

# *Tecnologia em Análise e Desenvolvimento de Sistemas - TADS*

## ***Estrutura de Dados***

***Prof. Luciano Vargas Gonçalves***

*E-mail: [luciano.goncalves@riogrande.ifrs.edu.br](mailto:luciano.goncalves@riogrande.ifrs.edu.br)*



# Estrutura da Dados

- **Aula 6 – Estruturas Dinâmicas**
  - **Pilhas**

# Sumário

- **Estruturas de Dados**

- **Dinâmicas:**

- O tamanho se altera com a necessidade;
    - Cresce ou Decresce

- Listas

- Simplesmente Encadeadas
          - Listas
          - Filas
          - **Pilhas**
        - Duplamente Encadeadas

# Pilha

- **Características**

- Estruturas para armazenar informações;
- Semelhante a uma lista encadeada ***com restrições***;
- Muito utilizada para controlar sistema de montagem e desmontagem (sequenciais);
- Exemplos



Pilha de Livros



Pilha de Roupas

# Pilha

- **Características**

- Quais Restrições??
- Podemos retirar qualquer peça??
  - Pegamos o prato que esta no topo!!!!
  - Colocamos de volta o prato no topo!!!



Pilha de Pratos

# Pilha

- **Características**

- Quais Restrições??

- Podemos pegar qualquer Carta no Jogo Cartas??

- Regras!!!!

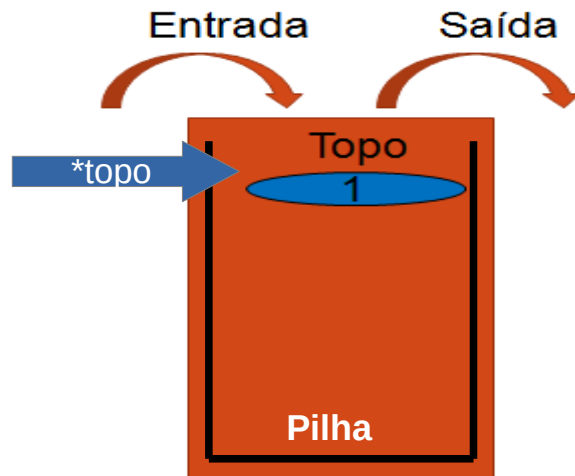
- Pegar somente a que estiver no topo!!!



Jogo De Cartas

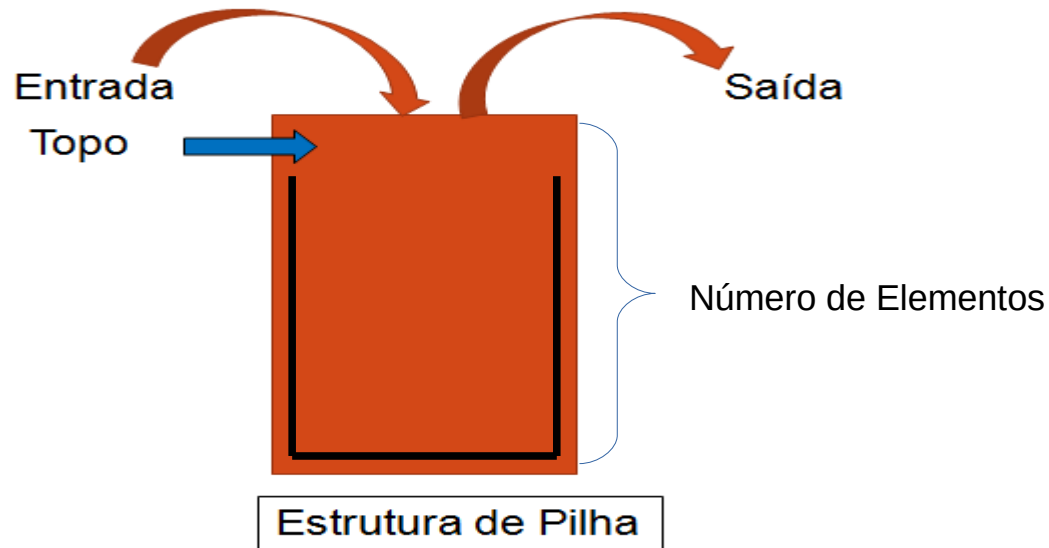
# Pilha

- Quais Restrições?? Estrutura de Pilha?
  - Só podemos retirar o elemento da pilha que está **no topo**. Sendo o elemento que está diretamente acessível;
  - Após a retirada do elemento que está no topo, o segundo elemento se torna acessível (Topo);
  - A inserção só pode ocorrer **no topo** da pilha;
  - Característica principal:
    - Inserção e Remoção pela mesma extremidade.
      - Ponteiro para o topo da Pilha;



# Pilha

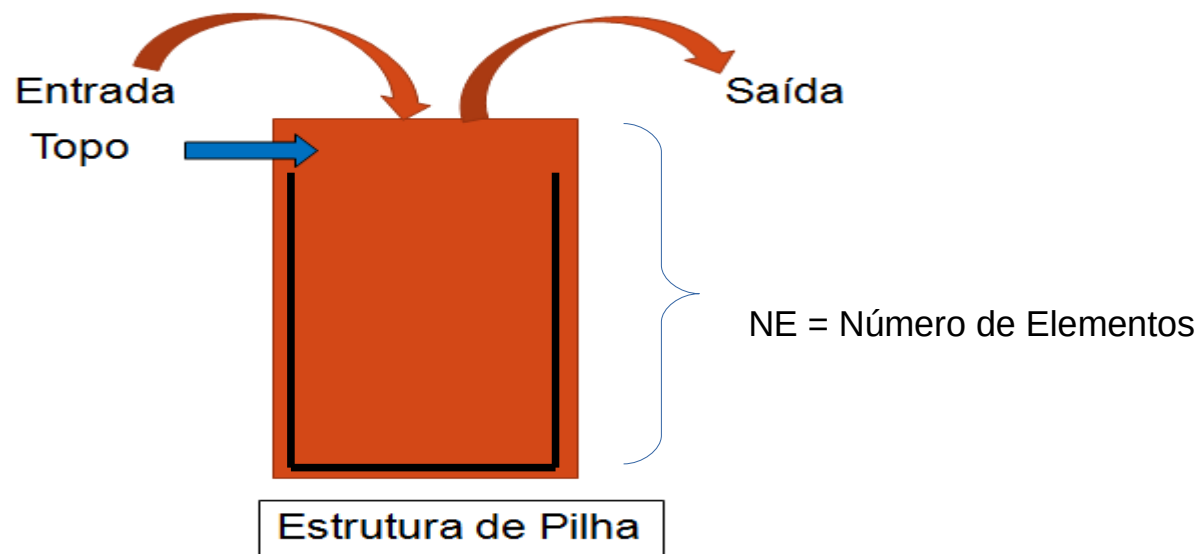
- Estrutura de Pilha:
  - Entrada e saída pelo mesma extremidade;
  - Topo é um apontador ou ponteiro para último elemento inserido;





# Pilha

- Estrutura de Pilha:
  - LIFO – Last IN Firt Out (o último a entrar é o primeiro a sair)

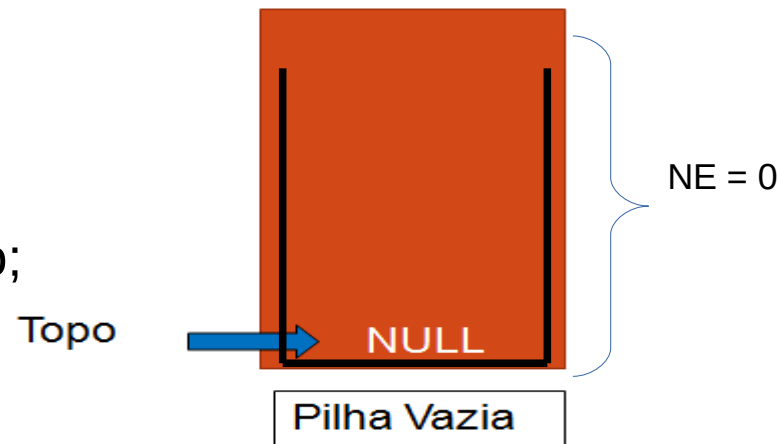


# Pilha

- Estrutura de Pilha:

- **Pilha Vazia**

- Topo aponta para NULL;
    - Número de Elementos é igual a zero;

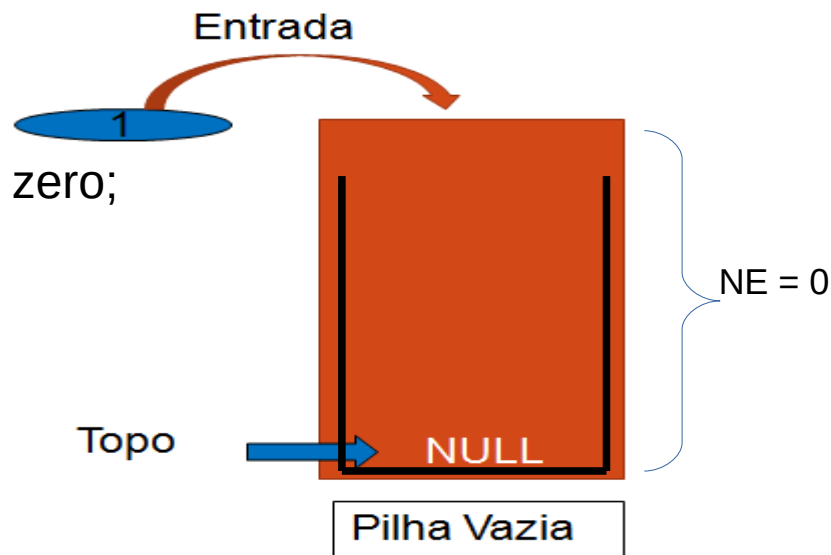


# Pilha – Comando Push

- Estrutura de Pilha:

- ***PUSH (empurar)***

- Pilha Vazia, número de elementos igual a zero;
    - \*Topo aponta para NULL;
    - ***Equivalente ao Inserir Elemento no Início;***

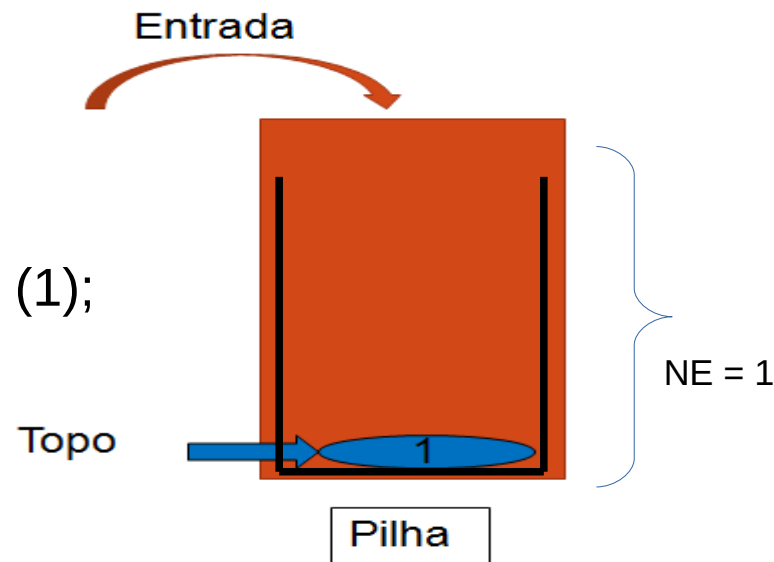


# Pilha – Comando Push

- Estrutura de Pilha:

- ***Após PUSH (empurar)***

- Número de Elementos é igual a 1;
    - \*Topo aponta para o primeiro elemento (1);

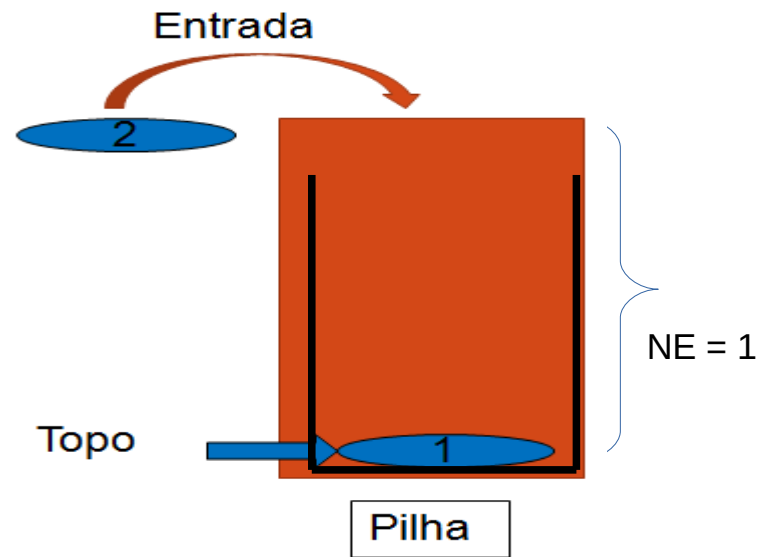


# Pilha – Comando Push

- Estrutura de Pilha:

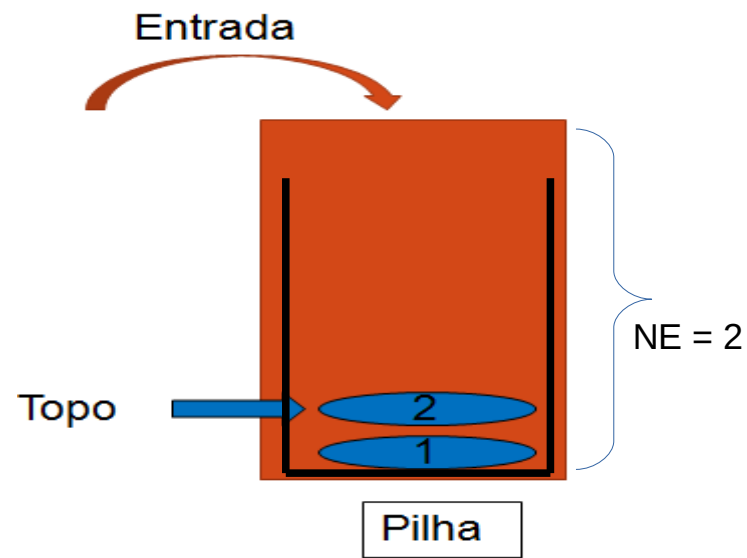
- ***Antes do PUSH (empurar) o elemento 1;***

- Número de Elementos é igual a 1;
    - \*Topo aponta para o primeiro elemento (1);



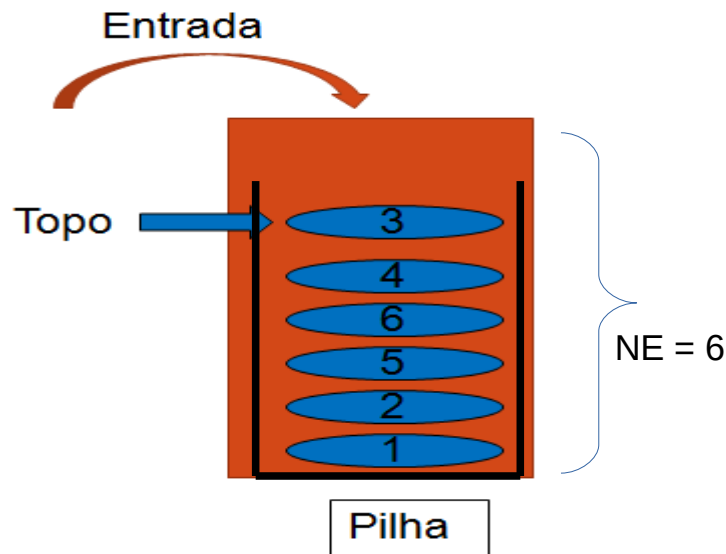
# Pilha – Comando Push

- Estrutura de Pilha:
  - *Após o PUSH do elemento 2;*
    - Número de Elementos é igual a 2;
    - \*Topo aponta para o elemento (2);



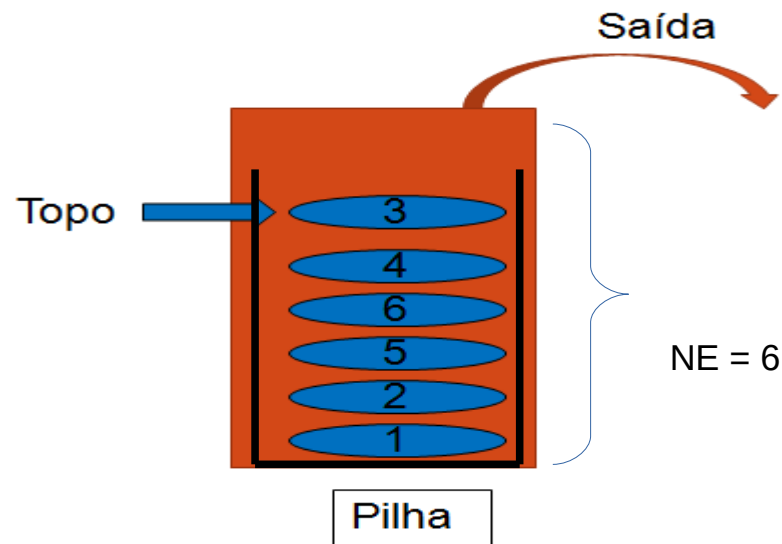
# Pilha – Comando Push

- Estrutura de Pilha:
  - ***Após PUSH (empurar) de vários elementos o elemento 3 é o último inserido;***
    - Número de Elementos é igual a 6;
    - \*Topo aponta para o elemento (3);
    - Vários elementos inseridos;



# Pilha – Comando POP

- Estrutura de Pilha (POP):
  - **POP (saltar da pilha) -**
    - Retira o elemento que está sendo apontado pelo ponteiro \*topo;
    - \*Topo aponta para o elemento (3);
    - Número de elementos igual a 6;
    - ***Equivale a função remove no Início de uma LDE;***



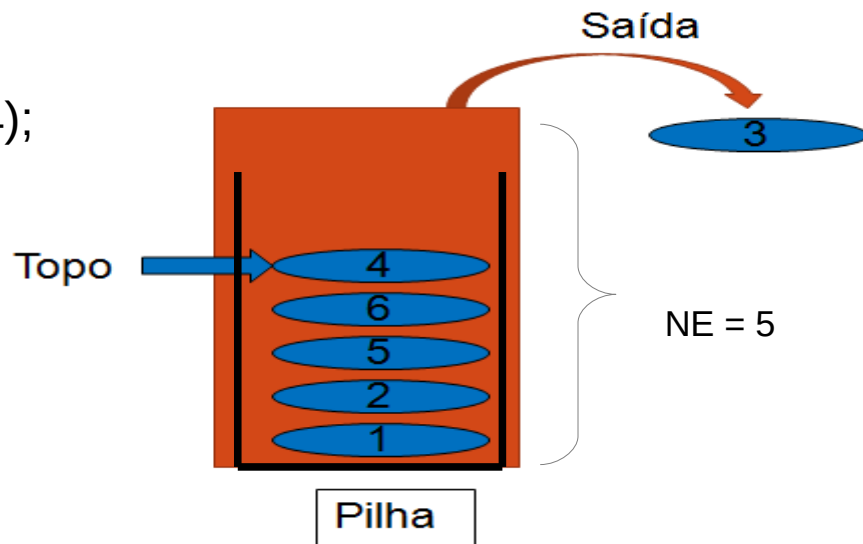


# Pilha – Comando POP

- Estrutura de Pilha (POP):

- *Após o POP do elemento (3);*

- \*Topo passa a apontar para o elemento (4);
    - Número de Elementos fica igual a 5;
    - O elemento (3) deixa a pilha;

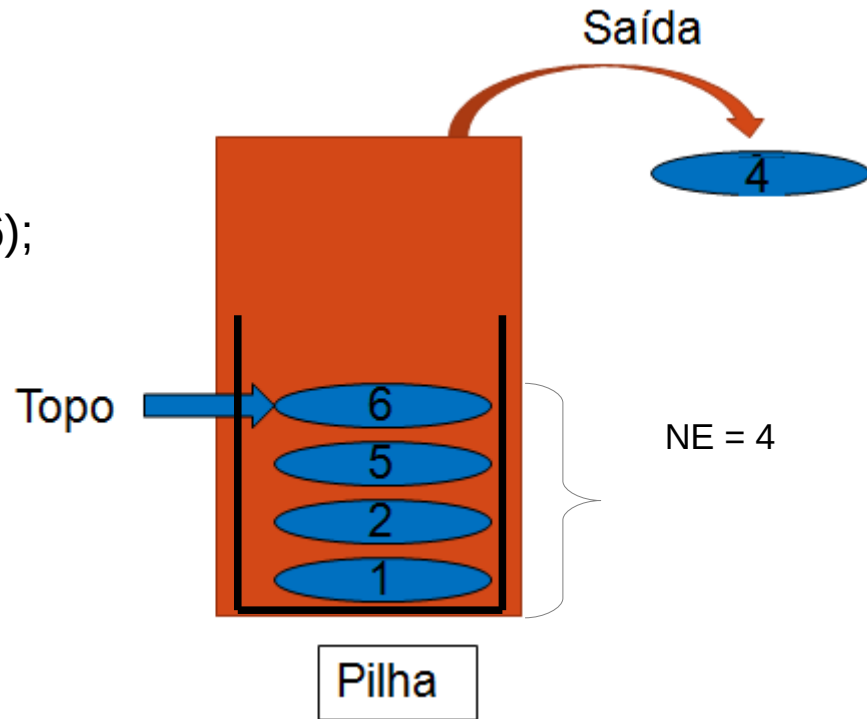


# Pilha – Comando POP

- Estrutura de Pilha (POP):

- *Após o POP do elemento (4);*

- \*Topo passa a apontar para o elemento (6);
    - Número de Elementos fica igual a 4;
    - O elemento (4) deixa a pilha;

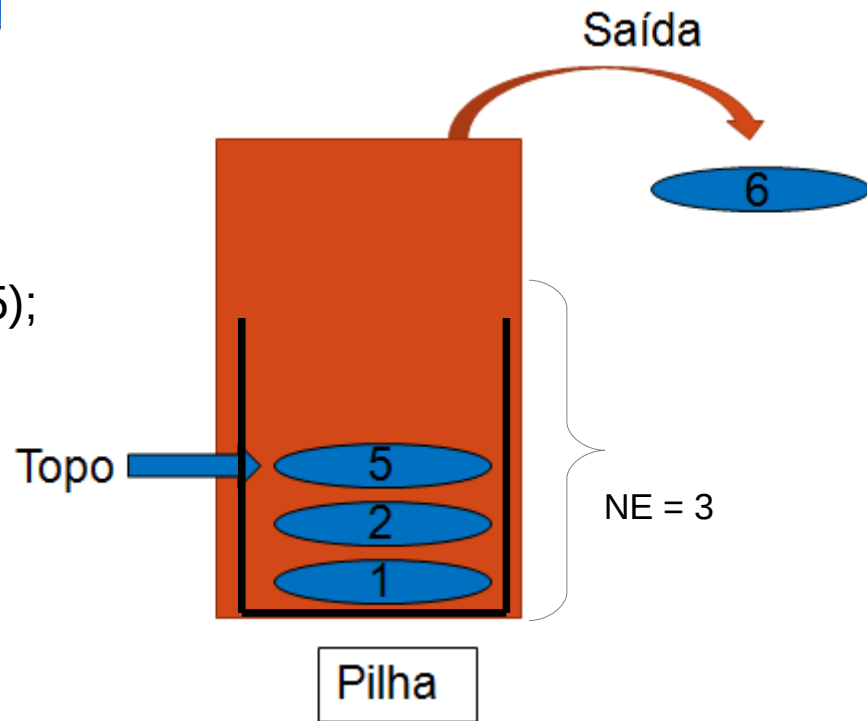


# Pilha

- Estrutura de Pilha (POP):

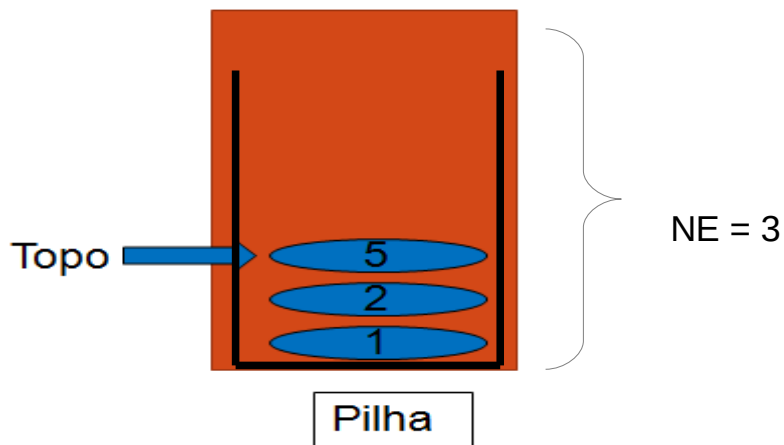
- *Após o POP do elemento (6);*

- \*Topo passa a apontar para o elemento (5);
    - Número de Elementos fica igual a 3;
    - O elemento (6) deixa a pilha;



# Pilha

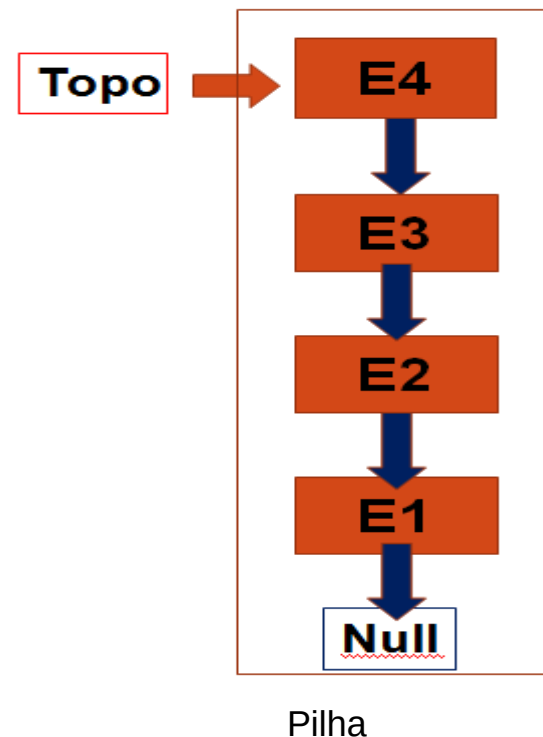
- Pilha (Consultar um Elemento):
  - Apenas o elemento que está no TOPO (5) pode ser consultado;
  - Consultar os demais elementos, somente ***desempilhando (removendo os elementos que estão no topo)***.



# Pilha

- **Pilha**

- Pilhas são Listas Simplesmente Encadeada (LSE) com restrições:
  - PUSH = Insere no Início (topo)
  - POP = Remove no Início (topo)
  - Consulta primeiro elemento (topo)
  - Não é possível varrer a Pilha, mostrar todos os elementos.



# Pilha

- **Estrutura de Pilha para Livros:**
  - Armazenar o título do livro, autor e ano de publicação;
  - Para apontar próximo elemento, iremos criar um apontador que irá armazenar uma referência para o “**próximo**” elemento da pilha;

```
typedef struct livro{    //d
    char autor [20];
    char titulo [20];
    int ano;
    struct livro *proximo;
}Livro;
```

Estrutura de Pilha para **Livro**

# Pilha

- **Estrutura para gerenciar a pilha;**
  - Quantidade elementos(qtd) e ponteiro para o topo da pilha (\*topo);

```
typedef struct pilha{  
    Livro *topo;  
    int qtd;  
}Pilha;
```

Definição da estrutura Pilha

# Pilha

- **Criar uma nova Pilha de Livros:**

- \*Topo aponta para NULL;
- Número de Livros igual a Zero;
- \*p é o parâmetro(ponteiro) que recebe o endereço da nova pilha;

```
Pilha* criaPilha(){  
    Pilha *p = (Pilha *)malloc(sizeof(Pilha));  
    p->topo = NULL;  
    p->qtd = 0;  
    return p;  
}
```

Função para criar uma nova pilha



# Pilha

- Criar um elemento de Pilha para Livro
  - Recebe os dados de um livro e cria um elemento de pilha;
  - Retorna um ponteiro para Livro;

```
Livro* criaElementoPilhaLivro(int ano, char at [], char tit []){  
    Livro *lv = (Livro *)malloc(sizeof(Livro));  
    lv->proximo = NULL;  
    lv->ano = ano;  
    strcpy(lv->autor,at);  
    strcpy(lv->titulo,tit);  
    return lv;  
}
```

# Pilha

- **Função PUSH (empurra)**

- *Empurra um Livro na Pilha;*

- Recebe um ponteiro de pilha(\*p) e um ponteiro de livro(\*lv);
    - Função sem retorno;

```
void push(Pilha *p, Livro *lv){  
    lv->proximo = p->topo;  
    p->topo = lv;  
    p->qtd++;  
}
```

Função PUSH para inserir um elemento na pilha

# Pilha

- **Função POP (remove)**

- Salta um elemento da pilha(\*p) e retorna o endereço do elemento removido.
  - Retorna um ponteiro para um elemento “Livro”;

```
Livro* pop(Pilha *p){  
    //salta um Livro da pilha  
    Livro *aux = p->topo;  
    if(aux != NULL){  
        p->topo = p->topo->proximo;  
        p->qtd--;  
        aux->proximo = NULL;  
    }  
    return aux;  
}
```

# Pilha

- **Funções Mostra Livro e Topo !!**

- Mostra dados de um livro;
- Mostra Elemento que está no Topo da Pilha

```
void mostraLivro(Livro lv){  
    printf("\n\t Titulo: %s \n\t Autor: %s \n\t Ano: %d \n",lv.titulo,lv.autor,lv.ano);  
}
```

```
void mostraTopo(Pilha *p){  
    if(p->topo != NULL){  
        printf("\n Mostra TOPO - Pilha");  
        mostraLivro(*p->topo);  
    }else{  
        printf("\n TOPO NULO - Pilha Vazia \n");  
    }  
}
```

# Exemplo Pilhas de Livros

- Criado duas Pilhas

- p1,p2

```
Pilha *p1,*p2;  
Livro *l1, *l2, *l3, *aux;
```

- Criado três ponteiros de elementos de pilha livros;

- l1,l2,l3

```
p1 = criaPilha();  
p2 = criaPilha();
```

```
l1 = criaElementoPilhaLivro(1812,"Irmaos Grimm","Branca de Neves");  
l2 = criaElementoPilhaLivro(1972,"Camara Cascudo","Mula sem Cabeça");  
l3 = criaElementoPilhaLivro(1852,"Joseph Jacobs","Três Porquinhos");
```

- Inserido os três elementos na pilha 'p1'

```
push(p1,l1);  
push(p1,l2);  
push(p1,l3);
```

# Exemplo Pilhas de Livros

- Pilha após inserções
  - Ordem inversa a inserção

Mostra TOP0:

Mostra TOP0 - Pilha

Titulo: Três Porquinos

Autor: Joseph Jacobs

Ano: 1812

Titulo: Mula sem Cabeça

Autor: Camara Cascudo

Ano: 1812

Titulo: Branca de Neves

Autor: Irmaos Grimm

Ano: 1812

Saída da pilha P1

# Exemplo Pilhas de Livros

- Menu das operações sobre as pilhas P1 e P2

Informe uma Opção:

- 1 - para PUSH:
- 2 - para POP:
- 3 - para Mostra TOP0:
- 4 - para Apagar Pilha:
- 5 - para Mover um Livro de P1 para P2:
- 6 - para Mover um Livro de P2 para P1:
- 7 - Mostrar Pilhas:
- 0 - para Sair do Programa:

Informe sua Opção:0

# Exercícios

## 1) Implemente a Função inverte Pilha P1

- Função recebe um pilha P1 e retorna um nova pilha (cópia P1) invertida;

## 2) Implemente a Função inverte Pilha 2

- Recebe um pilha P1 e inverte a própria pilha P1 (invertida);

## 3) Implemente uma função que troque o topo com o primeiro elemento da Pilha, sem alterar a ordem dos demais;

- Função recebe um pilha e troca os elementos na mesma pilha;