PRÁCTICA 03: Introducción - Implementación comparativa de estructuras de datos fundamentales en C++ y Python

Curso: Algoritmos y Estructuras de Datos
Código: SIS210
Ciclo: IV
Docente: Mg. Aldo Hernán Zanabria Gálvez
Duración: 3 horas
Fecha de desarrollo: Semana 3 – 2025-I

I. Objetivo general

Implementar y analizar las estructuras de datos lineales fundamentales (pila, cola y lista enlazada simple) utilizando los lenguajes de programación C++ y Python, comprendiendo su funcionamiento, sus ventajas y limitaciones, y promoviendo la capacidad crítica de evaluación algorítmica entre entornos compilados y de scripting.

II. Objetivos específicos

1. Comprender los principios de diseño y comportamiento de estructuras de datos lineales.
2. Implementar pilas, colas y listas enlazadas simples en C++ y Python.
3. Comparar los enfoques estructurales entre un lenguaje de bajo nivel (C++) y uno de alto nivel (Python).
4. Analizar el uso de memoria, la sintaxis y la facilidad de mantenimiento en ambas implementaciones.
5. Detectar posibles errores lógicos o limitaciones prácticas en cada implementación.

III. Marco teórico

Las estructuras de datos lineales son herramientas esenciales para la organización y manipulación eficiente de información en memoria. Las más fundamentales son las pilas, colas y listas enlazadas simples.

- **Pila (stack)**: estructura tipo LIFO (Last In, First Out), donde solo se permite insertar o eliminar elementos desde el tope.
- **Cola (queue)**: estructura tipo FIFO (First In, First Out), en la que los elementos se insertan por el final y se eliminan por el frente.
- **Lista enlazada simple**: colección dinámica de nodos conectados entre sí, donde cada nodo apunta al siguiente.

C++ proporciona control detallado de memoria y estructuras estáticas y dinámicas mediante punteros, siendo ideal para desarrollar desde cero estructuras como listas enlazadas y pilas con arreglos. **Python**, al ser un lenguaje interpretado y de alto nivel, facilita la implementación usando clases y listas dinámicas predefinidas, reduciendo el control explícito de memoria, pero con mayor facilidad para prototipado.

IV. Actividades prácticas

El estudiante deberá realizar las siguientes actividades en ambos lenguajes y documentar las diferencias observadas:

Parte A: Implementación de una pila

- Operaciones: insertar (push), eliminar (pop), mostrar el tope, recorrer.
- Insertar los elementos: 5, 10, 15, 20, 25.
- Eliminar dos elementos consecutivos.
- Mostrar el estado final.

Parte B: Implementación de una cola

- Operaciones: insertar (enqueue), eliminar (dequeue), mostrar el frente, recorrer.
- Insertar los elementos: 3, 6, 9, 12.
- Eliminar un elemento.
- Mostrar la cola final.

Parte C: Implementación de una lista enlazada simple

- Operaciones: insertar al inicio, insertar al final, eliminar del inicio, mostrar lista.
- Insertar al inicio los valores: 8, 4.
- Insertar al final el valor: 11.
- Eliminar el primer nodo.
- Mostrar el recorrido completo.

V. Comparación técnica esperada

Aspecto evaluado	C++	Python
Control de memoria	Explícito (punteros, `new`, `delete`)	Implícito (recolección automática de basura)
Estructura de código	Más detallada y extensa	Más concisa y abstracta
Flexibilidad de listas	Requiere implementación manual	Uso directo de listas dinámicas (`list`)
Complejidad de implementación	Alta para estructuras dinámicas	Baja, pero menos transparente internamente
Tipado de variables	Estático	Dinámico