**Imagen que contiene flor

Descripción generada automáticamente**

**FACULTAD DE ESTUDIOS EN AMBIENTES VIRTUALES**

**GUÍA 3**

**DESARROLLO DE SOFTWARE**

**AUTOR**

**HAROLD MAURICIO CAÑAVERAL ROMERO**

**JUNIOR CASTAÑEDA**

**TUTOR**

**LUIS ARMANDO COBO CAMPO**

**BOGOTÁ D.C., 22 DE FEBRERO DE 2021**

**Tabla De Contenido**

[1. INTRODUCCIÓN ACTIVIDA 1 3](#_Toc67334045)

[2. OBJETIVO GENERAL ACTIVIDA 1 4](#_Toc67334046)

[3. OBJETIVOS ESPECIFICOS ACTIVIDA 1 4](#_Toc67334047)

[Actividad 1 Estructura de datos en Java 5](#_Toc67334048)

[4. INTRODUCCIÓN ACTIVIDA 2 7](#_Toc67334049)

[5. OBJETIVO GENERAL ACTIVIDA 2 8](#_Toc67334050)

[6. OBJETIVOS ESPECIFICOS ACTIVIDA 2 8](#_Toc67334051)

[Actividad 2 Estructuras de datos lineales enlazadas 9](#_Toc67334052)

[7. INTRODUCCIÓN ACTIVIDA 3 12](#_Toc67334053)

[8. OBJETIVO GENERAL ACTIVIDA 3 13](#_Toc67334054)

[9. OBJETIVOS ESPECIFICOS ACTIVIDA 3 13](#_Toc67334055)

[Actividad 3 El TAD Pila y el TAD Cola 14](#_Toc67334056)

[Sol. 14](#_Toc67334057)

[10. CONCLUSIONES 17](#_Toc67334058)

[11. BIBLIOGRAFIA 18](#_Toc67334059)

# INTRODUCCIÓN ACTIVIDA 1

Las estructuras de datos son el mecanismo que se le proporciona al programador de computadores para manipular grandes cantidades de datos de manera eficiente.

# OBJETIVO GENERAL ACTIVIDA 1

Realizar una revisión concienzuda de la funcionalidad de diversas estructuras de datos, así como dominar los mecanismos de medición de la complejidad de los algoritmos que trabajan sobre estas estructuras de datos.

# OBJETIVOS ESPECIFICOS ACTIVIDA 1

* Diseña e implementa estructuras de datos en memoria principal necesarias para resolver un problema teniendo en cuenta un conjunto de restricciones y criterios de calidad.
* Aplica técnicas de desarrollo de algoritmos para la solución de problemas, analizando su complejidad en tiempo y en espacio.

## Actividad 1 Estructura de datos en Java

1. Para cada uno de los tiempos que toma un algoritmo en terminar, hallar el orden de complejidad correspondiente:

4. Calcule la complejidad del siguiente algoritmo

**void** XXXXXX(**int n**)

{

**int** x = 0**;**

**for (**int i = 1; i <= n; i += 5)

{

int j = 1;

while (j <= n)

{

x = x + j;

j \*= 4

}

**for (int k = n; k >= 1; k -= 2**)

{

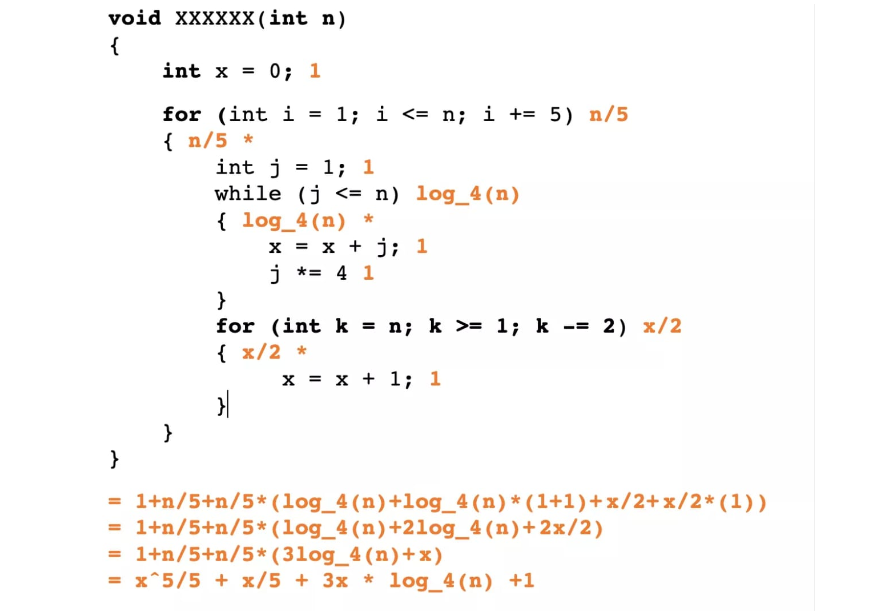
x = x + 1;

}

}

}

**SOL.**



1. (10%) Suponga que se tienen los siguientes dos algoritmos para resolver el mismo problema, y suponga que la función **buscar** tiene complejidad .

|  |  |
| --- | --- |
| **boolean** Ex1(int[] a, int elem) {  **int** pos = *buscar*(a, elem);  **int** n = a.length;  **int** x = pos;    **for** (int i = 0; i < n; ++i) {  x += 2;  **for** (int j = 0; j< n; ++j) {  **if** (a[j] > a[pos]) {  x++;  }  }  }  **return** x > elem; } | **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; } |

Indique con cuál de los dos algoritmos se queda para resolver el problema. Justifique muy bien su respuesta.

El segundo algoritmo es la correcta, la función buscar está por fuera del for lo que hace que la complejidad sea menor con respecto al otro algoritmo

# INTRODUCCIÓN ACTIVIDA 2

Las estructuras enlazadas de objetos son para modelar aplicaciones que manipulan cantidades dinámicas de información.

# OBJETIVO GENERAL ACTIVIDA 2

Solucionar la petición de un cliente que pide desarrollar un programa de computador para administrar la información de un grupo de pacientes que son atendidos en distintas clínicas de la ciudad.

# OBJETIVOS ESPECIFICOS ACTIVIDA 2

* Código fuente del proyecto «Central de pacientes», desarrollado en el lenguaje de programación Java usando los conceptos de nodos y listas sencillamente encadenadas. El proyecto busca solucionar la petición de un cliente que pide desarrollar un programa de computador para administrar la información de un grupo de pacientes que son atendidos en distintas clínicas de la ciudad.

## Actividad 2 Estructuras de datos lineales enlazadas

 Realizar los dos métodos de extensión que se encuentran al final de la clase «CentralPacientes»

**Sol**

**https://github.com/Hcanave33508/Desarrollo-de-Sofware**

/\*\*

\* Retorna la cantidad de hombres que hay en la lista

\*/

public int cantHombres() {

// No ubicamos al inicio de la lista

Paciente actual = primero;

// Inicializamos el contador

int contador = 0;

// Recorremos la lista de pacientes

while (actual != null) {

// Comprobamos si el paciente es hombre, dado el caso aumentamos en 1 el contado

if (actual.darSexo()==Paciente.HOMBRE)

contador++;

// Pasamos al siguiente paciente

actual = actual.darSiguiente();

}

// Retornamos la cantidad mujeres en al lista

return contador;

}

/\*\*

\* Retorna la cantidad de mujeres que hay en la lista

\*/

public int cantMujeres() {

// No ubicamos al inicio de la lista

Paciente actual = primero;

// Inicializamos el contador

int contador = 0;

// Recorremos la lista de pacientes

while (actual != null) {

// Comprobamos si el paciente es mujer, dado el caso aumentamos en 1 el contado

if (actual.darSexo()==Paciente.MUJER)

contador++;

// Pasamos al siguiente paciente

actual = actual.darSiguiente();

}

// Retornamos la cantidad mujeres en al lista

return contador;

}

# INTRODUCCIÓN ACTIVIDA 3

La importancia y utilidad de las estructuras de datos lineales dinámicas Pila (Stack) y Cola (Queue) en desarrollo de compiladores de lenguajes de programación.

# OBJETIVO GENERAL ACTIVIDA 3

El proyecto permite tomar una expresión aritmética (en notación postfija), y a través de la utilización de pilas, obtener el resultado numérico de la evaluación de esa expresión.

# OBJETIVOS ESPECIFICOS ACTIVIDA 3

* Importe el proyecto a su ambiente de desarrollo de su preferencia, sea Eclipse o IntelliJ o NetBeans. Revise la clase «EvaluadorPostfijo» que se encuentra en el paquete «universidadean. desarrollosw. postfijo» que se encuentra dentro de la carpeta «src» del proyecto.

## Actividad 3 El TAD Pila y el TAD Cola

 Su misión consiste en escribir el método «evaluarPostFija» que se encuentra dentro de esa clase. Este método realiza la evaluación de la expresión postfija utilizando una pila y produciendo el resultado final.

## Sol.

<https://github.com/Hcanave33508/Desarrollo-de-Sofware>

static int evaluarPostFija(List<String> expresion) {

// TODO: Realiza la evaluación de la expresión en formato postfijo

// Inicializamos la pila

Stack<Integer> pila = new Stack<>();

// Array para almacenar los operadores a utilizar

int numero[] = { 0, 0 };

// Recorremos la expresion

for (int i = 0; i < expresion.size(); i++) {

// Tratamos de guardar los numeros en la pila

try {

pila.push(Integer.parseInt(expresion.get(i)));

} catch (Exception e) {

// En caso de no ser posible guardar el numero, significa que es un operador por lo

// tanto procedemos a realizar la operacion

// Cargamos los dos ultimos elementos de la pila

numero[1] = pila.pop();

numero[0] = pila.pop();

// Validamos que operacion se desea realizar

switch (expresion.get(i)) {

case "+":

// Agregamos el resultado de la suma a la pila

pila.push(numero[0] + numero[1]);

break;

case "-":

// Agregamos el resultado de la resta a la pila

pila.push(numero[0] - numero[1]);

break;

case "\*":

// Agregamos el resultado de la multiplicacion a la pila

pila.push(numero[0] \* numero[1]);

break;

case "/":

// Agregamos el resultado de la division a la pila

pila.push(numero[0] / numero[1]);

break;

}

}

}

// Retornamos el ultimo elemento de la pila

return pila.pop();

}

# CONCLUSIONES

* La manera en que se colocan los elementos dentro de la **estructura** afectará la forma en que se realicen los accesos a cada elemento.
* Pueden descomponerse en los elementos que la forman.
* La colocación de los elementos y la manera en que se accede a ellos puede ser encapsulada.

# BIBLIOGRAFIA

Villalobos, J. A. (2008). *Introducción a las estructuras de datos: aprendizaje activo basado en casos*. Nivel 3, páginas 175-206. Bogotá, Colombia: Pearson

Villalobos S., J. A. (1996). *Diseño y Manejo de Estructura de Datos en C.*Capítulo 0, sección 0.3 y Capítulo 1, 2, 3, 4 y 6 Bogotá, Colombia: McGraw-Hill. Recuperado de [https://bit.ly/EDCVillalobos (Enlaces a un sitio externo.)](https://bit.ly/EDCVillalobos)

Weiss, M. A. (2012). *Data structures and algorithm analysis in Java (3rd edition).*Capítulo 2, 3, 4 y 9 New Jersey, EEUU: Pearson.