Imagen que contiene Icono

Descripción generada automáticamente

**DESARROLLO DE SOFTWARE**

**GUÍA 3**

**AUTORES:**

MAYCKOLL ANDRÉS TORRES DIAZ.

**TUTOR**

DILSA ENITH TRIANA MARTÍNEZ

**17 DE MARZO DE 2025**

**INTRODUCCIÓN**En el presente informe se especifica y representa los scripts de calculo de complejidad, ejercicio de arboles y ejercicio de grafos, con el fin de dar a conocer todas y cada una de las decisiones para optimizar el código con algoritmos complejos.

**OBJETIVOS**

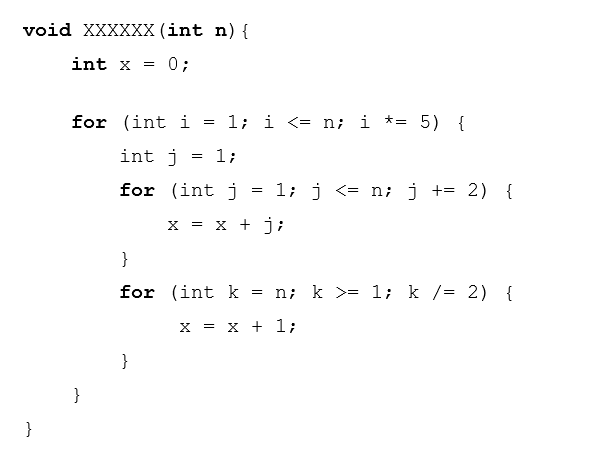
**Objetivo General:**

* Analizar y calcular la complejidad temporal del algoritmo proporcionado, identificando los diferentes bucles y su impacto en el rendimiento del código.

**Objetivos Específicos:**

* Identificar y calcular la complejidad temporal de cada bucle en el algoritmo (primer, segundo y tercer bucle).
* Determinar la combinación de las complejidades de los bucles para obtener la complejidad total del algoritmo.
* Explicar la relación entre el comportamiento del algoritmo y su rendimiento en función de su tamaño de entrada n

**Calcular Complejidad**



1. Se realiza la validación del script para lo cual tomamos como el primer bucle de los 3 que existen, en este caso es:  
     
   for (int i = 1; i <= n; i \*= 5)  
     
   Validación  
   tenemos una variable llamada i que es 1, y en cada paso del bucle, multiplica ese número por 5. En el primer paso, el número es 1, en el siguiente paso es 5, luego 25, luego 125, y así sucesivamente.

El bucle sigue ejecutándose hasta que ese número multiplicado por 5 sea mayor que un número que llamamos n.

El número de veces que el bucle se repite es cuántas veces puedes multiplicar ese número inicial 1 por 5 antes de que se haga más grande que n

Eso es lo que queremos calcular: cuántas multiplicaciones por 5 podemos hacer antes de que el número sea mayor que n.

for (int j = 1; j <= n; j += 2)

Validación

Este bucle comienza en j = 1 y avanza de 2 en 2 hasta que j alcanza n que es el numero que llamemos como n ejemplo ‘2-3-5-6 etc.’

for (int k = n; k >= 1; k /= 2)

Validación

Este código comienza en k = n y en cada iteración divide k entre 2, el número de iteraciones de este bucle es el número de veces que puedes dividir n entre 2 hasta que k sea menor que 1

1. Se realiza la validación de los 2 script para lo cual tomamos el segundo script como el indicado para este tipo de funciones.boolean Ex2(int[] a, int elem) {

int n = a.length;

int x = 0;

for (int i = 0; i < n; ++i) {

int pos = buscar(a, elem);

x += pos + 2;

for (int j = 0; j < n; ++j) {

if (a[j] > a[pos]) {

x++;

}

}

}

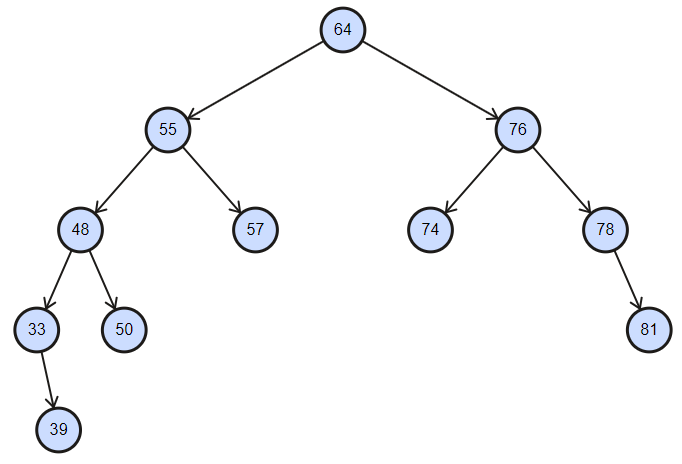
return x > elem;

}

Validación.  
  
la función buscar(a, elem) tiene complejidad de O(n log n), por lo tanto se elige el script de la derecha ya que, el primer bucle for (int i = 0; i < n; ++i) tiene complejidad de O(n), ya que se repite n veces.

Por lo tanto, el costo total del primer bucle junto con la función buscar(a, elem) y el segundo bucle anidado es:

O(n) \* O(n log n) + O(n) \* O(n) = O(n^2 log n) + O(n^2).

**Árboles binarios** Peso: 9 (total de nodos)

b. Altura: 5 (desde la raíz 64 hasta la hoja 81: 64 → 74 → 76 → 78 → 81)

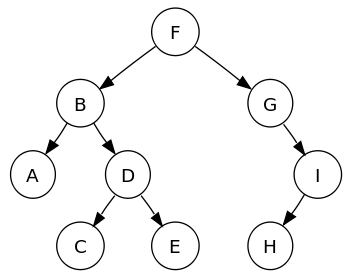
c. Hojas: 39, 57, 81

d. Una rama: 64 → 55 → 48 → 39

e. Recorrido en inorden: 39, 48, 55, 57, 64, 74, 76, 78, 81

f. Recorrido en preorden: 64, 55, 48, 39, 57, 74, 76, 78, 81

g. Recorrido en postorden: 39, 48, 57, 55, 81, 78, 76, 74, 64



a. Altura: 4  
b. Número de niveles: 4  
c. Ancestro común de E y A: B  
d. Peso del árbol izquierdo de F: 5 (nodos: B, A, D, C, E)  
e. Recorrido en inorden: A, B, C, D, E, F, G, H  
f. Recorrido en preorden: F, B, A, D, C, E, G, H  
g. Recorrido en postorden: A, C, E, D, B, H, G, F  
h. Recorrido por niveles: F, B, G, A, D, H, C, E  
i. Hojas: A, C, E, H

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | 1 |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  | 2 |  |  |  |  | 6 |  |
| 3 |  | 4 |  |  |  |  | 7 |
|  | 5 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Preorden: 1 – 2 – 3 – 4 – 5 – 6 – 7

Inorden: 3 – 2 – 5 – 4 – 1 – 6 - 7

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Postorden: A – C – E – D – B – H – I – G – F  Inorden: A – B – C – D – E – F – G – H – I | |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  | F |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  | B |  |  |  | G |  |  |  | | A |  |  | D |  |  |  | I |  |  | |  |  | C |  | E |  | H |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Preorden: 59 – 37 – 28 – 16 – 43 – 48 – 74 – 80 – 79  Inorden: 16 – 28 – 37 – 43 – 48 – 59 – 74 – 79 – 80 | |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  | 59 |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  | 37 |  |  |  | 74 |  |  |  |  | | 28 |  | 43 |  |  |  | 80 |  |  |  | | 16 |  | 48 |  |  |  | 79 |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |



* 1. Encuentre (paso a paso) los costos mínimos de los caminos que parten del vértice **A** usando el algoritmo de Dijkstra

Iteración 1:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 4 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |
| Anterior: | - | A | A | - | - | - | - | - | - |

Iteración 2:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***G*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 6 | ∞ | ∞ | ∞ | ∞ | ∞ |
| Anterior: | - | A | B | B | - | - | - | - | - |

Iteración 3:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 4 | 6 | 10 | ∞ | ∞ | ∞ |
| Anterior: | - | A | B | C | C | D | - | - | - |

Iteración 4:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 4 | 6 | 10 | ∞ | ∞ | ∞ |
| Anterior: | - | A | B | C | C | D | - | - | - |

Iteración 5:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 4 | 6 | 7 | ∞ | ∞ | ∞ |
| Anterior: | - | A | B | C | C | E | - | - | - |

Iteración 6:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 4 | 6 | 7 | 9 | ∞ | ∞ |
| Anterior: | - | A | B | C | C | E | F | - | - |

Iteración 7:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 4 | 6 | 7 | 9 | 12 | ∞ |
| Anterior: | - | A | B | C | C | E | F | G | - |

Iteración 8:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Vértice** | **Seleccionados** |  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***F*** | ***G*** | ***H*** | ***I*** |
|  |  | Costo | 0 | 1 | 3 | 4 | 6 | 7 | 9 | 12 | 13 |
| Anterior: | - | A | B | C | C | E | F | G | H |

* 1. ¿Cuál es el camino entre el vértice A y el vértice I y cuánto es el costo de tal camino, de acuerdo con el algoritmo de Dijkstra?

Camino más corto

El camino más corto de A a I es:

- A → B → C → D → I

Costo total

El costo total del camino es 7.

Por lo tanto, el camino entre el vértice A y el vértice I es A -B -C -D -I y el costo es 7.

LINK GITHUB: <https://github.com/mehirakiva/EAN_guia3.git>

**CONCLUSIONES**

Esta arquitectura de software con base en arboles y algoritmos permite las adaptabilidad y actualización del software con el objetivo principal de crear una aplicación escalable, adaptable y confiable guiados por patrones de diseño de software.