
**Nombre completo:** Nathan Esquivel Mendez

**Carrera:** Bachillerato en Desarrollo de Software

**Curso:** Estructuras de datos

**Código del curso:** SOFT-10

**Ciclo**: C-III 2025

---

# 1. Introducción

Las pilas son una estructura de datos muy utilizada en programación. Su funcionamiento se basa en el principio **LIFO (Last In, First Out)**, que significa “el último en entrar es el primero en salir”. Esto quiere decir que el último elemento que se agrega a la pila será el primero en eliminarse.

En Java, una pila puede implementarse de diferentes formas, como usando arreglos, listas enlazadas o la clase **Stack** que ya viene incluida en la biblioteca del lenguaje. Sin embargo, crear una pila desde cero ayuda a entender mejor cómo funcionan sus operaciones básicas: **push** (agregar un elemento), **pop** (quitar un elemento), **peek** (ver el elemento superior) y **isEmpty** (verificar si está vacía).

Aprender a implementar pilas es importante porque se usan en muchos procesos de los programas, cómo deshacer acciones, evaluar expresiones matemáticas o manejar llamadas de funciones.

# 2. Desarrollo

## 2.1. Por qué una pila es la estructura de datos adecuada para la implementación de la funcionalidad seleccionada (Análisis de cadenas de impresión):

Una pila es la estructura ideal para el **análisis de cadenas de impresión** porque trabaja con el principio LIFO (Last In, First Out), lo que permite procesar los elementos en orden inverso al que fueron ingresados. Esto es útil para verificar el orden correcto de símbolos, como paréntesis, llaves o etiquetas, que deben cerrarse en el mismo orden en que se abrieron.

Al analizar una cadena, la pila guarda los símbolos de apertura y comprueba que cada cierre sea el correcto. Así se puede **validar la estructura de la cadena** antes de imprimirla o procesarla.

En resumen, la pila facilita el control del orden, la detección de errores y el manejo de estructuras anidadas, por lo que es la mejor opción para esta funcionalidad.

## 2.2. Comentario sobre la seguridad, orden y eficiencia que una pila aporta a la realización de la funcionalidad seleccionada:

El uso de una pila en el análisis de cadenas de impresión aporta **seguridad**, porque permite verificar que cada símbolo o elemento tenga su apertura y cierre correcto, evitando errores en la estructura de la cadena. También ofrece **orden**, ya que los datos se procesan de forma controlada siguiendo el principio LIFO, lo que garantiza que las operaciones se realicen en el orden adecuado.

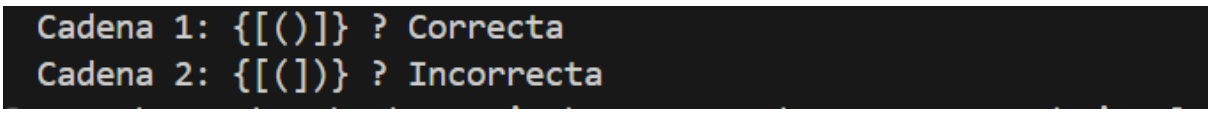
Además, la pila brinda **eficiencia**, ya que sus operaciones básicas, como agregar o eliminar elementos, se realizan en tiempo constante, sin necesidad de recorrer toda la estructura. Esto permite analizar cadenas de forma rápida y confiable, manteniendo la integridad del proceso.

## 2.3 Descripción de un ejemplo concreto en software de sistema en el que la funcionalidad seleccionada está implementada, acompañada de imágenes que ilustran sus resultados y comportamiento.

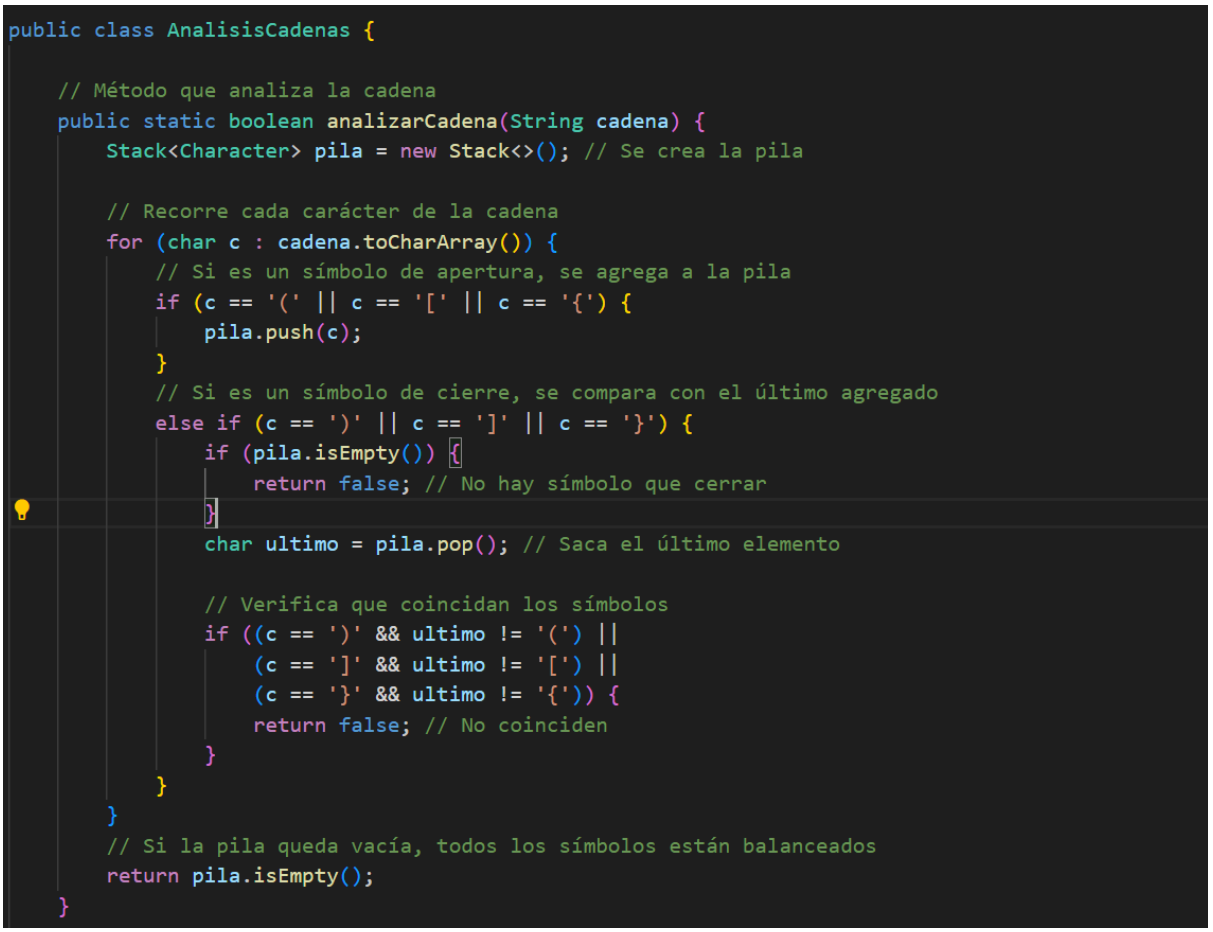
Un ejemplo concreto donde se implementa la funcionalidad de **análisis de cadenas de impresión con pilas** se encuentra en los **compiladores y editores de texto**. Estos programas utilizan pilas para verificar la correcta apertura y cierre de símbolos como paréntesis, llaves y corchetes en el código fuente.

Por ejemplo, cuando un programador escribe en un entorno como Eclipse o Visual Studio Code, el sistema analiza cada carácter que se ingresa. Si se abre un símbolo como “{”, este se almacena en una pila; cuando se escribe “}”, el programa revisa que coincida con el último símbolo abierto. Si no hay coincidencia o falta un cierre, el editor muestra un error de sintaxis o un mensaje de advertencia.

Este comportamiento garantiza orden y seguridad en la estructura del código, evitando fallos al momento de compilar o ejecutar el programa. En el caso del ejemplo en Java, el sistema revisa una cadena y muestra si está correctamente balanceada o incorrecta, simulando la misma lógica que usan los compiladores reales.



2.4 Representación gráfica de las operaciones de una pila que implementa la funcionalidad seleccionada:



```java
public class AnalisisCadenas {

    // Método que analiza la cadena
    public static boolean analizarCadena(String cadena) {
        Stack<Character> pila = new Stack<>(); // Se crea la pila

        // Recorre cada carácter de la cadena
        for (char c : cadena.toCharArray()) {
            // Si es un símbolo de apertura, se agrega a la pila
            if (c == '(' || c == '[' || c == '{') {
                pila.push(c);
            }
            // Si es un símbolo de cierre, se compara con el último agregado
            else if (c == ')' || c == ']' || c == '}') {
                if (pila.isEmpty()) {
                    return false; // No hay símbolo que cerrar
                }
                char ultimo = pila.pop(); // Saca el último elemento

                // Verifica que coincidan los símbolos
                if ((c == ')' && ultimo != '(') ||
                    (c == ']' && ultimo != '[') ||
                    (c == '}' && ultimo != '{')) {
                    return false; // No coinciden
                }
            }
        }
        // Si la pila queda vacía, todos los símbolos están balanceados
        return pila.isEmpty();
    }
```

# 3. Conclusión

La implementación de pilas en Java permite comprender de forma práctica cómo una estructura de datos sencilla puede resolver problemas complejos de manera eficiente. En el caso del **análisis de cadenas de impresión**, la pila resulta fundamental para mantener el **orden, la seguridad y la coherencia** de los símbolos que se abren y cierran dentro de una secuencia.

Gracias a su principio LIFO (Last In, First Out), la pila ofrece una forma clara y ordenada de procesar la información, garantizando que los elementos se manejan en el orden correcto. Esta estructura no solo se aplica en el análisis de cadenas, sino también en otras áreas del software, como la compilación de código, el control de llamadas a funciones y la gestión de memoria.

En conclusión, aprender a implementar y aplicar pilas en Java fortalece las bases del pensamiento lógico y algorítmico, siendo una herramienta esencial en la construcción de programas confiables, seguros y bien estructurados.

# 4. Referencias bibliográficas:

1. Albarrán, J. (2023). *Estructuras de datos básicas: Listas, pilas, colas y matrices* [E book]. Recuperado de https://www.amazon.com/-/es/Jesus-Albarran-ebook/dp/B0BT428GJK Amazon
2. Universidad de Valencia, Departamento de Informática. (s. f.). *Estructura de datos – Capítulo "Pila"* [PDF]. Recuperado de https://informatica.uv.es/docencia/fguia/TI/Libro/PDFs/CAPI5.pdf Departament d'Informàtica
3. Instituto Politécnico Nacional, ESCOM. (2013). *Estructuras de datos* [PDF]. Recuperado de https://www.escom.ipn.mx/docs/oferta/matDidacticoISC2009/EDts/Libro_EstructuraDatos.pdf