

# **UNIVERSIDAD NACIONAL DEL ALTIPLANO DE** **PUNO**

*Facultad de ingeniería mecánica eléctrica, electrónica y  
sistemas*

Escuela profesional de ingeniería de sistemas



## **Actividad 01 – Complemento**

### **PRESENTADO POR:**

CCALLO ARPITA JHAK ESNAYDER

### **CURSO:**

ALGORITMOS Y ESTRUCTURAS DE DATOS

### **DOCENTE:**

ZANABRIA GALVEZ ALDO HERNAN

**SEMESTRE IV**

**PUNO – PERÚ**

**13 de Abril - 2025**

## 1- Tabla de trazado manual:

### Pilas:

TABLA DE TRAZADO MANUAL (PILA)

Linea	datos	tope	Max	Valor	i
1	[ ] [ ] [ ] [ ]	-1	5	-	-
2	↓ [ 10 ] [ ] [ ] [ ]	0	5	10	-
3	↓ [ 10 20 ] [ ] [ ] [ ]	1	5	20	-
4	↓ [ 10 20 30 ] [ ] [ ] [ ]	2	5	30	-
5	↑ [ 10 20 30 ] [ ] [ ] [ ]	2	5	-	2
6	↑ [ 10 20 30 ] [ ] [ ] [ ]	2	5	-	1
7	↑ [ 10 20 30 ] [ ] [ ] [ ]	2	5	-	0
8	↑ [ 10 20 30 ] [ ] [ ] [ ]	1	5	-	-
9	↑ [ 10 20 30 ] [ ] [ ] [ ]	1	5	-	1
10	↑ [ 10 20 30 ] [ ] [ ] [ ]	1	5	-	0

Consola:

```
> 30 20 10
> 20 10
> cima: 20
```

### Colas:

TABLA DE TRAZADO MANUAL (COLA)

Linea	datos	max	fuente	fin	valor	i	
1	[ ] [ ] [ ] [ ] [ ] [ ]	100	-1	-1	-	-	← inicializamos
2	[ 5 ] [ ] [ ] [ ] [ ] [ ]	100	0	0	5	-	← insertamos 5
3	[ 5 15 ] [ ] [ ] [ ] [ ] [ ]	100	0	1	15	-	← insertamos 15
4	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	0	2	25	-	← insertamos 25
5	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	0	2	-	0	← mostramos 5
6	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	0	2	-	1	← mostramos 15
7	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	0	2	-	2	← mostramos 25
8	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	0	2	-	3	← no mostramos nada ya q' i > fin
9	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	1	2	-	-	← desincoloramos 5
10	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	1	2	-	1	← mostramos 15
11	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	1	2	-	2	← mostramos 25
12	[ 5 15 25 ] [ ] [ ] [ ] [ ] [ ]	100	1	2	-	3	← no mostramos nada

Consola:

```
> 5 15 25
> 15 25
> Fuente: 15
```

## Listas:

	cabeza	valor	nuevo	siguiente	actual	dato	temp
①	Nullptr	-	-	-	-	-	-
②	0x1	10	0x1	Nullptr	-	10	-
③	0x2	20	0x2	0x1	-	20	-
④	0x2	30	0x3	Nullptr	0x2	30	-
⑤	0x2	30	0x3	Nullptr	0x1	-	-
⑥	0x2	30	0x3	0x3	0x1	-	-
⑦	0x2	-	-	0x1	0x2	20	-
⑧	0x2	-	-	0x3	0x1	10	-
⑨	0x2	-	-	Nullptr	0x3	30	-
⑩	0x2	-	-	-	Nullptr	-	-
⑪	0x2	-	-	0x1	-	-	0x2
⑫	0x2	-	-	-	-	-	Nullptr
⑬	0x2	-	-	0x3	0x2	10	-
⑭	0x1	-	-	Nullptr	0x3	30	-
⑮	0x2	-	-	-	Nullptr	-	-

consolas:

20 → 10 → 30 → NULL  
 10 → 30 → NULL

## 2- Capturas del código en ejecución en OnlineGDB

### Pilas:

```
main.cpp
1  #include <iostream>
2  using namespace std;
3  #define MAX 10
4
5  class Pila {
6  private:
7      int datos[MAX];
8      int tope;
9
10 public:
11     Pila() {
12         tope = -1;
13     }
14
15     bool estaVacia() {
16         return tope == -1;
17     }
18
19     bool estaLlena() {
20         return tope == MAX - 1;
21     }
22
23     void apilar(int valor) {
24         if (estaLlena()) {
25             cout << "Pila llena\n";
26             return;
27         }
28         datos[++tope] = valor;
29
30 20 10
20 10
Cima: 20
```

### Colas:

```
main.cpp
1  #include <iostream>
2  #define MAX 100
3  using namespace std;
4
5  class Cola {
6  private:
7      int datos[MAX];
8      int frente, fin;
9
10 public:
11     Cola() {
12         frente = fin = -1;
13     }
14
15     bool estaVacia() {
16         return frente == -1;
17     }
18
19     bool estaLlena() {
20         return fin == MAX - 1;
21     }
22
23     void encolar(int valor) {
24         if (estaLlena()) {
25             cout << "Cola llena\n";
26             return;
27         }
28         if (estaVacia())
```

```
5 15 25
15 25
Frente: 15
```

## Listas:

```
1  #include <iostream>
2  using namespace std;
3
4  struct Nodo {
5      int dato;
6      Nodo* siguiente;
7  };
8
9
10 class Lista
11 {
12 private:
13     Nodo* cabeza;
14
15 public:
16     Lista()
17     {
18         cabeza = NULL;
19     }
20
21     void insertarInicio(int valor) {
22         Nodo* nuevo = new Nodo();
23         nuevo->dato = valor;
24         nuevo->siguiente = cabeza;
25         cabeza = nuevo;
26     }
27 }
```

20 -> 10 -> 30 -> NULL  
10 -> 30 -> NULL  
10 -> 15 -> 30 -> NULL

## 3- Comparación entre salida real y teórica

### Pilas:

```
30 20 10
20 10
cima: 20
```

Consola:

```
> 30 20 10
> 20 10
> cima: 20
```

### Colas:

```
5 15 25
15 25
Frente: 15
```

Consola:

```
> 5 15 25
> 15 25
> Frente: 15
```

### Listas:

```
20 -> 10 -> 30 -> NULL
10 -> 30 -> NULL
```

Consola:

```
20 -> 10 -> 30 -> NULL
10 -> 30 -> NULL
```

La salida es la misma para todas las estructuras de datos

#### 4- Lista de errores o comportamientos inesperados

El único error que se encontró fue al compilar el código de la estructura de la pila ya que en este no se definía el valor de MAX que representa el tamaño del array.

#### 5- Propuesta de mejora en el código o el control de errores

Añadir una verificación en la función mostrar que diga explícitamente que no hay elementos en la estructura, ya sea de la pila o la cola.

Implementar una **cola circular**, donde frente y fin se muevan en forma circular esto con el fin de no desperdiciar espacio una vez que la cola se llene y se hayan desencolado algunos elementos.

Usar nodos en lugar de arrays para las pilas y colas, esto evitara el desperdicio de memoria cuando no se utiliza todos los espacios de los arrays.

#### 6- Conclusiones personales (mínimo 3)

- Implementar una pila o una cola estáticas es sencillo y comprensible, pero estas estructuras son limitadas y poco eficientes cuando se quiere trabajar con un numero de datos variables.
- El código proporcionado además de funcionar correctamente, es robusto ya que proporciona mensajes cuando cuando algo sale “mal” como cuando tratamos de desapilar una pila que ya esta vacía o cuando apilamos un numero de elementos que supera el máximo posible.
- Al analizar el código y seguir el flujo de datos en las tres estructuras (lista enlazada, pila y cola), se puede comprender con mayor claridad la diferencia entre las estructuras estáticas y dinámicas. Esto no solo facilita entender cómo funcionan internamente, sino que también permite identificar los beneficios y limitaciones de cada tipo de implementación.

Link a los códigos en online GDB:

Pila: <https://onlinegdb.com/TMKb59ZwY>

Cola: <https://onlinegdb.com/qLnHDxMPP>

Lista: [https://onlinegdb.com/X5\\_8z4vgt](https://onlinegdb.com/X5_8z4vgt)

Función de lista que permite insertar un nuevo nodo en cualquier nodo de la lista.

```
void insertarEn(int valor, int vNodo){

    Nodo* nuevo= new Nodo();
    nuevo->dato =valor;
    nuevo->siguiente=NULL;

    if(cabeza==NULL){
        cout<<"La lista esta vacia, no pudes insertar el nodo despues de "<<vNodo<<endl;
        delete nuevo;
    }

    else{
        Nodo* temp = cabeza;//creamos un temporal para poder recorrer la lista

        while(temp!=NULL ){//revisamos hasta el ultimo nodo

            if(temp->dato==vNodo ){
                Nodo* temp2 = temp->siguiente;//creamos otro temporal para no perder al resto de la lista
                temp->siguiente=nuevo;
                nuevo->siguiente=temp2;
                return;//una vez insertado salimos de la funcion
            }

            temp=temp->siguiente;
        }
        cout<<"el nodo "<<vNodo<<" no existe no puede insertar un nuevo nodo"<<" "<<endl;
    }
}
```