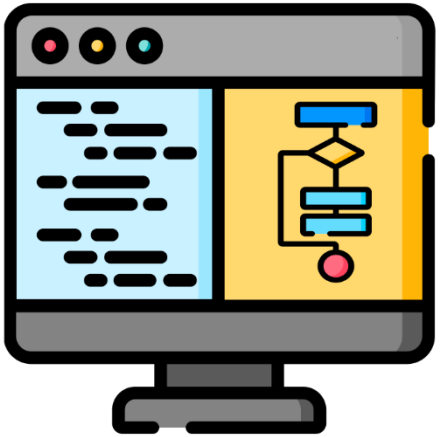


2022

ALGORITMOS Y ESTRUCTURAS DE DATOS II

Serie Práctica Nro. 1:
Listas, Pilas y Colas implementadas con Arreglos.

Docentes:
Aquino – Burghardt – Princich

Estructuras de Datos Dinámicas

Lineales	No Lineales
Pilas Colas Listas	Árboles Grafos

Las estructuras de datos dinámicas son una colección de elementos, que normalmente son registros, con la particularidad que crecen a medida que se ejecuta un programa.

La estructura de datos dinámica se amplia y se contrae a medida que se ejecuta el programa.

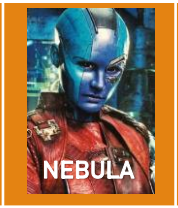
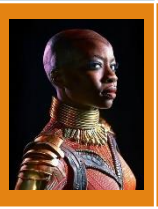
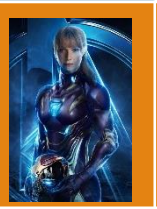
Se pueden dividir en dos grandes grupos:

Estructuras de Datos Dinámicas

A cada elemento de la estructura lo denominamos **NODO**.

Puede modificar su estructura mediante el programa. Puede **modificar su tamaño** añadiendo o eliminando Nodos mientras esta en ejecución el programa.

Personajes MCU - Avengers

						
1	2	3	4	5	6	7
						
8	9	10	11 HAWKEYES	12	13	14
						
15	16	17	18	19	20	21
						
22	23	24	25			

personaje personajes[25];

Listas

Una lista es un conjunto ordenado de elementos de un tipo.

Representación:

Lista Lineal

Operaciones:

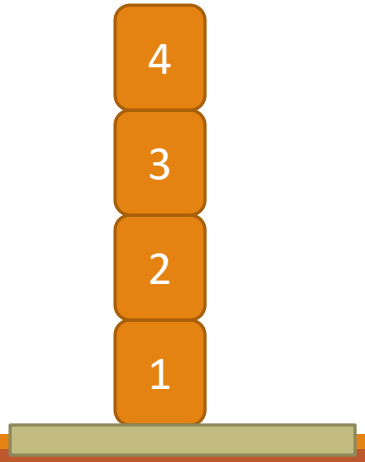
Recorrido de la lista.
Inserción de un elemento.
Eliminación de un elemento.
Búsqueda de un elemento.

Pilas

Pilas

Un **PILA** es una **LISTA** de elementos a la cual se puede insertar o eliminar alguno de ellos solo por uno de los extremos.

Los elementos de la pila serán eliminados en orden inverso al que se insertaron. **LIFO**: ultimo en entrar, primero en salir.



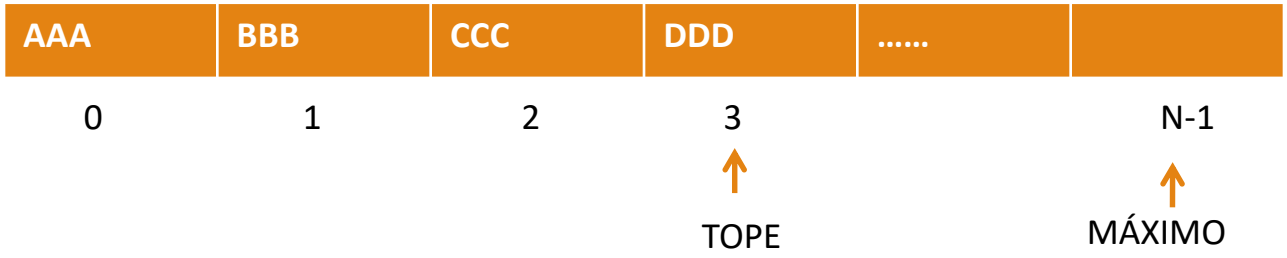
Representación de Pilas

Se pueden representar mediante el uso de arreglos o listas enlazadas.

```
4  #define MAX 5
5  typedef int tArrayEnteros [MAX];
6  typedef struct {
7      tArrayEnteros listaEnteros;
8      int tope;
9  }tPila;
10
11 tPila pila;
```

Tope se inicializa en -1, para no preocuparnos por el índice del arreglo que comienza en 0.

Al utilizar arreglos debemos definir el tamaño máximo de la pila y la variable auxiliar tope o cima.
Cuando el Tope = Máximo → Pila Llena



Operaciones con **Pila**: Poner un elemento (**PUSH**)

PONER

Si TOPE < MAX *(verifica que haya espacio libre)*

entonces

TOPE = TOPE + 1 *(actualiza TOPE)*

PILA (TOPE) = DATO

sino

escribir desbordamiento

Fin_si

Operaciones con **Pila:** Quitar un elemento (**Pop**)

QUITA

Si TOPE > 0 *(verifica que la pila no este vacía)*

entonces

DATO = PILA(TOPE)

TOPE = TOPE – 1; *(actualiza TOPE)*

sino

escribir *"Subdesbordamiento"*

Fin_si

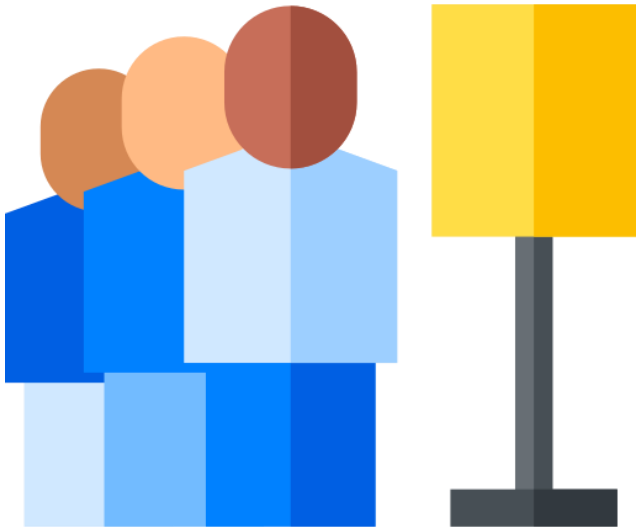
Ejemplo:

La Biblioteca de la Facultad recibe libros diariamente, que corresponden a devoluciones o nuevas adquisiciones. Los libros se apilan, máximo de 10 libros, según su orden de ingreso y se eliminan de la pila a medida que se acomodan en los estantes. Por cada libro se registra el ISBN para su identificación.

```
void crearPilaVacia();
bool pilaVacia(tPila);
bool pilaLlena(tPila);
void apilar(int);
void desapilar();
int cima(tPila);
void visualizarElementos(tPila);
```

Colas

Colas



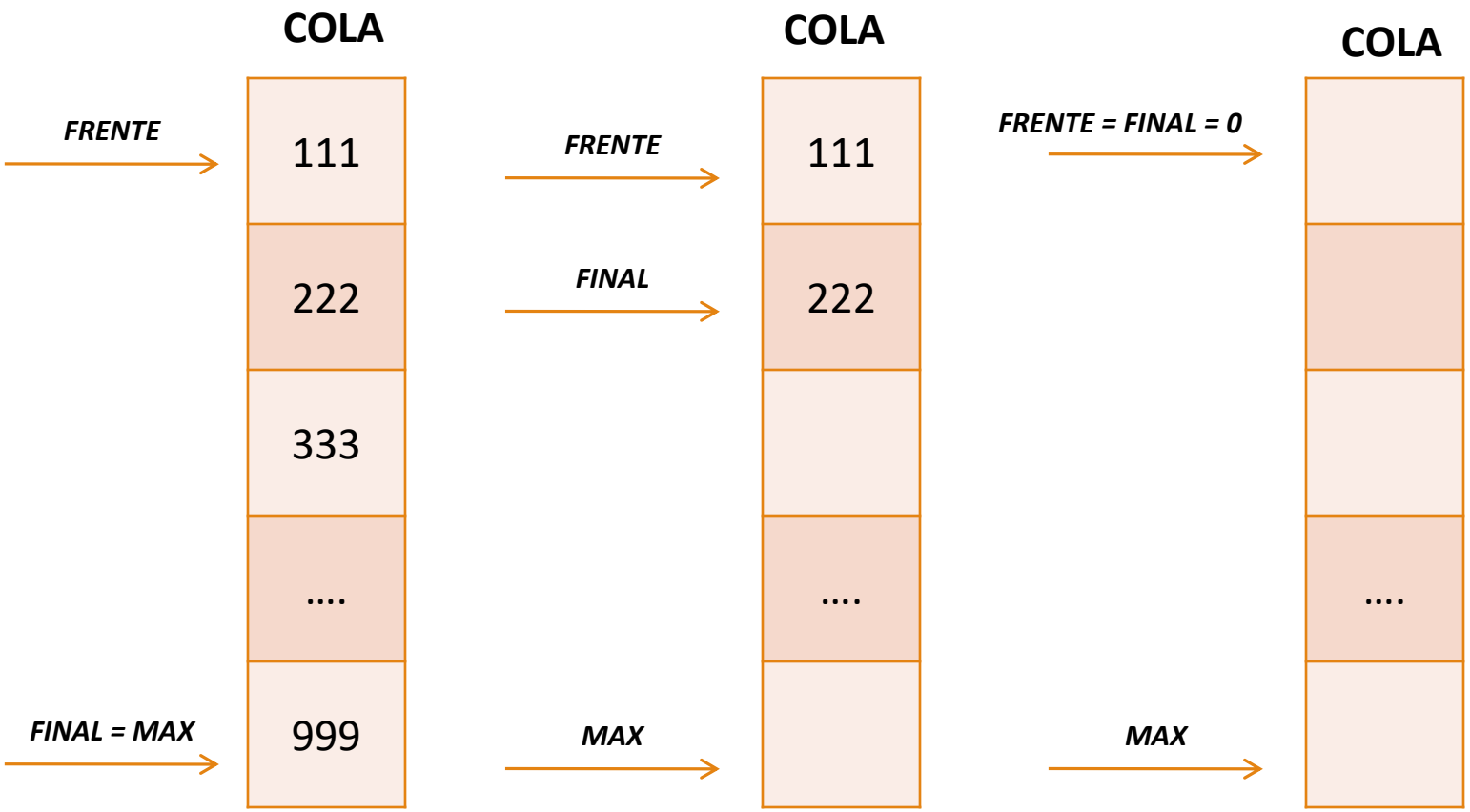
Las **COLAS** son **LISTAS** de elementos en la que estos se introducen por un extremo (final) y se eliminan por otro (frente). **FIFO**

Por esta característica este tipo de estructura se utiliza para almacenar datos que necesitan ser procesados según el orden de llegada.

Ejemplos:

- Cola de supermercado.
- Cola de impresión.

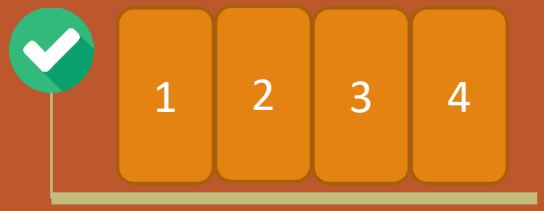
Colas – Representación



Operaciones

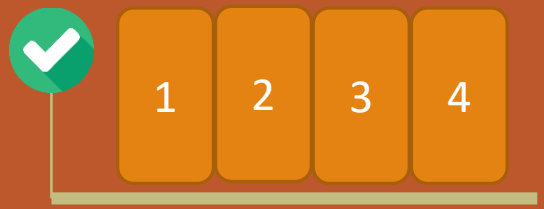
- Crear la cola.
- Saber si la cola esta vacía.
- Saber si la cola esta llena.
- Agregar un elemento.
- Eliminar un elemento.
- Recorrer:
  - Mostrar los elementos.
  - Buscar un elemento.
  - Etc.

Colas – Operaciones: Agregar Elemento



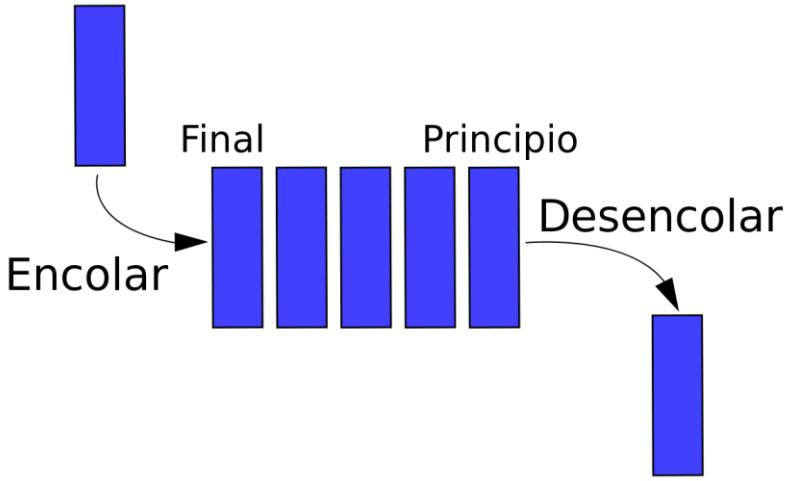
PONER
Si FINAL < MAX-1 *(verifica que haya espacio libre)*
entonces
FINAL = FINAL + 1 *(actualiza FINAL)*
COLA(FINAL) = DATO
Si FINAL = 0
entonces
FRENTE = 0 *(actualiza FRENTE)*
fin_si
sino
escribir *"Desbordamiento"*
Fin_si

Colas – Operaciones: Quitar



QUITAR
Si FRENTE<> -1 *(verifica que no esté vacía)*
entonces
DATO = COLA(FRENTE)
Si FRENTE = FINAL *(si hay un solo elemento)*
entonces
FRENTE = -1
FINAL = -1
sino
FRENTE = FRENTE + 1
fin_si
sino
escribir *"Subdesbordamiento"*
Fin_si

```
4  #define MAX 5
5  typedef int tArrayEnteros [MAX];
6  typedef struct {
7      tArrayEnteros listaEnteros;
8      int frente, final;
9  }tCola;
10
11 tCola cola;
```



Colas

DEFINICIÓN DE LA ESTRUCTURA

Bibliografía

Material de teoría de la catedra Algoritmos y Estructuras de Datos II.

Pablo A. Sznajdleder. Algoritmos a fondo, con implementaciones en C y Java. Editorial Alfaomega. 2012.

Luis Joyanes Aguilar. Fundamentos de programación. Algoritmos , estructuras de datos y objetos. Mc Graw Hill. 4ta edición.