa sin usar `Queue` o `PriorityQueue`. En la Figura 1, se muestra un ejemplo, pero el árbol recibido podría estar conectado de otra forma. Los nodos del árbol de ejemplo tiene valores que están en el mismo orden que uno recorrería de forma iterativa. Los valores en el árbol podrían venir de forma distinta, pero el recorrido debería matener este formato.

Cualquier dato que se necesite del árbol, ya sea total de nodos, altura, etc; tiene permitido obtenerse de forma recursiva.

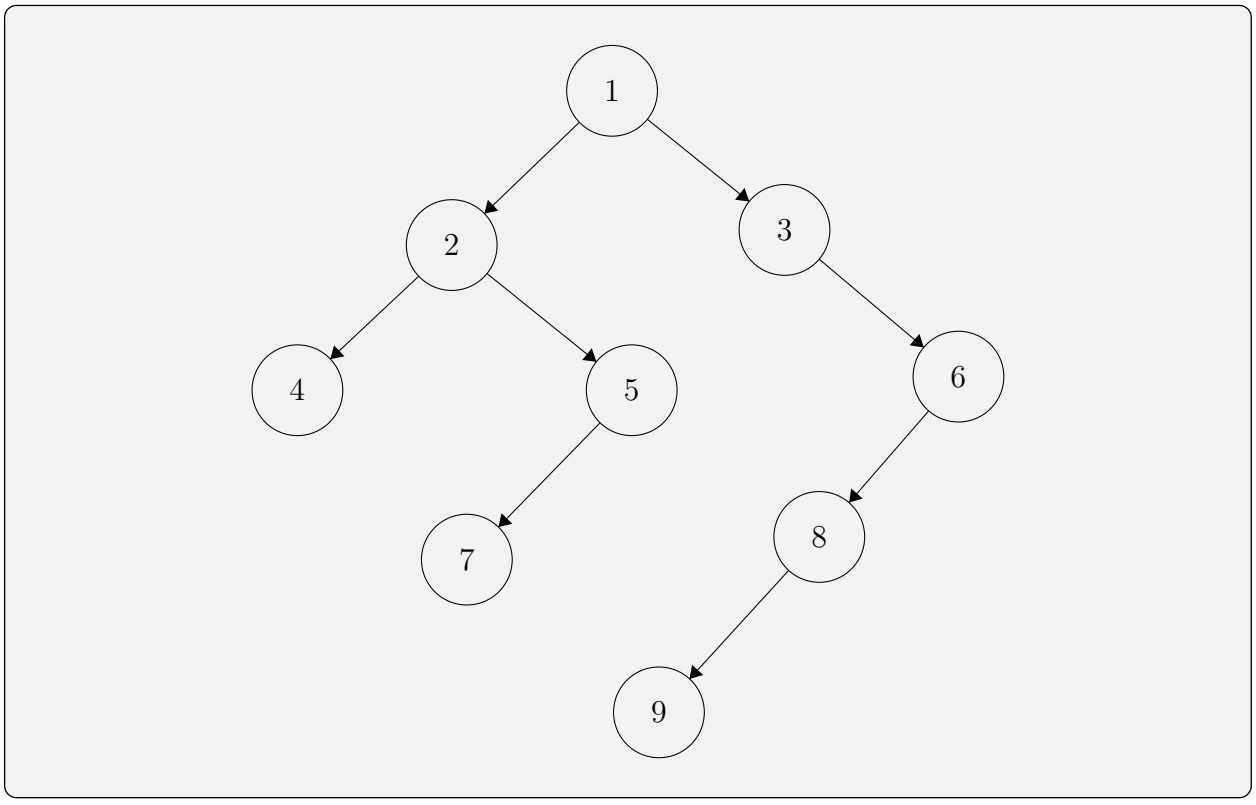

Figura 1: Ejemplo de un posible árbol con forma de matriz.

3. Teoría (%20)

Tenemos cuatro números α, β, γ y δ, tales que $\alpha > \beta < \gamma > \delta$, ¿Cuántos árboles binarios distintos pueden formarse que tengan estos números en sus nodos, y sabiendo que no hay otros nodos? ¿Cuántos podrían ser SBT? Dibujarlos.