**GUÍA DE LABORATORIO NO. 6**

**HANNA K. ABRIL**

**JUAN BOCANEGRA**

**SANTIAGO MORENO**

**UNIVERSIDAD MANUELA BELTRÁN**

**GERENCIA DE PROYECTOS**

**DOCENTE**

**HUGO ALFONSO ORTIZ BARRERO**

**BOGOTA DC 21 ABRIL 2025**

**Preguntas**

1. **¿Qué diferencias encuentra entre un arreglo y una lista Enlazada?**

|  |  |  |
| --- | --- | --- |
| Característica | Arreglo | Lista Enlazada |
| Tamaño | Fijo (se define al crearlo) | Dinámico (puede crecer o reducirse) |
| Accesos a elementos | Acceso directo por índice (rápido) | Acceso secuencial (más lento) |
| Inserción/Eliminación | Costosa (hay que mover elementos) | Eficiente si se conoce el nodo previo |
| Uso de memoria | Contiguo (menos memoria extra) | No contiguo (requiere punteros extra) |
| Eficiencia | Mejor para lectura aleatoria | Mejor para inserciones/ eliminaciones frecuentes |

1. **¿Cuáles son las diferencias entre las clases ArrayList, LinkedList y Vector utilizadas en Java?**

|  |  |  |  |
| --- | --- | --- | --- |
| Característica | ArrayList | LinkedList | Vector |
| Estructura interna | Arreglo dinámico | Lista doblemente enlazada | Arreglo dinámico |
| Accesos aleatorios | Rápido (acceso por el índice) | Lento (recorrido secuencial) | Rápido (como ArrayList) |
| Inserción/Eliminación | Lenta si es en el medio | Rápida en medio/inicio | Lenta (sincronización) |
| Sincronización (Hilos) | No sincronizado (no thread-safe) | No sincronizado | Sincronizado (thread-safe) |
| Uso común | Ideal para listas generales | Ideal para pilas, colas, listas | Poco usado (remplazado por otros) |
| Crecimiento automático | Incremento del 50% | Dinámico según nodos | Doble de su tamaño |

1. **Identifique 5 aplicaciones de las listas circulares en el ámbito de su carrera**

* Planificación de procesos en sistemas operativos: Se usan en algoritmos como Round Robin para gestionar el uso del CPU entre múltiples procesos de forma cíclica.
* Reproducción de medios (listas de reproducción): En reproductores de música o video que repiten una lista de canciones indefinidamente.
* Sistemas de monitoreo en tiempo real: Como en aplicaciones IoT donde los datos del sensor se sobreescriben de forma cíclica cuando se llena el buffer.
* Juegos de turnos (como tableros): Para manejar el cambio de turno entre jugadores de forma continua y automática.
* Manejo de buffers circulares: En redes o comunicaciones para almacenar temporalmente datos que entran y salen en flujo constante.

1. **Explique claramente en qué consiste una pila y como es su funcionamiento.**

Una pila es una estructura de datos lineal que almacena elementos uno encima del otro, como una torre de bloques o una pila de platos.

Sigue el principio LIFO:

**Last In, First Out →** El último en entrar es el primero en salir.

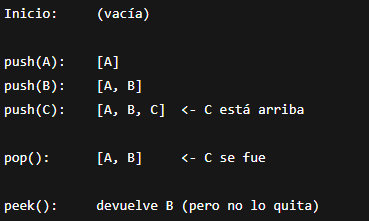
Imagina una caja donde solo puedes poner y sacar cosas por la parte de arriba.  
Hay dos operaciones principales:

1. **push()** – Agregar (insertar) un nuevo elemento en la parte superior.  
   Ejemplo: si tienes la pila [A, B] y haces push(C), la pila queda así: [A, B, C]
2. **pop()** – Quitar (eliminar) el elemento que está en la parte superior.  
   Si haces pop() sobre [A, B, C], el resultado es [A, B] y se elimina C.

También hay una tercera operación opcional:

* **peek()** o **top()** – Ver cuál es el elemento superior sin eliminarlo.

**Representación visual:**



**¿Dónde se usa?**

* En programación, cuando una función llama a otra, el sistema guarda el estado en una pila de llamadas.
* Cuando haces deshacer en un editor de texto, las acciones se guardan en una pila.
* Al evaluar expresiones matemáticas, como (3 + (2 \* 5)), se usa una pila para resolverlas correctamente.

1. **Describa 3 aplicaciones de las pilas en su vida como ingeniero.**

* Control de llamadas de funciones (call stack): Cuando se invocan funciones recursivas o anidadas, el sistema usa una pila para guardar el estado de cada función.
* Deshacer/rehacer en editores de código o texto: Las acciones se guardan en una pila, permitiendo retroceder (undo) y avanzar (redo) en los cambios hechos.
* Evaluación de expresiones matemáticas o sintácticas: En compiladores o intérpretes, se usa una pila para convertir expresiones infijas a postfijas y para evaluar resultados.

1. **Describa 3 aplicaciones de las colas en su vida como ingeniero.**

Una cola es una estructura de datos que funciona con el principio FIFO (First In, First Out) — es decir, el primero en entrar es el primero en salir.

Aplicaciones:

* Gestión de tareas en impresión: Cuando varios usuarios mandan a imprimir documentos, estos se encolan en el orden de llegada y se procesan uno a uno.
* Procesamiento de solicitudes en servidores web: Las peticiones que llegan a un servidor se colocan en una cola para ser atendidas en orden, garantizando equidad y eficiencia.
* Simulación de colas en proyectos de software (sistemas bancarios, atención al cliente, etc.): Puedes usar colas para modelar procesos reales donde las personas esperan su turno para ser atendidas.

1. **Preguntas Orientadoras**

* ¿Cuáles fueron los aprendizajes obtenidos al realizar esta guía?, liste como mínimo 3 aprendizajes y relaciónelos con su futuro que hacer profesional.

**Comprensión de estructuras de datos**:  
Aprendí a diferenciar entre arreglos, listas enlazadas, pilas y colas, y entendí cómo se comportan y cuándo usarlas.  
*Esto es clave para desarrollar software eficiente y bien estructurado.*

**Aplicación de estructuras en problemas reales**:  
Pude conectar conceptos teóricos con casos prácticos como servidores, reproductores o sistemas de turnos.  
*Esto fortalece mi capacidad para modelar soluciones tecnológicas aplicadas a la vida cotidiana.*

**Análisis comparativo entre clases de Java**:  
Comparar ArrayList, LinkedList y Vector me enseñó a elegir la clase más adecuada según el contexto.  
*Esto me prepara para tomar decisiones técnicas fundamentadas en mis futuros proyectos.*

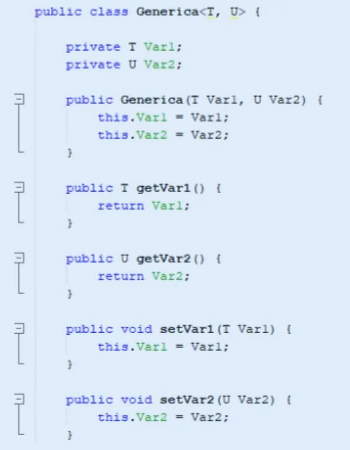
* ¿Dónde presento mayor dificultad resolviendo la guía? y como lo resolvieron cuales fueron las estrategias de solución?

**Dificultad:**  
La mayor dificultad fue entender las diferencias prácticas entre las clases ArrayList, LinkedList y Vector, ya que son similares en varios aspectos, pero tienen diferencias técnicas importantes.

**Estrategia de solución:**

* Busqué ejemplos prácticos y los ejecuté en código Java para ver el comportamiento.
* Consulté documentación oficial y resúmenes gráficos que me ayudaron a visualizar las diferencias.
* También pregunté aquí (¡gracias ChatGPT!) para tener una explicación clara y adaptada a mi contexto.

1. **Describa 3 aplicaciones de las colas en su vida como ingeniero.**

****

Este es un ejemplo de una clase genérica en Java, llamada Generica<T, U>, que permite manejar dos tipos de datos genéricos, uno representado por T y otro por U.

Atributos:



* Var1 puede ser de cualquier tipo de dato (Integer, String, etc.).
* Var2 también puede ser de cualquier tipo diferente o igual a Var1.

Métodos del programa:

* 1. Constructor:



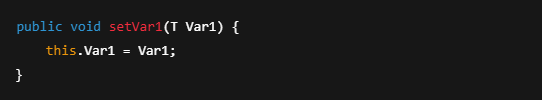
* 1. Getter Var1



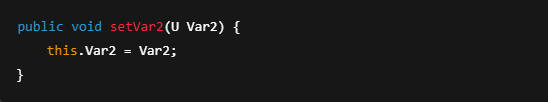
* 1. Getter Var2



* 1. Setter Va1



* 1. Setter Var2



1. Son secuencias de elementos almacenados en una lista encadenada

a. Pila

b. Cola

c. Lista

d. Variable

2. Comparadas con los vectores, estas permiten una mayor rapidez de inserción y borrado, pero una menor velocidad de acceso aleatorio Pilas Listas Colas.

a. Pila

b. Cola

c. Lista

d. Matriz

3. Es una estructura FIFO, first in, firstout.

a. Pilas

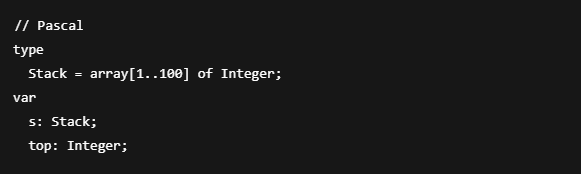
b. Colas

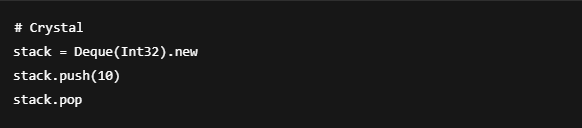
c. Listas

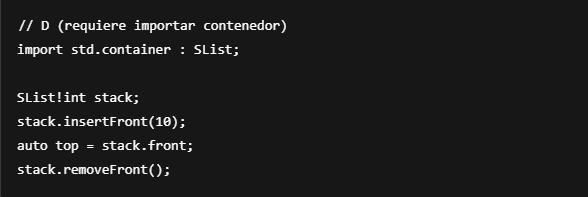
d. Árboles

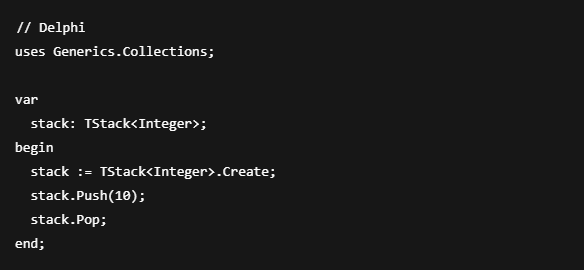
1. **Describa 3 aplicaciones de las colas en su vida como ingeniero.**

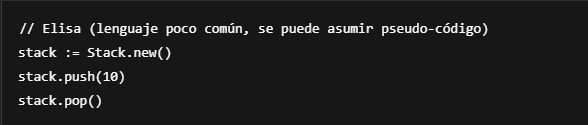
Describir cómo se declara una pila en los siguientes lenguajes de programación: Pascal, Crystal, D, Delphi, Elisa, Forth, Julia, Lingo, Mercury, Prolog, Ruby, UnixPipes, R.

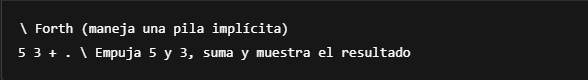




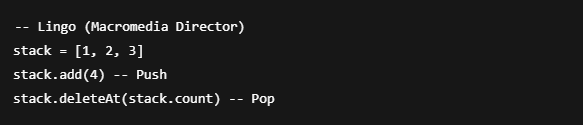
****

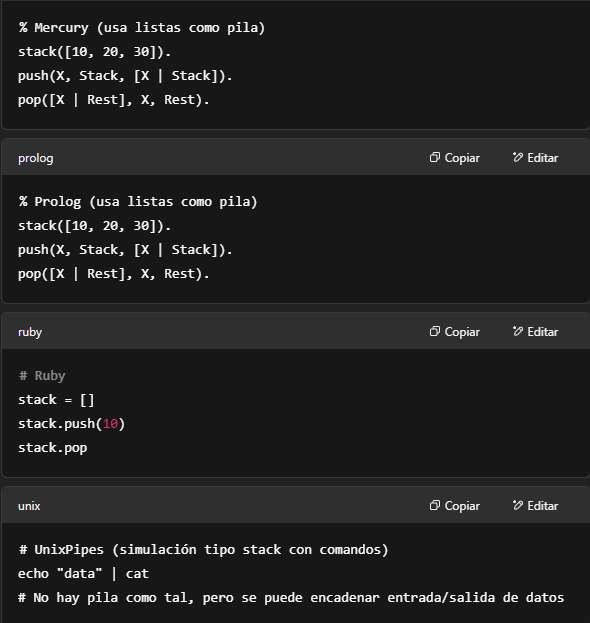
****

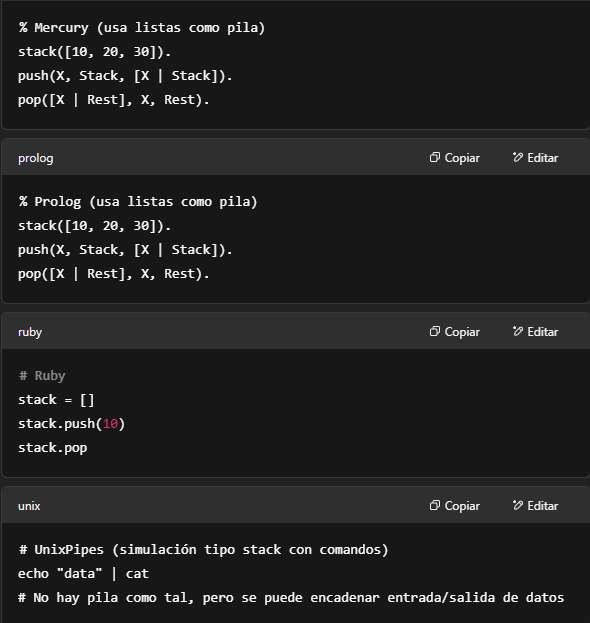
****

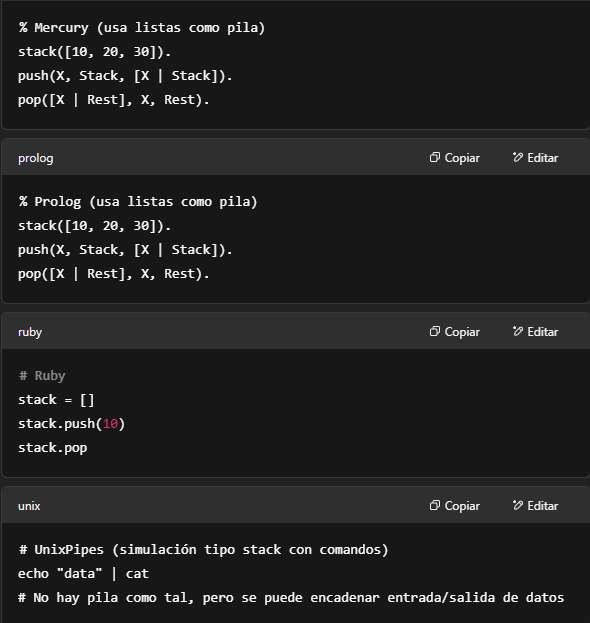
****

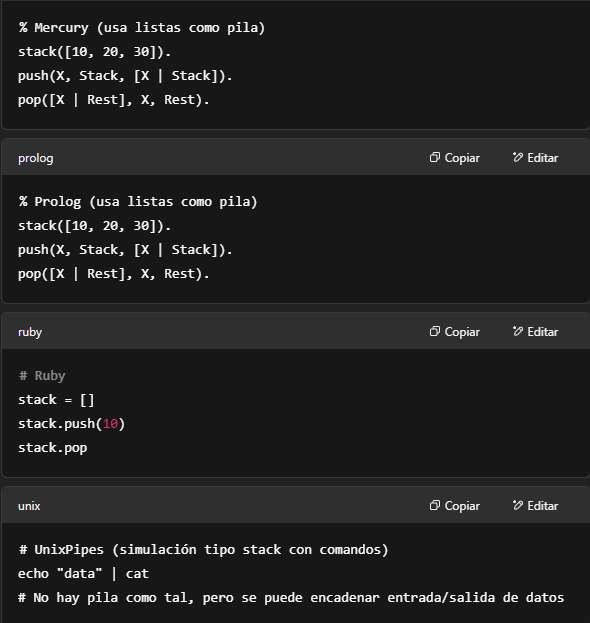


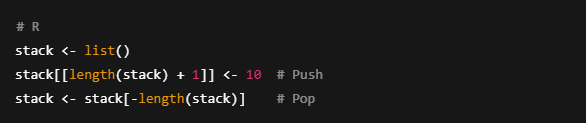












1. **Bibliografía:**

* Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3ra ed.). MIT Press.
* Horstmann, C. (2019). Big Java: Early Objects (7th ed.). Wiley.
* Ruby Language Documentation: <https://www.ruby-lang.org>
* Julia Documentation: <https://docs.julialang.org/>
* Crystal Docs: <https://crystal-lang.org/>
* Prolog Manual: <https://www.swi-prolog.org/pldoc/>
* R Programming Documentation: <https://cran.r-project.org/manuals.html>
* Forth Primer: <https://www.forth.com/starting-forth/>
* D Programming Language: <https://dlang.org/>