

Practica 03 Pilas

- Ejercicio 1. Simulador de Operaciones en una Pila.

-Codigo-

```
public class Ejercicio01 {  
    static class PilaSimple {  
        private int[] datos;  
        private int tope;  
  
        public PilaSimple (int capacidad) {  
            datos = new int [capacidad];  
            tope = -1;  
        }  
  
        public void push (int valor) {  
            if (tope < datos.length - 1) {  
                tope ++;  
                datos [tope] = valor;  
                System.out.println ("PUSH(" + valor + ")");  
            }  
        }  
  
        public int pop () {  
            if (tope >= 0) {  
                int valor = datos [tope];  
                tope --;  
                System.out.println ("POP() Se extrajo " + valor);  
                mostrar ();  
                return valor;  
            }  
            return -1;  
        }  
    }  
}
```

```

public void mostrar () {
    System.out.println("Pila : [");
    for (int i = 0; i <= tope; i++) {
        System.out.print(datos[i]);
        if (i < tope) System.out.print(", ");
    }
    System.out.println("\n");
}

```

```

public void mostrarFinal () {
    System.out.println("-- Contenido Final --");
    System.out.println("De arriba a abajo:");
    for (int i = tope; i >= 0; i--) {
        System.out.println("\nNumero de elementos: " + (tope + 1));
    }
}

```

```

public static void main (String[] args) {
    System.out.println("--- Ejercicio 01 ---\n");

    PilaSimple pila = new PilaSimple (10);

```

```

    pila.push (10);
    pila.push (20);
    pila.push (5);
    pila.pop ();
    pila.push (8);
    pila.pop ();
    pila.push (12);
    pila.mostrarFinal ();
}

```

Practica 03 Pilas

Ejercicio 4 - Pila aplicada: Deshacer (UNDO).

```
public class Ejercicio4 {
    static class PilaAcciones {
        private String [] datos;
        private int tope;

        public PilaAcciones (int capacidad) {
            datos = new String [capacidad];
            tope = -1;
        }

        public void push (String accion) {
            if (tope < datos.length - 1) {
                tope ++;
                datos [tope] = accion;
            }
        }

        public String pop () {
            if (tope >= 0) {
                String accion = datos [tope];
                tope --;
                return accion;
            }
            return "";
        }

        public boolean estaVacia () {
            return tope == -1;
        }
    }
}
```

(Estado Inicial: Pila vacía $[]$)

03

1. PUSH (10)

Pila: $[10]$

2. PUSH (20)

Pila: $[10, 20]$

3. PUSH (5)

Pila: $[10, 20, 5]$

4. POP ()

Elemento Eliminado: 5

Pila: $[10, 20]$

5. PUSH (8)

Pila: $[10, 20, 8]$

6. POP ()

Elemento Eliminado: 8

Pila: $[10, 20]$

7. PUSH (12)

Pila: $[10, 20, 12]$

- Contenido Final de arriba hacia abajo: 12, 20, 10
- Número de Elementos: 3.

Practica 03 Pilas

Ejercicio 2. Verificar si una palabra es palindroma usando una Pila. "RADAR"

```
public class Ejercicio 2 {  
    static class PilaChar {  
        private [] datos;  
        private int tope;  
  
        public PilaChar (int capacidad) {  
            datos = new char [capacidad];  
            tope = -1;  
        }  
  
        public void push (char letra) {  
            if (tope < datos.length - 1) {  
                tope++;  
                datos [tope] = letra;  
            }  
        }  
  
        public char pop () {  
            if (tope >= 0) {  
                char letra = datos [tope];  
                tope--;  
                return letra;  
            }  
            return '';  
        }  
  
        public boolean estaVacia () {  
            return tope == -1;  
        }  
    }  
}
```

```

public static void main (String[] args) {
    System.out.println("-- Ejercicio 2 --\n");
    String palabra = "RADAR";
    System.out.println("Palabra a verificar: " + palabra);
    Pila<Char> pila = new Pila<Char> (palabra.length());

    System.out.println("\n -- Paso 1: Inserta letras en pila --");
    for (int i = 0; i < palabra.length(); i++) {
        char letra = palabra.charAt(i);
        pila.push (letra);
        System.out.println("PUSH('"+ letra + "')");
    }

    System.out.println("\n --- Paso 2: Extraer y comparar ---");
    boolean esPalindromo = true;
    for (int i = 0; i < palabra.length(); i++) {
        char original = palabra.charAt(i);
        char extraido = pila.pop();
        System.out.print ("Posicion " + i + ": '" + original + "' = '"
            + extraido + "'");
        if (original == extraido) {
            System.out.println ("✓");
        } else {
            System.out.println ("✗");
            esPalindromo = false;
        }
    }
}

```

```

System.out.println("\n -- Resultado -- ");
if (esPalindromo) {
    System.out.println(palabra + "Si es Palindromo");
} else {
    System.out.println(palabra + "No es Palindromo");
}
}
}

```

PALABRA: RADAR

Paso 1: Insertar letras en pila

PUSH('R') - Pila: [R]
 PUSH('A') - Pila: [R A]
 PUSH('D') - Pila: [R A D]
 PUSH('A') - Pila: [R A D A]
 PUSH('R') - Pila: [R A D A R]

Paso 2: Extraer y Comparar

Posición 0: 'R' == 'R' ✓
 Posición 1: 'A' == 'A' ✓
 Posición 2: 'D' == 'D' ✓
 Posición 3: 'A' == 'A' ✓
 Posición 4: 'R' == 'R' ✓

RADAR es un Palindromo.

Todas las letras coinciden en orden inverso.

Higuel Lozano Arvizu GTID141

Practica 03 Pilas

Ejercicio 3.- Convertir una expresion de infijo a postfijo

```
public class Ejercicio3 {  
    static class PilaOperadores {  
        private char [] datos;  
        private int tope;  
  
        public PilaOperadores (int capacidad) {  
            datos = new char [capacidad];  
            tope = -1;  
        }  
  
        public void push (char operador) {  
            if (tope < datos.length - 1) {  
                tope ++;  
                datos [tope] = operador;  
            }  
        }  
  
        public char pop () {  
            if (tope >= 0) {  
                char op = datos [tope];  
                tope --;  
                return op;  
            }  
            return '';  
        }  
    }  
}
```

```

public char peek () {
    if (tope >= 0) {
        return datos [tope];
    }
    return ' ';
}

```

```

public boolean estaVacia () {
    return tope == -1;
}

```

```

public void mostrar () {
    System.out.print("[");
    for (int i = 0; i <= tope; i++) {
        System.out.print(datos [i]);
        if (i < tope) System.out.print(", ");
    }
    System.out.print("]");
}

```

```

static int precedencia (char op) {
    if (op == '*' || op == '-') return 1;
    if (op == '/' || op == '/') return 2;
    return 0;
}

```

```

public static void main (String [] args) {
    System.out.println("-- Ejercicio 3 -- \n");
    String expresion = "A+B * C-D";
    System.out.println("Expresion ingresada: " + expresion);
    System.out.println("\n -- Proceso de Conversion -- \n");
}

```

```
PilaOperadores pila = new PilaOperadores(10);  
String posfija = "";
```

```
System.out.println("Simbolo | Pila | Salida");  
System.out.println("-----|-----|-----");
```

```
for (int i = 0; i < expression.length(); i++) {  
    char c = expression.charAt(i);  
    System.out.print(" " + c + " ");
```

```
    if (Character.isLetter(c)) {  
        posfija += c + " ";  
        pila.mostrar();  
        System.out.println(" | " + posfija);
```

```
    }  
    else if (c == '+' || c == '-' || c == '*' || c == '/') {  
        while (!pila.estaVacia() & precedencia(pila.peek())  
            >= precedencia(c)) {  
            posfija += pila.pop() + " ";
```

```
        }  
        pila.push(c);  
        pila.mostrar();  
        System.out.println(" | " + posfija);
```

```
    }  
    while (!pila.estaVacia()) {  
        posfija += pila.pop() + " ";  
    }
```

```

System.out.print(" FIN    1");
pila.mostrar();
System.out.println("    1" + posfija);
System.out.println("\n-- Resultado Final--");
System.out.println("Notación posfija: " +
    posfija.trim());
}
}

```

Símbolo	Pila de operadores	Salida posfija
A	[]	A
+	[+]	A
B	[+]	A B
*	[+, *]	A B
C	[+, *]	A B C
-	[-]	A B C * +
	(se saca * y +)	
D	[-]	A B C * + D
FIN	[]	A B C * + D -
	(se saca -)	

Resultado Final: A B C * + D -

```

public void mostrar () {
    System.out.println("Pila: [");
    for (int i = 0; i <= tope; i++) {
        System.out.print(datos[i]);
        if (i < tope) System.out.print(", ");
    }
    System.out.println("]");
}

```

```

public String getTextoActual () {
    String texto = "";
    for (int i = 0; i <= tope; i++) {
        texto += datos[i];
    }
    return texto;
}

```

```

public static void main (String[] args) {
    System.out.println("-- Ejercicio 4 -- \n");
    PilaAcciones pila = new PilaAcciones (10);
    System.out.println("Accion | Texto | Pila");
    System.out.println("-----|-----|-----");

    pila.push("A");
    System.out.printf("%-20s | %-8s | ", "Escribir 'A'", "\n");
    pila.getTextoActual() + "\n";
    pila.mostrar();
}

```

pila.push("B");
System.out.printf("%-20s|%-8s|", "Escribir 'B'", "\"\" +
pila.getTextoActual() + "\"\"");
pila.mostrar();

pila.push("C");
System.out.printf("%-20s|%-8s|", "Escribir 'C'", "\"\" +
pila.getTextoActual() + "\"\"");
pila.mostrar();

String eliminado = pila.pop();
System.out.printf("%-20s|%-8s|", "UNDO (elimina '" + eliminado + "'
", "\"\" + pila.getTextoActual() + "\"\"");

pila.push("D");
System.out.printf("%-20s|%-8s|", "Escribir 'D'", "\"\" +
pila.getTextoActual() + "\"\"");

eliminado = pila.pop();
System.out.printf("%-20s|%-8s|", "UNDO (elimina '" +
eliminado + "')", "\"\" + pila.getTextoActual() + "\"\"");

eliminado = pila.pop();
System.out.printf("%-20s|%-8s|", "UNDO (elimina '" +
eliminado + "')", "\"\" + pila.getTextoActual() + "\"\"");
pila.mostrar();

System.out.println("\n - Resultado Final - ");
System.out.println("Contenido del texto: \"\" +
pila.getTextoActual() + "\"\"");
System.out.print("Contenido de la pila: ");

Contenido de la pila: mostrar ();

Acción	Texto actual	Pila de acciones
-	-	-
Inicial	" "	[]
1 - Escribir "A"	"A"	[A]
2 - Escribir "B"	"AB"	[A, B]
3 - Escribir "C"	"ABC"	[A, B, C]
4 - UNDO	"AB"	[A, B]
(Se elimina C)		
5 - Escribir "D"	"ABD"	[A, B, D]
6 - UNDO	"AB"	[A, B]
(Se elimina D)		
7 - UNDO	"A"	[A]
(Se elimina B)		

Contenido final del texto: "A"

Contenido de la pila: [A]

Miguel Lozano Arvizu GTID141

Practica 03 Pilas

Ejercicio 5 - Completa la Semi-Implementación de una Pila en Java.

```
public class Pila {  
    private int [] datos;  
    private int tope;  
  
    public Pila (int capacidad) {  
        datos = new int [capacidad];  
        tope = -1;  
    }  
  
    public void push (int valor) {  
        if (estaLlena()) {  
            System.out.println ("Error: Pila Llena (Overflow). No se  
            puede insertar " + valor);  
            return;  
        }  
        tope ++;  
        datos [tope] = valor;  
    }  
  
    public int pop () {  
        if (estaVacía ()) {  
            System.out.println ("Error: Pila vacía (Underflow). No  
            se puede extraer");  
            return -1;  
        }  
        int valor = datos [tope];  
        tope --;  
        return valor;  
    }  
}
```

```

public int peek () {
    if (estaVacia ()) {
        System.out.println ("Error: Pila vacia. No hay elemento
        en el tope");
        return -1;
    }
    return datos [tope];
}

public boolean estaVacia () {
    return tope == -1;
}

public boolean estaLlena () {
    return tope == datos.length - 1;
}

public void mostrarPila () {
    if (estaVacia ()) {
        System.out.println ("La pila esta vacia");
        return;
    }
    System.out.println ("\n -- Contenido de la Pila --");
    System.out.println ("(De arriba hacia abajo)");
    for (int i = tope; i >= 0; i--) {
        System.out.println ("| " + datos[i] + " |");
    }
    System.out.println ("-----");
}

```

```
public static void main (String [] args) {  
    System.out.println ("--- Ejercicio 5 ---\n");  
  
    Pila pila = new Pila (5);  
    System.out.println ("--- Prueba 1: Insertar elementos ---");  
    pila.push (10);  
    pila.push (20);  
    pila.push (30);  
    pila.mostrarPila ();  
  
    System.out.println ("\n-- Prueba 2: peek() --");  
    System.out.println ("Elemento en el tope: " + pila.peek ());  
  
    System.out.println ("\n-- Prueba 3: pop() --");  
    System.out.println ("Extraido: " + pila.pop ());  
    System.out.println ("Extraido: " + pila.pop ());  
    pila.mostrarPila ();  
  
    System.out.println ("\n-- Prueba 4: llenar la pila --");  
    pila.push (40);  
    pila.push (50);  
    pila.push (60);  
    pila.push (70);  
    pila.mostrarPila ();  
    System.out.println ("\n-- Prueba 5: Vaciar pila --");  
    while (!pila.estaVacia ()) {  
        System.out.println ("Extraido: " + pila.pop ());  
    }  
    pila.pop ();  
}
```

System.out.println("\n -- Prueba 6: Estado final --");
System.out.println("¿Esta vacía?" + pila.estaVacia());
pila.mostrarPila();
}
}

Practica 03 Pilas

Ejercicio 6: Aplicación conceptual.

1.- Navegación "atras" en un navegador:

Estructura adecuada: PILA (Stack).

Por que usa el principio LIFO (Last In, First Out), la ultima pagina visitada es la primera en volver al presionar "atras".

La pila guarda cada URL. Al hacer clic en "atras", se extrae la ultima URL visitada.

2.- Evaluación de expresiones matematicas

Estructura adecuada: PILA (Stack)

Porque permite procesar operadores respetando precedencia.

La pila facilita el manejo de parentesis y precedencia de operadores (* y / antes que + y -).

3.- Verificación de Parentesis balanceados

Estructura adecuada: Pila (Stack)

Porque es parecido al ejercicio 2, pero verificando balance, se usan las mismas operaciones PUSH/POP como los ejercicios.

4. Implementación de Sistema deshacer (UNDO)

Estructura adecuada: PILA (stack).

Porque es exactamente como el ejercicio 4 que realice, El ultimo elemento agregado es el primero en salir (LIFO), tal como vimos en el ejercicio 1 como el PUSH.

El ultimo elemento es el primero en procesarse.