**Universidad Nacional Autónoma de México**

Facultad de Ingeniería

**Laboratorios de computación**

**salas A y B**

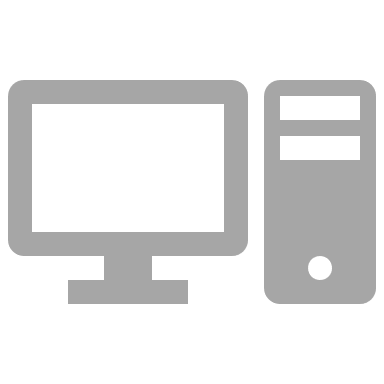
**PROFESOR:** M.I. Marco Antonio Martínez Quintana

**ASIGNATURA:** Estructura de Datos y Algoritmos I

**GRUPO:** 17

**NO DE PRÁCTICA:** 5

**NOMBRE:** Reyes Mendoza Miriam Guadalupe

**SEMESTRE:** 2020-2

**FECHA DE ENTREGA:** 07/03/2020

**OBSERVACIONES:**

**CALIFICACIÓN:**

**ESTRUCTURAS DE DATOS LINEALES: PILA Y COLA**

**OBJETIVO**

Revisarás las definiciones, características, procedimientos y ejemplos de las estructuras lineales Pila y Cola, con la finalidad de que comprendas sus estructuras y puedas implementarlas.

**INTRODUCCIÓN**

**ESTRUCTURAS DE DATOS**

Los conjuntos (colecciones de datos) son tan fundamentales para las ciencias de la computación como lo son para las matemáticas.

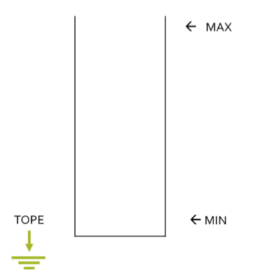
Una ***estructura de datos*** *consiste en una colección de nodos o registros del mismo tipo que mantienen relaciones entre sí. Un nodo es la unidad mínima de almacenamiento de información en una estructura de datos.*

**PILA**

La pila (**stack**) es una estructura de datos lineal y dinámica, en la cual el elemento obtenido a través de la operación ELIMINAR está predefinido.

Las operaciones que se pueden realizar sobre una pila son **INSERTAR** (que es llamada **PUSH**) y **ELIMINAR** (que es llamada **POP**). Debido a la política LIFO que implementa esta estructura, el orden en el que los elementos son extraídos de la pila (POP) es inverso al orden en el que los elementos fueron insertados en la pila (PUSH). Además, el único elemento accesible de la pila es el que está hasta arriba y que se conoce como tope de la pila. Para poder diseñar un algoritmo que defina el comportamiento de una pila se deben considerar 3 casos para ambas operaciones (push y pop):

* Estructura vacía (caso extremo).
* Estructura llena (caso extremo).
* Estructura con elemento(s) (caso base).

**PILA VACIA**

Una pila vacía no contiene elemento alguno dentro de la estructura y el tope de esta apunta a nulo.

En una pila vacía no es posible realizar POP, debido a que la estructura no contiene información.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteCuando la pila está vacía sí se puede realizar PUSH, en tal caso, el nodo que entra a la estructura sería el único elemento de la pila y el tope apuntaría a él.

Imagen que contiene captura de pantalla

Descripción generada automáticamente

**PILA LLENA**

Imagen que contiene captura de pantalla

Descripción generada automáticamentePor definición, una estructura de datos tipo pila tiene un tamaño fijo. Cuando la pila ha almacenado el número máximo de nodos definido, se dice que la pila está llena. En una pila llena no es posible hacer PUSH de un nuevo elemento, ya que se ha alcanzado el tamaño máximo permitido.

Cuando la pila está llena se puede hacer POP de la información contenida en la estructura. En tal caso, el tope apunta al elemento siguiente de la estructura.

Imagen que contiene captura de pantalla

Descripción generada automáticamente**PILA CON ELEMENTOS**

Una pila que contiene elementos (sin llegar a su máxima capacidad) representa el caso general.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteEn una pila con elementos se pueden realizar PUSH. En tal caso, el tope apuntara al elemento que se insertó y el nuevo elemento apunta al elemento al que apuntaba tope.

En una pila con elementos es posible realizar POP. En tal caso, el nodo al que apunta tope se extrae y ahora tope apunta al elemento al que apuntaba éste (sucesor).

**COLA**

La cola (**queue**) es una estructura de datos lineal, en la cual el elemento obtenido a través de la operación ELIMINAR está predefinido y es el que se encuentra al inicio de la estructura.

La cola es una estructura de datos de tamaño fijo y cuyas operaciones se realizan por ambos extremos; permite INSERTAR elementos al final de la estructura y permite ELIMINAR elementos por el inicio de esta. La operación de **INSERTAR** también se le llama **ENCOLAR** y la operación de **ELIMINAR** también se le llama **DESENCOLAR**.

Para poder diseñar un algoritmo que defina el comportamiento de una COLA se deben considerar 3 casos para ambas operaciones (INSERTAR y ELIMINAR):

* Estructura vacía (caso extremo).
* Imagen que contiene captura de pantalla

  Descripción generada automáticamenteEstructura llena (caso extremo).
* Estructura con elemento(s) (caso base).

Imagen que contiene captura de pantalla

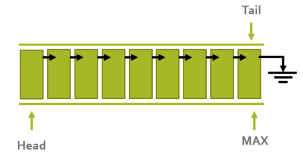
Descripción generada automáticamente**COLA VACIA**

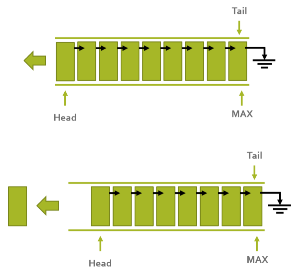
Imagen que contiene captura de pantalla

Descripción generada automáticamenteLa cola posee dos referencias, una al inicio (**HEAD**) y otra al final (**TAIL**) de la cola. En una cola vacía ambas referencias (HEAD y TAIL) apuntan a nulo.

En una cola vacía no es posible desencolar debido a que la estructura no posee elementos.

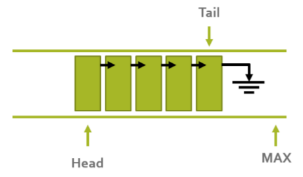
En una cola vacía sí se pueden encolar elementos, en este caso las referencias HEAD y TAIL apuntan al mismo elemento, que es el único en la estructura.

**COLA LLENA**

Cuando la referencia a tail de una cola llega a su máxima capacidad de almacenamiento (MAX) se dice que la cola está llena.

En una cola llena no es posible encolar más elementos.

En una cola llena sí se pueden desencolar elementos, en tal caso se obtiene el elemento al que hace referencia head y esta referencia se recorre al siguiente elemento (sucesor).



**COLA CON ELEMENTOS**

Imagen que contiene captura de pantalla

Descripción generada automáticamenteUna cola que contiene elementos (sin llegar a su máximo tamaño) representa el caso general de la estructura.

En una cola con elementos es posible desencolar nodos, recorriendo la referencia al inicio de la cola (HEAD) al siguiente elemento de la estructura.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteAsí mismo, se pueden encolar elementos en una cola mientras la referencia al final (TAIL) de la estructura no sea mayor al tamaño máximo de la misma. Cuando se encola un elemento, el nodo al que apunta TAIL tiene como sucesor el nuevo elemento y la referencia a TAIL apunta al nuevo elemento.

**DESARROLLO Y RESULTADOS**

**Imagen que contiene captura de pantalla

Descripción generada automáticamente**

**RESULTADOS**

**Imagen que contiene captura de pantalla

Descripción generada automáticamente**

**EXPLICACION**

1. Son estructuras de datos lineales utilizadas en el manejo de la información:

Las **pilas** y las **colas** son estructuras de datos lineales pues son aquellas en las que los elementos ocupan lugares sucesivos en la estructura y cada uno de ellos tiene un único sucesor y predecesor.

2. Una de sus aplicaciones es el manejo de vistas de una aplicación móvil.

Las **pilas** se usan en la aplicaciones móviles un ejemplo es WhatsApp puesto que cuando la aplicación inicia tu vez los mensajes en orden consecutivo empezando del mensaje más reciente situado hasta arriba hasta el mensaje más antiguo que se encuentra en la parte inferior, con esto podemos observar pues el primer elemento que entra (el mensaje más reciente) es el ultimo en salir, mientras que el que entra (el mensaje más reciente) queda como el tope de la pila y es el primero que se muestra en la pantalla principal de la aplicación.

3. Una de sus aplicaciones es la forma de trabajo de una impresora

La **cola** de impresión permite enviar documentos de gran tamaño, o varios documentos, a una impresora sin tener que esperar que se complete la impresión para seguir con la siguiente tarea. Cuando se envía un archivo a imprimir, se crea un archivo de almacenamiento intermedio, donde se almacena lo que se envía a la impresora. Las impresiones se van realizando según vayan llegando los archivos.

4. ¿Qué forma de acceso a la información tiene las pilas?

Las pilas implementan las políticas **LIFO** (*Last-In, First-Out*) y **FILO** (*First-In, Last-Out*) esto quiere decir que el ultimo elemento en entrar va a ser el primero en salir o el primero que en entrar va a ser el ultimo en salir, que es lo mismo.

5. ¿Qué forma de acceso a la información tiene las colas?

Las colas funcionan de manera inversa a las pilas pues utilizan las políticas **FIFO** (*First-In, First-Out*) y **LILO** (*Last-In, Last-Out*) aquí lo que sucede es que el primero en entrar va a ser el primero en salir o en otros caso el ultimo en entrar va a ser el ultimo el salir.

**CONCLUSIONES**

En los temas anteriores hemos estudiado distintos ejemplos donde se almacena una serie de objetos en estructuras. Con este tema pude profundizar en las estructuras de datos lineales que implementan tipos de datos abstractos que son un conjunto de elementos organizados secuencialmente. Pudimos ver algunas formas de su implementación y ver que son realmente útiles pero que también muchas de las aplicaciones y algunos dispositivos electrónicos tienen como bases algunos tipos de pilas y de colas.

Este tipo de estructuras nos son muy útiles pues en caso de tener muchos elementos es mucho más eficiente implementamos una de manera adecuada, usando también memoria dinámica a copiar y pegar datos. Realmente todos los temas que hemos visto hasta ahora nos han enseñado a hacer más eficiente, más rápido y con mayor rendimiento un programa y a final de cuentas son bases solidas para que con el paso del tiempo sepamos optimizar código y memoria de acuerdo con lo que queramos hacer.

**BIBLIOGRAFÍA**

* Garrido C. Antonio, Fernández. V. Joaquín. (2006). *Abstracción y Estructuras de Datos en C++* (1ª Ed.). Madrid, España: Delta Publicaciones Universitarias.