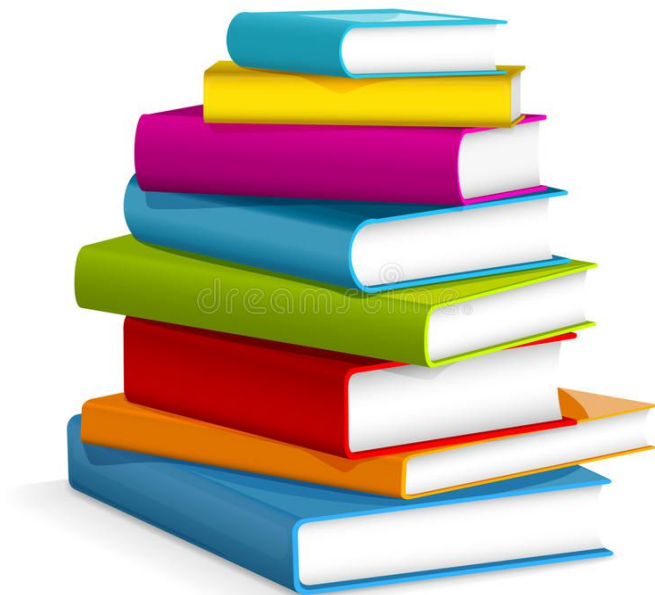




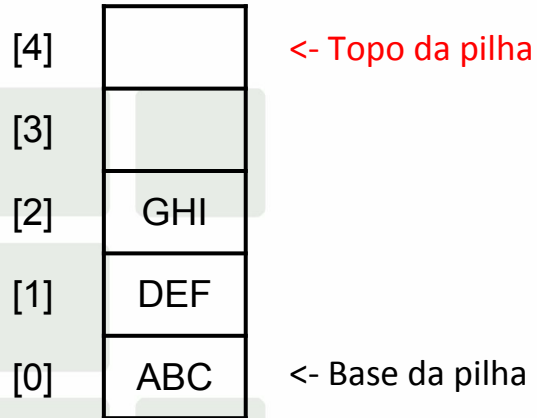
Programação O. O. II - Pilha

Prof. Ronaldo Serpa da Rosa
ronaldo.rosa@bento.ifrs.edu.br

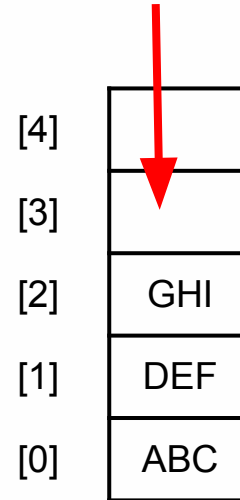
Pilha



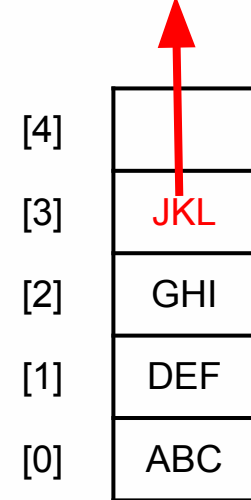
Pilha



Empilhar



Desempilhar



LIFO - LAST IN FIRST OUT
ÚLTIMO A ENTRAR - PRIMEIRO A SAIR

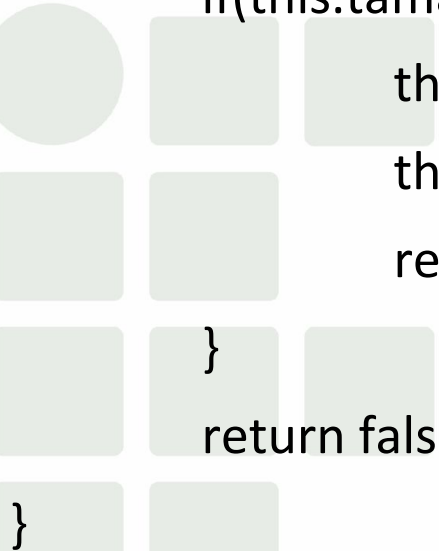
Implementação da Pilha

```
public class Pilha {  
    private String [] elementos;  
    private int tamanho;  
    public int getTamanho() {  
        return tamanho;}  
    public Pilha(int capacidade) {  
        this.elementos = new String[capacidade];  
        this.tamanho = 0;  
    }  
}
```



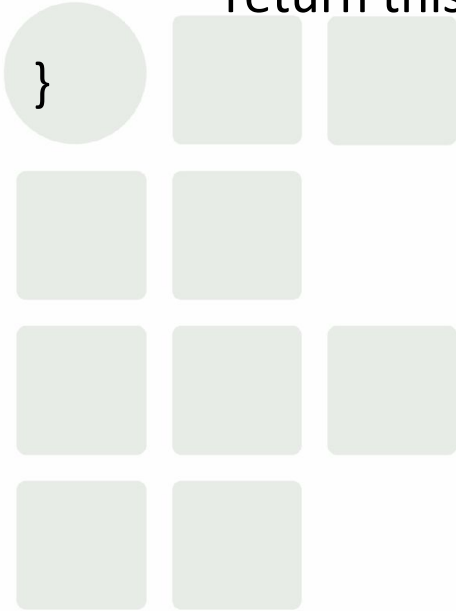
Empilhar elementos

```
public boolean empilhar(String elemento) {  
    if(this.tamanho<this.elementos.length) {  
        this.elementos[this.tamanho] = elemento;  
        this.tamanho++;  
        return true;  
    }  
    return false;  
}
```



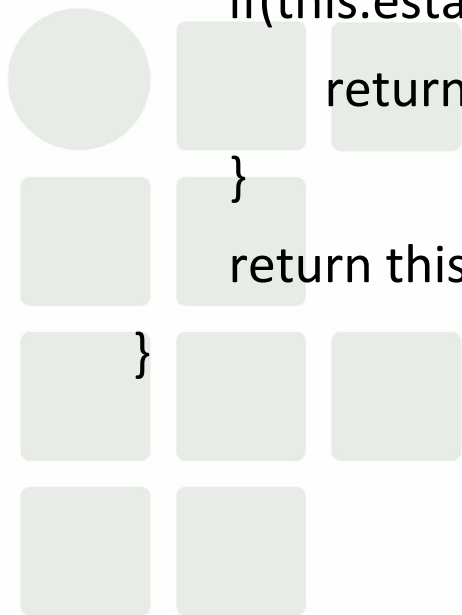
Verificando se a pilha está vazia

```
public boolean estaVazia() {  
    return this.tamanho == 0;  
}
```



Visualizando o topo da pilha

```
public String topo() {  
    if(this.estaVazia()) {  
        return null;  
    }  
    return this.elementos[tamanho-1];  
}
```



Desempilhando um elemento

```
public String desempilhar() {
```

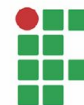
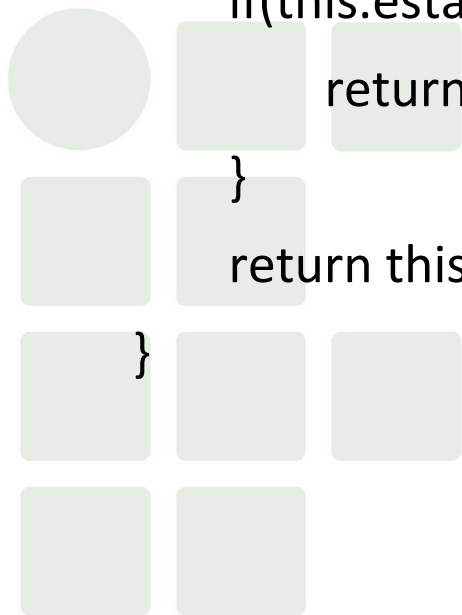
```
    if(this.estaVazia()) {
```

```
        return null;
```

```
    }
```

```
    return this.elementos[--this.tamanho];
```

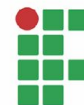
```
}
```



Imprimindo os elementos da pilha

@Override

```
public String toString() {  
    StringBuilder s = new StringBuilder();  
    s.append("[");  
    for(int i=0;i<tamanho;i++) {  
        s.append(this.elementos[i]);  
        s.append(",");    }  
    if(this.tamanho>0) {  
        s.append(this.elementos[this.tamanho-1]);    }  
    s.append("]");  
    return s.toString();}
```



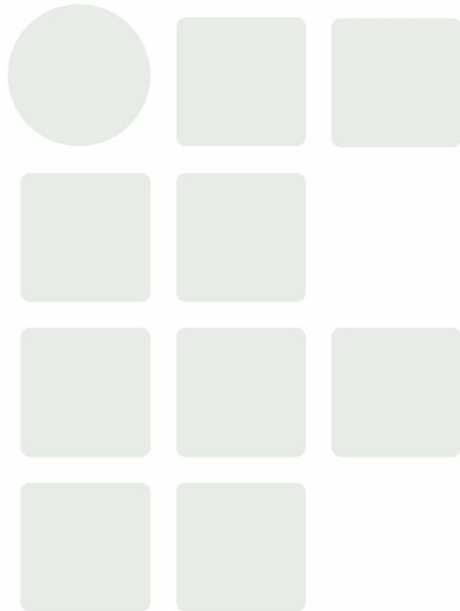
Aumentando a capacidade do vetor

```
private void aumentaCapacidade() {  
    if(this.tamanho == this.elementos.length) {  
        String[] elementosNovos = new String[this.elementos.length*2];  
        for(int i=0;i<this.elementos.length;i++) {  
            elementosNovos[i] = this.elementos[i];  
        }  
        this.elementos = elementosNovos;  
    }  
}
```

*******Chamar este método na primeira linha do método empilhar*******

Exercício 1

- Criar uma classe chamada Pilha e implementar os métodos de acordo com conteúdo apresentado do slide 5 até o slide 11



Exercício 2

- Criar uma classe chamada TestaPilha (com método main);
- Importar a classe Pilha (criada anteriormente);
- Realizar as operações abaixo utilizando os métodos da classe Vetor;
 - Criar um vetor com capacidade para 5 elementos;
 - Verificar e imprimir se a pilha está vazia;
 - Adicionar 3 elementos na pilha;
 - Imprimir o tamanho da pilha;
 - Imprimir todos os elementos da pilha;
 - Imprimir o elemento do topo da pilha;
 - Adicionar mais 3 elementos na pilha;
 - Verificar e imprimir se a pilha está vazia;
 - Imprimir o elemento do topo da pilha;
 - Desempilhar 1 elemento da pilha;
 - Imprimir o elemento do topo da pilha;
 - Desempilhar 1 elemento da pilha;
 - Imprimir todos os elementos da pilha novamente;
 - Imprimir o tamanho da pilha novamente;



Exercício 3 - Pilha pares e ímpares

- Escreva um programa que leia 10 valores números. Para cada número lido verifique e codifique de acordo com as regras a seguir:
 - Se o número for par, empilhe na pilha;
 - Se o número for ímpar, desempilhe um número da pilha. Caso a pilha esteja vazia, mostre uma mensagem;
 - Se ao final do programa a pilha não estiver vazia, desempilhe todos os elementos e os imprima na tela.

Exercício 4 - Jogo Pilha

- Escreva um programa que crie 3 pilhas com tamanho 3 cada uma.
- Para cada pilha você irá gerar 3 valores aleatórios (1-9) e adicioná-los na pilha.
- Após você irá desempilhar um valor de cada pilha e compara los. O maior valor entre os elementos receberá como pontuação a soma dos elementos.
- Em caso de empate entre 1, 2 ou 3 valores, a soma ficará aculmulada para o ganhador da próxima rodada.
- Ao final das três rodadas, o jogador que tiver mais pontos ganha o jogo.

Referências

- GOODRICH, Michael T.; TAMASSIA, Roberto. Estruturas de dados e algoritmos em Java. 5. ed. Porto Alegre: Bookman, 2013
- <https://www.youtube.com/playlist?list=PLGxZ4Rq3BOBrgumpzz-l8kFMw2DLERdxi>