

Algoritmos e Estruturas de Dados

Aula 2

Prof Dr Tanilson Dias dos Santos

Universidade Aberta do Brasil – UAB
Universidade Federal do Tocantins - UFT



Da Aula Anterior...

- **Na Aula 1 tivemos:**
 - **Breve apresentação da Disciplina;**
 - **Conceitos Básicos de Estruturas de Dados.**
- **Prática de Programação:**
 - **Listas em Python;**
 - **Algumas Funções de Manipulação de Listas.**



Na Aula de Hoje, Veremos...

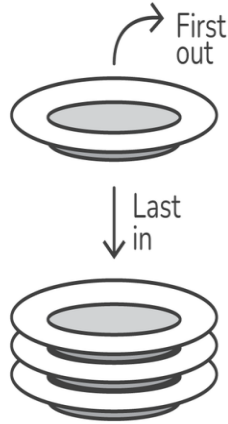
- **Conceito de Pilha:**
 - **Política Associada à Estrutura de Dados Pilha;**
 - **Algumas Aplicações.**
- **Prática de Programação:**
 - **Pilhas em Python;**
 - **Listas como Pilhas.**

O que são pilhas?



- Pilha é uma estrutura de dados que encontramos com alguma frequência no mundo real;
- A pilha consiste em uma estrutura onde você insere elementos um acima do outro, e na hora de retirar, então retira-se os elementos na ordem inversa de inserção;
- Isso define uma política

O que são pilhas?



the LIFO principle
last in - first out

Pilha

- A política ou princípio que está relacionada à Pilha é a LIFO;
- **LIFO – Last in, First Out**
- Em bom português, dizemos que "O último a entrar é o primeiro a Sair", ou ainda de forma análoga:
- **FILO – First in, Last Out**
- "Primeiro a Entrar é o último a Sair"

Onde Vemos Pilhas?



Pilha de Livros

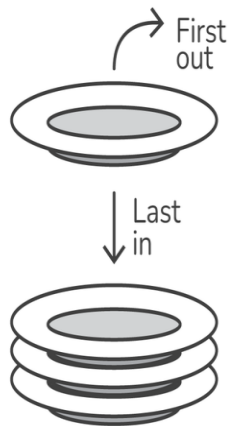


Pilha de Legos



Pilha de Pratos

Funcionamento da Estrutura



the LIFO principle
last in - first out

Pilha

- As inserções na Pilha ocorrem sempre no TOPO da pilha (**push**);
- Analogamente, as remoções também ocorrem sempre no TOPO da pilha (**pop**).

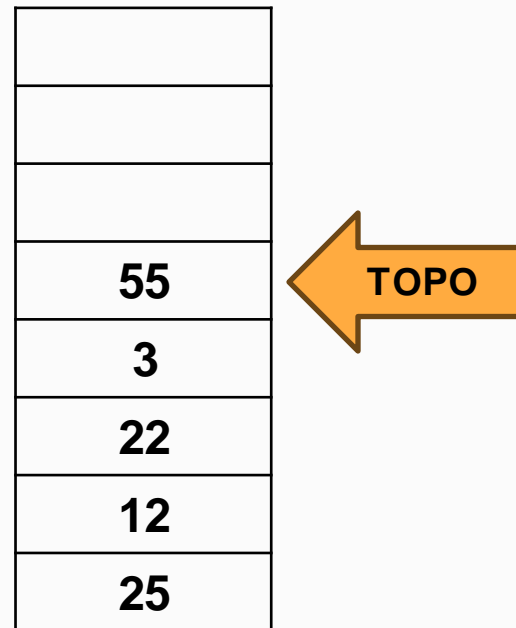
Listas como Pilhas

- Perceba que a estrutura de dados Lista, que já estudamos, é um tipo mais genérico de Pilha;
- Inclusive a Lista possui funções de Pilha;
- É possível manipular uma lista utilizando somente as funções de pilha e assim emular o funcionamento de uma pilha na lista;
- Vamos ver como isso é possível?

Listas como Pilhas

```
pilha = [25, 12, 22, 3, 55]  
print("Pilha: ", pilha)
```

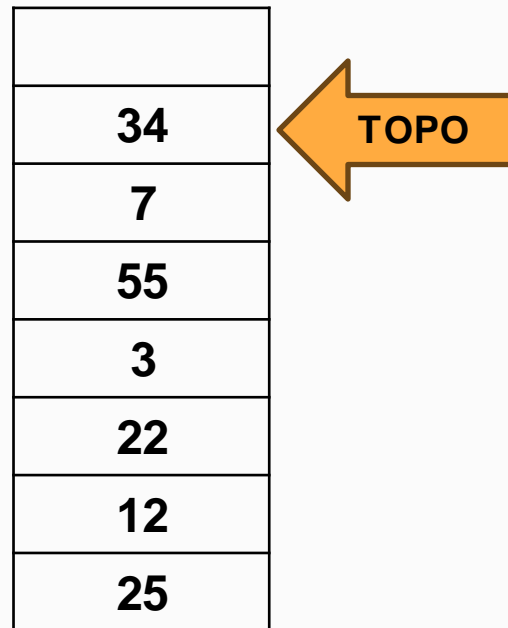
- Utilize a função "append()" para inserir os elementos 7 e 34 na pilha.
- Como fica a nova configuração?
- Quem é o novo topo?



Listas como Pilhas


```
pilha = [25, 12, 22, 3, 55]  
print("Pilha: ", pilha)
```

- pilha.append(7)
- pilha.append(34)
- pilha = [25, 12, 22, 3, 55, 7, 34]
- 34 agora é o topo da pilha



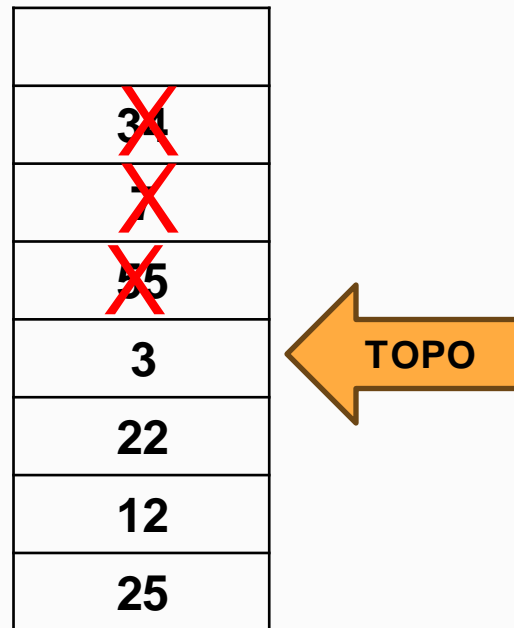
Listas como Pilhas

- pilha.**pop()** - Remove o último item da lista/pilha, e devolve como retorno o elemento removido.
- O que acontece se executarmos o comando **pilha.pop() 3 vezes** na nossa estrutura?
- Como fica a nova configuração?

	 TOPO
34	
7	
55	
3	
22	
12	
25	

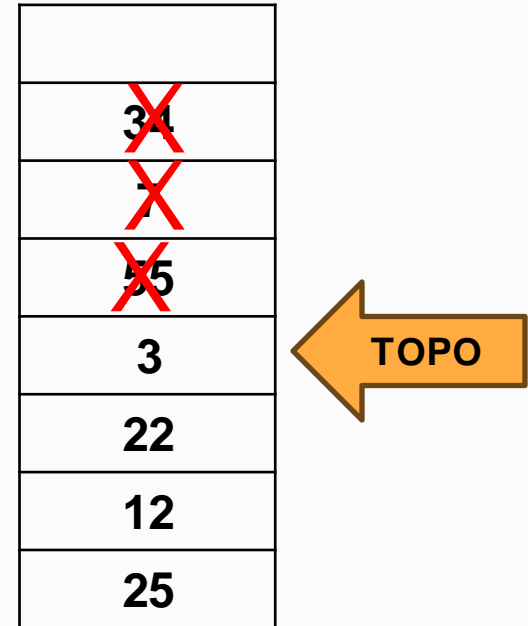
Listas como Pilhas

- O que acontece se executarmos o comando **pilha.pop()** 3 vezes na nossa estrutura?
- Como fica a nova configuração?
- Percebam que como a remoção e inserção de elementos na pilha ocorrem sempre no final, então essas operações sempre consomem tempo **$O(1)$** para serem executadas.



Listas como Pilhas

- A notação **$O(1)$** descrita no slide anterior faz referência à complexidade computacional assintótica do problema;
- Esse tema será abordado com mais profundidade na aula 4.





Listas Como Pilhas são uma Boa Solução?

- Manter uma lista como pilha pode ser fácil, mas seu código pode ser facilmente modificado e por algum motivo você pode se esquecer como a estrutura deve funcionar e fica tentado a utilizar funções da lista para manipular a sua pilha;
- Uma solução alternativa seria a criação de um Tipo Abstrato de Dado ou utilizar uma biblioteca específica para manipulação de pilhas;
- Vamos tentar a primeira abordagem, e vamos construir a nossa própria classe Pilha.



Listas Como Pilhas são uma Boa Solução?

- Alguns pontos importantes para entender:
 1. Essa disciplina não é focada na orientação a objetos, portanto vamos aprender somente alguns conceitos básicos sobre classes e objetos de forma que seja o suficiente para desenvolvermos nossa proposta;
 2. Vocês terão uma disciplina específica no próximo módulo para tratar somente questões de orientação a objetos em linguagens de programação;
 3. Dito isso, mãos ao código!

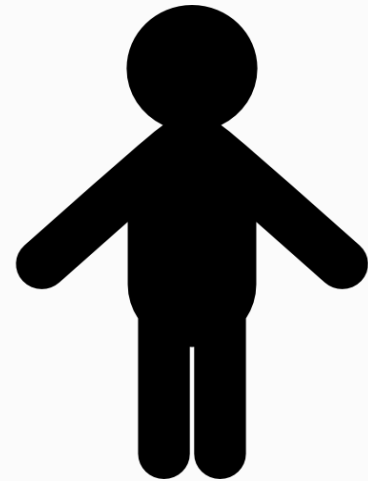


Conceitos Básicos de Orientação a Objetos

- **Classe** - é um molde ou modelo genérico que vai servir como base para a construção de objetos particulares. A Classe possui uma "fôrma" com as características (atributos) e comportamentos (métodos) que o objeto real deve ter;
- **Objeto** - é uma instância de uma classe. É uma classe que ganhou atributos próprios que o diferencia de todos os demais;
- Primeiro fazemos a modelagem da classe para depois poder instanciar o objeto.

Conceitos Básicos de Orientação a Objetos

- Exemplo de classes e objetos.
- Pense que queremos modelar uma pessoa.....
- Precisamos pensar em uma classe Pessoa, porque pessoa é um elemento genérico;
- Na classe você só indica quais as características que a pessoa deve ter.



Conceitos Básicos de Orientação a Objetos

Class Pessoa:

#características

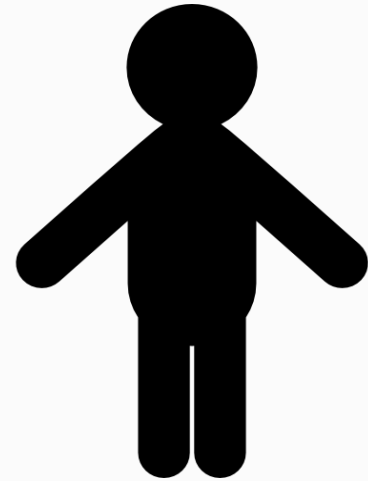
Nome

Sexo

#comportamentos

caminhar()

comer()



Conceitos Básicos de Orientação a Objetos

Fulano = Pessoa()

Fulano.comer()

Fulano.caminhar()

#Fulano vai ter um nome


#Fulano vai ter um sexo

#Percebam que dessa forma Pessoa

#deixou de ser um elemento genérico e

#se tornou um objeto!





```
class Pilha:
    def __init__(self):
        self.items = [ ]

    def estaVazia(self):
        return self.items == [ ]

    def push(self, item):
        self.items.append(item)

    def pop(self):
        return self.items.pop()

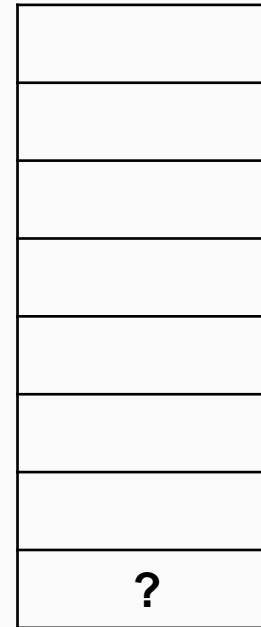
    def topo(self):
        return self.items[len(self.items)-1]

    def tamanho(self):
        return len(self.items)
```

TAD Pilhas

```
p = Pilha()  
print( p.estaVazia() )
```

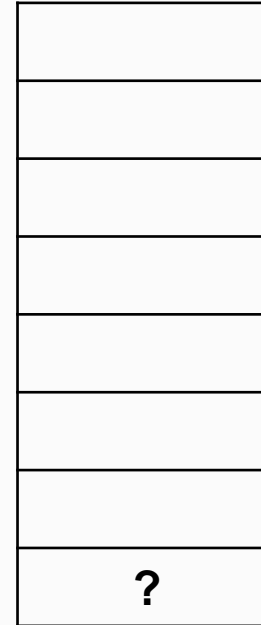
- Insira o código acima após a declaração da classe.
- Qual a saída?



TAD Pilhas

```
p.push(10)
p.push(15)
p.pop()
p.push(8)
p.push(3)
p.pop()
p.push(1)
p.push(5)
print( p.estaVazia() )
print( p.tamanho() )
```

- Insira o código ao lado dessa vez.
- Qual a saída?



TAD Pilhas

```
p.push(10)
p.push(15)
p.pop()
p.push(8)
p.push(3)
p.pop()
p.push(1)
p.push(5)
print( p.estaVazia() )
print( p.tamanho() )
```

- Insira o código ao lado dessa vez.
- Qual a saída?

5
1
8
10



TAD Pilhas

```
p2 = Pilha()  
p2.push(p.pop())  
p2.push(15)  
p2.push(p.pop())  
print( p2.topo() )
```

- Insira o código acima, sem retirar o código anterior.
- Qual a saída? Qual a configuração da pilha p2 ?

5
1
8
10



TAD Pilhas

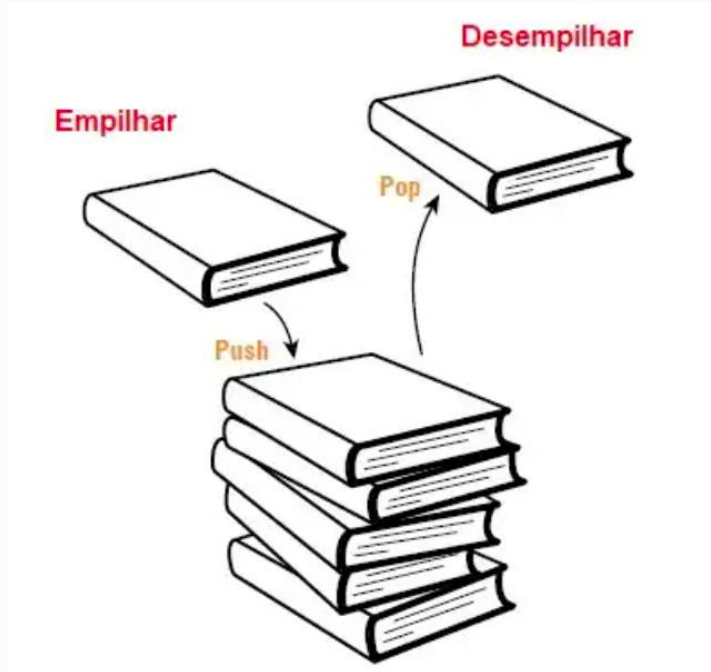
```
p2 = Pilha()  
p2.push(p.pop())  
p2.push(15)  
p2.push(p.pop())  
print( p2.topo() )
```

- Insira o código acima, sem retirar o código anterior.
- Qual a saída? Qual a configuração da pilha p2 ?

1
15
5



Funcionamento da Pilha



- Já deu pra entender como a Estrutura de Dados Pilha funciona?
- **Push()** -> empilha
- **Pop()** -> desempilha



Aplicações com Pilha

- Implementação de uma calculadora com **notação polonesa**, proposta pelo matemático polonês Jan Lukasiewicz (1920) -> Charles Hamblin (1957);
- Balanceamento de pares de parênteses (de símbolos, como colchetes e chaves também);
- Conversão de bases numéricas;
- Implementar recursão em funções;
- Reverter cadeias de caracteres (“*strings*”);
- Problemas usando *backtracking*, revisitando caminhos já passados;
- Desfazer ações.
- Armazenamento de variáveis e chamadas de métodos após ocorrer exceções de *stack overflow*;

Exercício 1

- Considere a pilha p1, inicialmente como descrita ao lado. Como transferir todos os elementos de p1 para p2 **utilizando somente as funções da Classe Pilha que criamos**, de forma que os elementos em p2 permaneçam na mesma ordem?

4
29
32
58
74
18
61



Pilha p1

Exercício 2

- Considere a pilha p1, inicialmente como descrita ao lado. Como remover o elemento **74** de p1 utilizando somente as funções da Classe Pilha que criamos?

4
29
32
58
74
18
61



Pilha p1

Resumo da Aula e Plano de Estudos



Tarefas Semanais

- Refazer Exercícios da Aula;
- Responder Questionário Avaliativo (vale 1.0 ponto);
- Responder Fórum;
- Fazer leitura recomendada;
- Estudar Listas, Filas e Pilhas em Python.



Conclusão e Próxima Aula

- **Aula de Hoje:**
 - **Apresentação principais conceitos de Pilha;**
 - **Tipos Abstratos de Dados e Manipulação de Pilhas;**
 - **Prática de Programação.**
- **Próxima Aula:**
 - **Filas em Python;**
 - **Listas como Filas.**

Algoritmos e Estruturas de Dados

Aula 2

Prof Dr Tanilson Dias dos Santos

Universidade Aberta do Brasil – UAB
Universidade Federal do Tocantins - UFT