

# **A Pilha**

## **Aula 01**

---

**DPEE 1038 – Estrutura de Dados para Automação**  
**Curso de Engenharia de Controle e Automação**  
**Universidade Federal de Santa Maria**

---

**Prof. Rafael Concatto Beltrame**  
**beltrame@mail.ufsm.br**

# Sumário

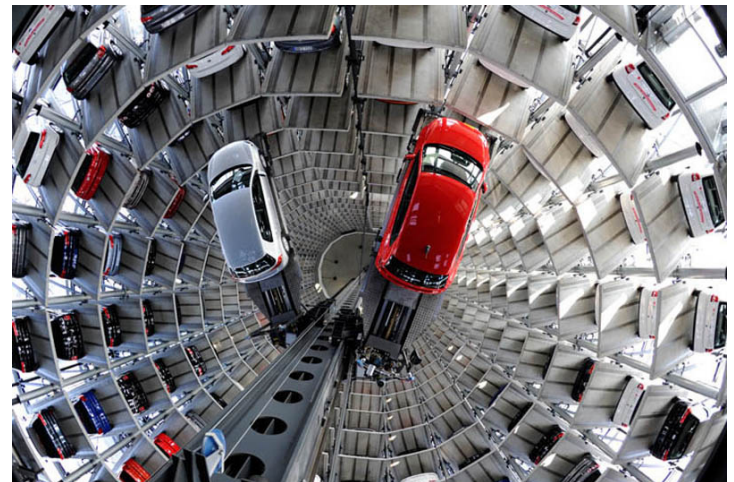
- **Definição e exemplos**
- **Operações primitivas**
  - `push( )` e `pop( )`
  - `empty( )`
  - `stacktop( )`
- **Exemplo**



# Pilha

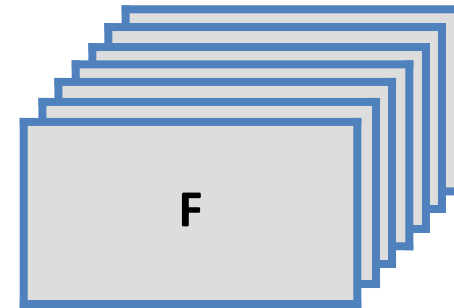
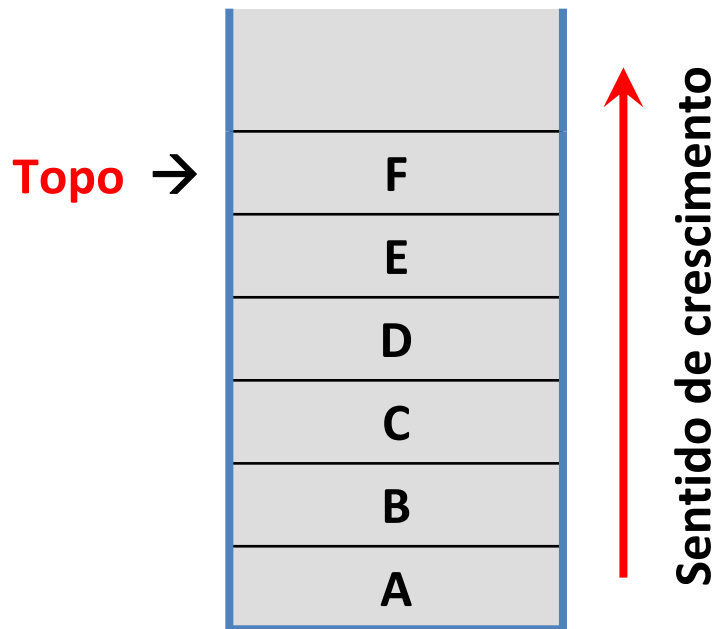
## Definição

- **Conjunto ordenado de itens** no qual novos itens podem ser inseridos e/ou eliminados em uma extremidade chamada **topo da pilha**
  - Diferentemente do vetor, a pilha é um objeto dinâmico, constantemente **mutável**



# Pilha

- Identificação dos elementos da pilha



“F” foi o último elemento a ser inserido e será o primeiro a ser eliminado

- last-in, first-out (LIFO)

Verdadeira imagem de uma pilha:

- Visualização de cima para baixo
- Não se sabe quantos itens compõe a pilha

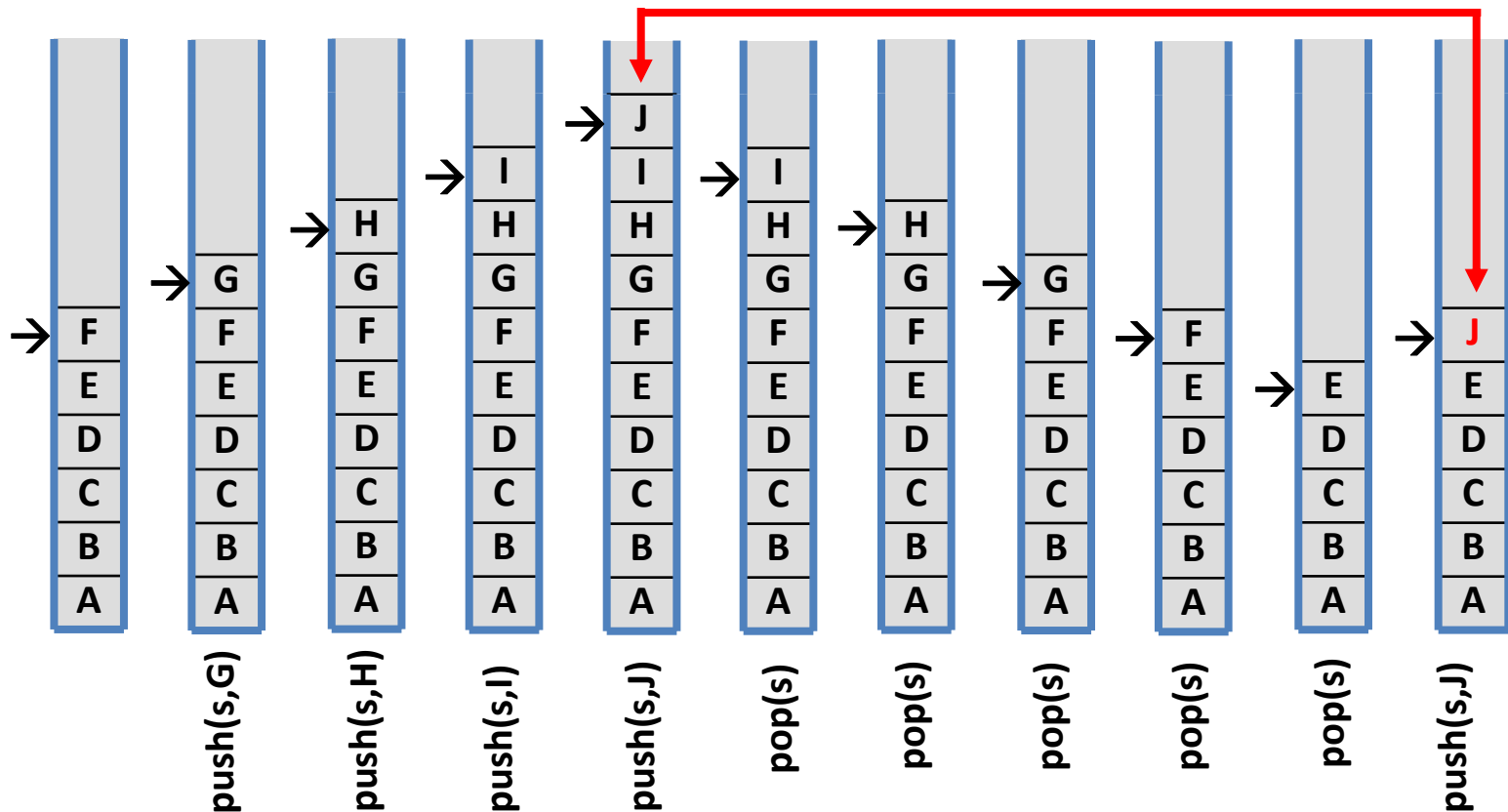
# Operações Primitivas

- Empilhar → Incluir um item no topo:

`push(s, i)`

- Desempilhar → Retirar um item do topo:

`pop(s)`



# Operações Primitivas

- Teoricamente, não existe **limite máximo** para a pilha
  - Na prática, é limitado pela memória física do computador
- Existe um **limite mínimo**: **pilha vazia**
  - Não deve ser permitido aplicar a operar **pop()** a uma pilha vazia (**underflow**)
  - Teste se a pilha está vazia: `empty(s)`
- Leitura do topo, sem desempilhá-lo: `i = stackltop(s)`

```
i = stacktop(s);      // Chamada da função

// Equivalente à
i = pop(s);           // Desempilha e salva topo em "i"
push(s, i);           // Empilha o valor novamente
```

# Exemplo

- Seja a expressão matemática

$$7 - ((X * ((X + Y) / (J - 3)) + Y) / (4 - 2.5))$$

- Verificar se

- 1) Existe um **número igual** de parênteses **esquerdos** e **direitos**
- 2) Todo parêntese da **direita** está **precedido** por um parêntese da **esquerda** correspondente

- Expressões

$((A + B)$  ou  $A + B($  violam o **critério 1**

$)A + B(-C$  ou  $(A + B) - (C + D$  violam o **critério 2**

# Exemplo

- Solução
  - Parêntese esquerdo → Abertura de escopo
  - Parêntese direito → Fechamento de escopo
  - Profundidade de aninhamento (profundidade de agrupamento)
    - Número de escopos abertos e não-fechados
    - No exemplo: nº de parênteses esquerdos encontrados cujos correspondentes parênteses direitos não foram encontrados

7	-	(	(	X	*	(	(	X	+	Y	)	/	(	J	-	3	)	)	+	Y	)	/	(	4	-	2.5	)	)
0	0	1	2	2	2	3	4	4	4	4	3	3	4	4	4	4	3	2	2	2	1	1	2	2	2	2	1	0

(	(	A	+	B	)
1	2	2	2	2	1

)	A	+	B	C	(	-	C
-1	-1	-1	-1	-1	0	0	0



# Exemplo

- Avaliar a expressão matemática

$(A + B) + \{C * [(A + B) * (D - E)] + F\} * G$

- Solução

- Três tipos de escopo

- parênteses **( )**, colchetes **[ ]**, chaves **{ }**
- Um finalizador deve ser do mesmo tipo que seu iniciador
- Expressões inválidas:  $(A+B]$  ,  $[(A+B)]$  ,  $\{A-(B] \}$

- É necessário rastrear o número de escopos abertos/fechados e também seus tipos

# Exemplo (Algoritmo)

```
valid = true;
s      = pilha vazia

while (não encontrar o caracter \0) {
    symb = fc_ler_proximo_caractere();
    if (symb == '(' || symb == '[' || symb == '{')
        push(s, symb);
    if (symb == ')' || symb == ']' || symb == '}')
        if (empty(s))
            valid = false;
        else {
            i = pop(s);
            if (i != Abertura de escopo de symb)
                valid = false;
        }
}

if (!empty(s))
    valid = false;
```

# Revisão

- **Pilha**
  - **Conjunto ordenado de itens** no qual novos itens podem ser inseridos e/ou eliminados em uma extremidade chamada **topo da pilha**
- **Visualização de uma pilha**
  - Visualização de **cima para baixo**. Não se sabe quantos itens possui
- **Operações primitivas**
  - **push(s, i)** → Copia o dado 'i' para a o topo da pilha 's' (empilha)
  - **pop(s)** → Retira o dado presente no topo da pilha 's' (desempilha)
  - **empty(s)** → Verifica se a pilha 's' está vazia
  - **stacktop(s)** → Lê o dado do topo da pilha 's' sem desempilhá-lo