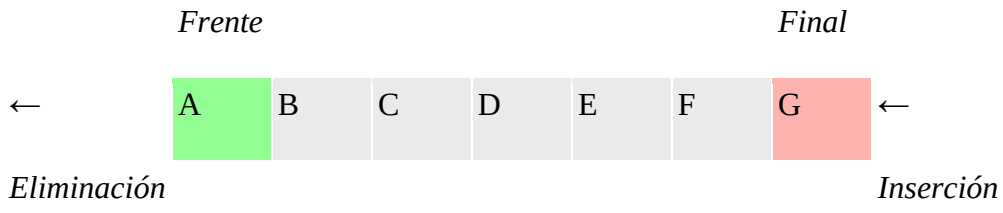
Asignatura: Teoría de Autómatas
Profesor: D. Sc. Gerardo García Gil
2021-B
Ingeniería en Desarrollo de Software
Centro de Enseñanza Técnica Industrial (CETI)

-Presentación

En términos muy generales, el área de estudio de estructuras de datos como disciplina de las ciencias computacionales (CS) es el almacenamiento y la manipulación de la información. Las estructuras de datos, siendo una disciplina dentro de la categoría de la computación teórica, estudian de manera más abstracta la relación entre una colección de datos y las operaciones que pueden ser aplicadas a dicha colección. A continuación, se hablará acerca de una de las estructuras de datos fundamentales en el manejo de colecciones: las colas (queues).

-Introducción

Una cola es un grupo ordenado de elementos del mismo tipo, en la cual dichos elementos se añaden por un extremo (*Final*) y se quitan por el otro extremo (*Frente*). Esto significa que los elementos se sacan en el mismo orden en el que fueron insertados o introducidos en la cola, siendo por ello considerada como una estructura de datos FIFO (First In First Out), es decir, que el primer elemento en entrar es el primer elemento en salir.



A modo de ejemplo, una estructura FIFO funciona exactamente igual que una cola en un establecimiento, cuando llegas lo primero que debes hacer es pedir la vez, en ese momento te sitúas detrás del último que llegó, esto es, al final de la cola. A la hora de atender, el dependiente atiende a quién ocupa la primera posición de la cola, y así sucesivamente hasta que te toque a ti.

Otros ejemplos del uso de colas pueden ser una cola para la impresión de documentos, o en un videojuego la animación de varios elementos en movimiento dentro de la pantalla, así se crea una cola para ir animando uno a uno los personajes del juego, de forma que cuando avanzamos alguno una posición se manda otra vez al final de la cola, lo que da la sensación de que se están moviendo todos simultáneamente.

-Desarrollo

Operaciones básicas sobre una cola

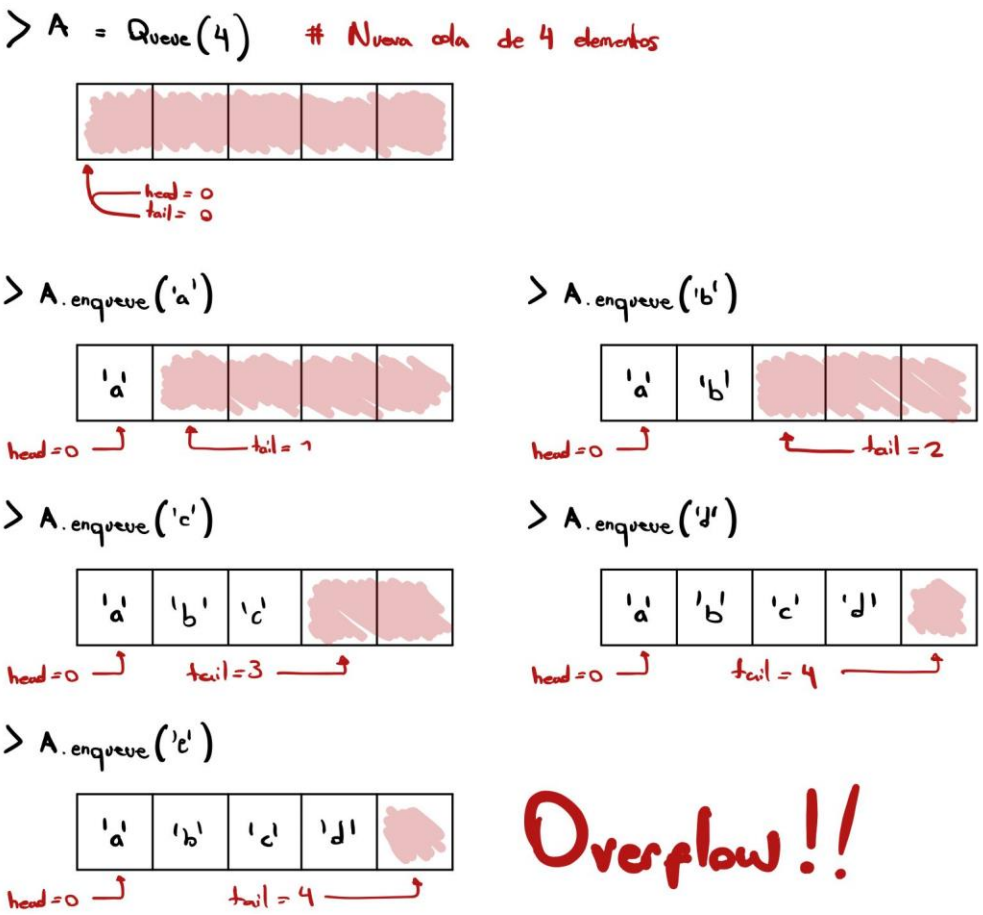
Para manejar una estructura de datos de tipo cola, un programador debe definir un conjunto de operaciones que permitan al usuario acceder y manipular los elementos almacenados en ella. La terminología más común utilizada para referenciar este conjunto de operaciones es la mostrada a continuación:

- **InicializarCola**: nos permite dejar inicialmente vacía la cola una vez creada.
- **Encolar**: permite añadir un elemento al final de la cola.
- **Desencolar**: se usará para sacar un elemento de la cola.
- **Cola Vacia**: devolverá cierto si la cola está vacía antes de sacar un elemento de la misma.
- **Cola Llena**: sólo en aquellos casos en los que sea necesario determinar si la cola se encuentra llena antes de añadir un nuevo elemento debido a la implementación utilizada (sólo en el caso de implementación con tablas), emplearemos esta operación de tipo lógico.

En las colas la operación que agrega coloca los nuevos elementos después del último. Esta operación es conocida como enqueue. Por otro lado, la operación que elimina al primero en la fila se conoce como dequeue, es decir, ya atendieron al que llegó primero. Y cuando se llega al límite y se quiere agregar un elemento más se dice que sucede un queue overflow, y de manera opuesta un queue underflow cuando se desean eliminar elementos de una cola vacía.

-Implementación

Una forma de implementar las colas es utilizando un apuntador head con la ubicación del primer elemento de la cola y otro apuntador tail con la posición siguiente del último elemento.

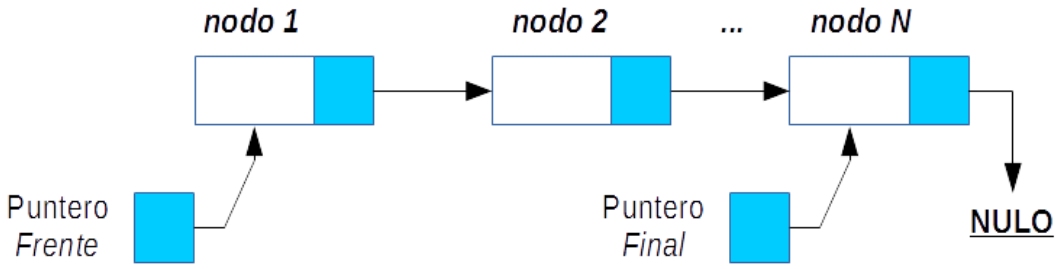


Una cola, al igual que una pila, no está incorporada en la mayoría de los lenguajes de programación y puede ser implementada de dos formas diferentes:

- Utilizando una tabla unidimensional (estructura de datos estática).
- Mediante el uso de punteros (estructura de datos dinámica).

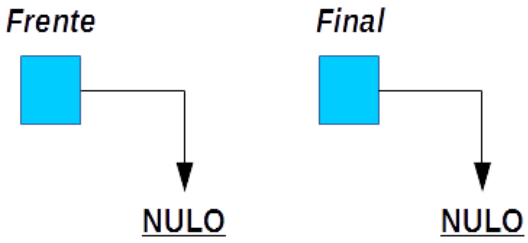
En el caso de la implementación con punteros, se representa por medio de una lista enlazada, y para controlar la inserción o eliminación de elementos utilizamos dos punteros externos denominados Frente (que apunta al primer elemento introducido en la cola) y Final (que apunta al último elemento introducido en la cola).

Implementación de una cola con punteros



Cuando la cola se crea, los punteros Frente y Final tienen valor Nulo, ya que inicialmente la cola debe encontrarse vacía.

Implementación de una cola con punteros (creación)



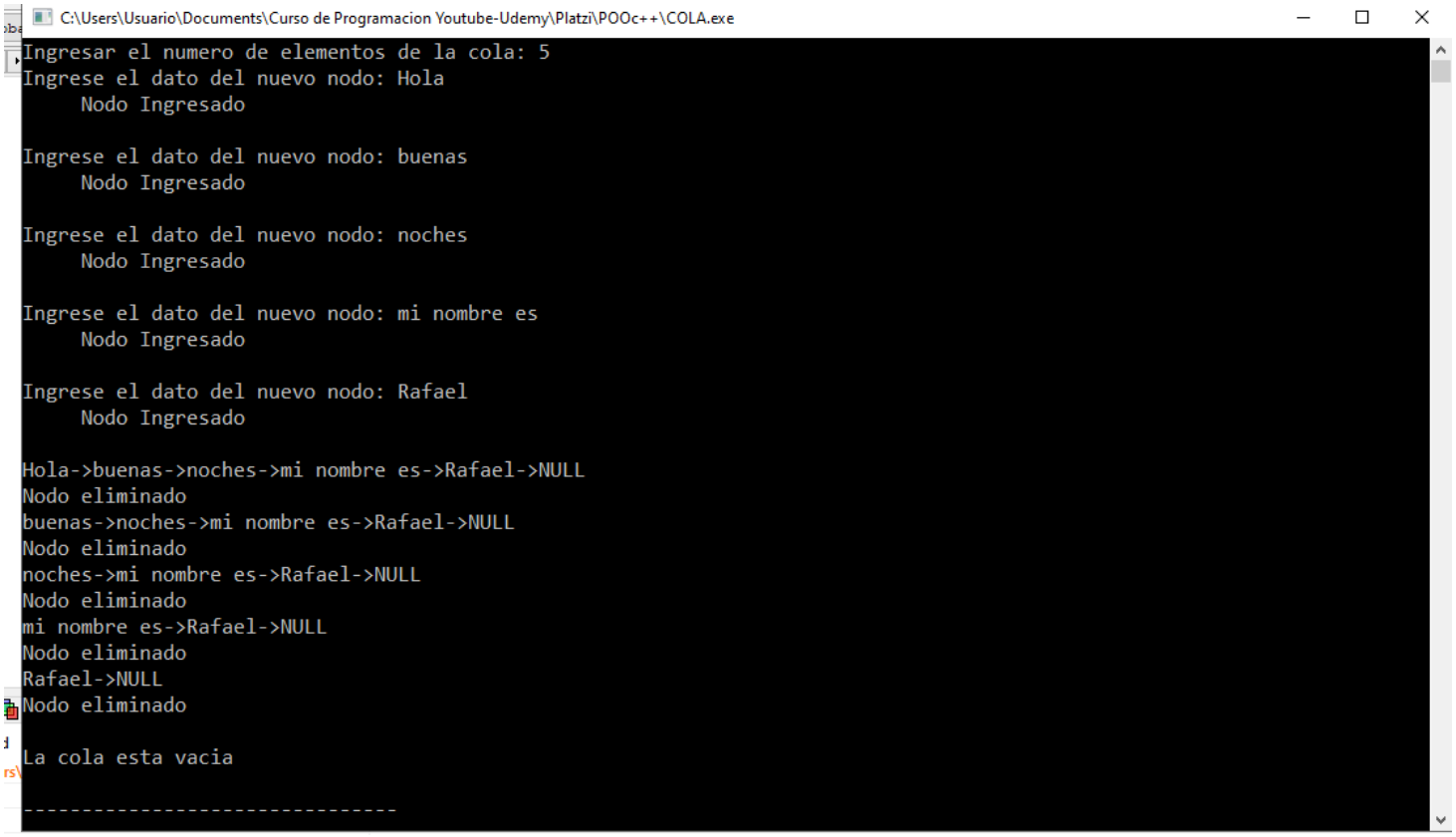
Conforme vayamos añadiendo elementos a la cola, Frente tomará el valor correspondiente a la dirección del primer elemento de la cola y Final tomará el valor correspondiente a la dirección del último elemento de la cola. De esta manera podemos determinar en todo momento el comienzo y final de la misma.

Ejemplo de cola con strings en C++

```
#include<iostream>
#include<string>
using namespace std;
struct Nodo{ //Definimos nuestra estructura nodo
	string dato;
	Nodo *siguiente;
}*primero, *ultimo; // Definimos el nodo puntero que va a apuntar al principio de la cola (primero) y
		// al final de la cola (ultimo)
void push();
void mostrarCola();
void eliminarNodo(Nodo *&,Nodo *&);
int tam;
int main(){
	cout<<"Ingresar el numero de elementos de la cola: ";
	cin>>tam;
	cin.ignore();
	for(int i=0;i<tam;i++){
		push();
	}
	mostrarCola();
	for(int i=0;i<tam;i++){
	eliminarNodo(primero,ultimo);
	mostrarCola();
}
	return 0;
}
void push(){
	Nodo* nuevo = new Nodo();
	cout<<"Ingrese el dato del nuevo nodo: ";
	getline(cin,nuevo->dato);
	if(primero==NULL){ //si el primer nodo es nulo
		primero = nuevo; // el primero nodo es igual al nuevo dato del nodo que se ingreso
```

```
                primero->siguiente = NULL; // entonces el nodo que sigue del que ingresamos es nulo
                ultimo  = primero; // el ultimo nodo ahora es igual al primer nodo debido a que solo existe un nodo
en la cola
        }else{ // si el primer nodo no es nulo
                ultimo->siguiente = nuevo; //entonces el ultimo nodo apunta al nuevo ingresado
                nuevo->siguiente = NULL; // y el nodo que sigue de nuevo nodo es nulo
                ultimo = nuevo;   // ahora nodo ultimo es igual al nuevo
        }
        cout<<"     Nodo Ingresado"<<endl<<endl;;
}
void mostrarCola(){
        Nodo* actual = new Nodo(); // se crea un nodo (actual) en la que se mostrara el dato de cada nodo de la cola
        actual = primero;    //se iguala el nodo actual al primer nodo de la cola
        if(primero != NULL){ // se verifica si el primer nodo es igual a NULL
        while(actual!=NULL){
        cout<<actual->dato<<"->"; //se imprime el dato del nodo
        actual = actual->siguiente;         //el dato de actual cambia al siguiente nodo de la cola
        }cout<<"NULL"<<endl;}
        else{
                cout<<"\nLa cola esta vacia"<<endl;
        }
}
void eliminarNodo(Nodo *&frente,Nodo *&fin){
        cout<<"Nodo eliminado"<<endl;
        Nodo* aux = new Nodo;
        aux  = frente;// el nodo auz se iguala a primer nodo
        if(frente==fin){ // si el primer nodo es igual al final
                frente = NULL;
                fin = NULL; //primero y ultimo se iguala a nulo
        }else{
                frente = frente->siguiente; // el primero se iguala al siguiente nodo que apunta
        }
        delete aux; // se elimina el nodo aux
}
```

-Resultados



Como podemos observar primero se ingresa el numero de elementos de la cola, después por medio del for se piden los datos. Enseguida se imprime los datos conforme fueron ingresados y entonces se llama a la funcion “eliminarNodo” para asi eliminar cada nodo y al mismo tiempo imprimir la cola despues de cada eliminacion hasta que dar vacia y dar el anuncio de que la cola está vacía.

-Conclusión

Este tema me parecio un poco complicado debido a que no lo podia entender muy bien, entendia el concepto de como funcionaba pero en el momento de implementarlo en código no lo podia comprender, fue por ello mismo que lei los pdf y explicación del profesor, aparte de que leí informacion en internet y observé una cantidad considerable de videos hasta que pude comprender este tema, me di cuenta que para poder entender estos temas de TDA hay que tener muy claro el tema de punteros para poder comprender estos temas. Ya al final cuando comprendi como funcionan las colas me resulto mas facil realizar el código del programa a traves del uso de strings.

-Referencias

Barrera, F. J., & Bejarano Gómez, L. D. (s. f.). *3.4. Colas | Programación avanzada: Estructuras de datos y funciones*. IEDA. Recuperado 30 de septiembre de 2021, de http://agrega.juntadeandalucia.es/repositorio/02122016/a5/es-an_2016120212_9131705/34_colas.html

Trinidad, P. (2019, 18 abril). *Estructuras de Datos[3] - Arreglos, pilas y colas*. Pablo Trinidad. https://pablotrinidad.me/edd-3-arreglos-pilas-y-colas/