Algoritmos e Estrutura de Dados

Bruno Feres de Souza
bferes@gmail.com

Universidade Federal do Maranhão
Bacharelado em Ciência e Tecnologia

1° semestre de 2016

Na aula anterior...

Dados e Tipos de Dados

- Um **dado** é uma informação que um algoritmo recebe ou manipula
- Exemplos de dados são nomes, datas, valores (preços, notas, etc.) e condições (verdadeiro e falso)
- Todo dado é de um certo **tipo** que define sua natureza (p. ex., um nome é diferente de um valor), identificando seu uso, e define as operações que podem ser realizadas com o dado
- Por exemplo, podemos somar dois valores numéricos, mas não podemos somar um número e uma frase

Dados e Tipos de Dados

- Em Python:

  - Tipos de dados **atômicos**:
    - int e float: +, -, *, /, %, **
    - bool: and, or, not

  - Tipos de dados de **coleção**:
    - Listas: criar, acessar, modificar, etc
    - Tuplas: criar, acessar, etc
    - String: criar, acessar, etc
    - Dicionários: criar, acessar, modificar, etc

Estrutura de Dados (ED)

- Definição: organização de dados e operações (algoritmos) que podem ser aplicados sobres esses dados como forma de apoio à solução de problemas.

- Exemplos de EDs:
  - Pilhas
  - Filas
  - Listas lineares
  - Árvores
  - ...

Pilhas

Definição

- Uma **pilha** é uma coleção ordenada de zero ou mais itens, de um mesmo tipo ou não, tal que suas operações principais são realizadas na mesma extremidade, denominada topo.

Pilhas

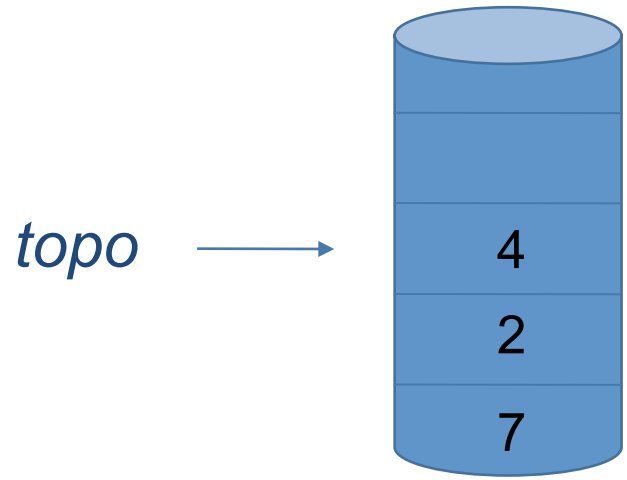
Definição

- As operações básicas pilhas são:
  - **Empilhar**: insere um novo item no **topo** da estrutura.
  - **Desempilhar**: remove o item do **topo** da estrutura

Pilhas

Definição

- As operações Empinhar/Desempilhar seguem a ordem: o último item a entrar no conjunto é o primeiro item a sair.
  - *Last in / First out* (LIFO)

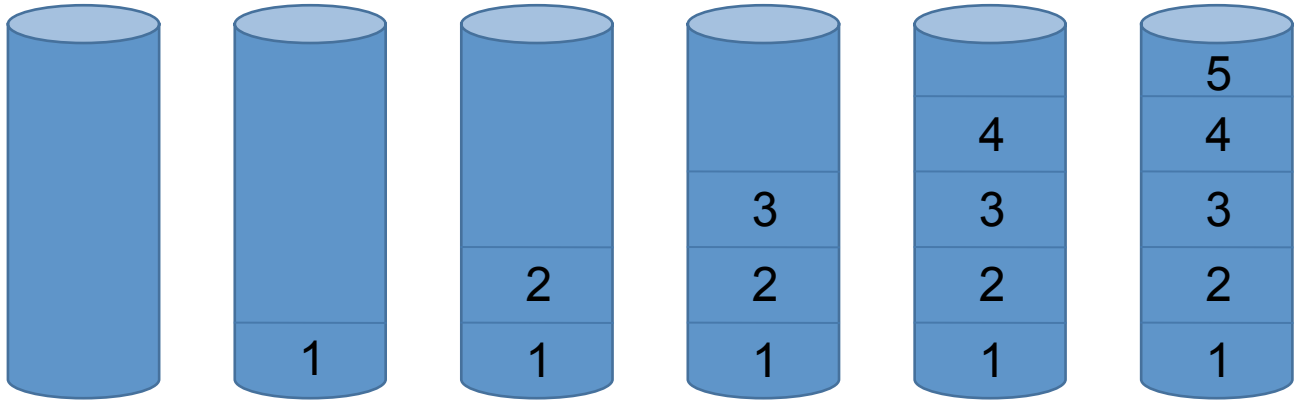


Pilhas

Comportamento

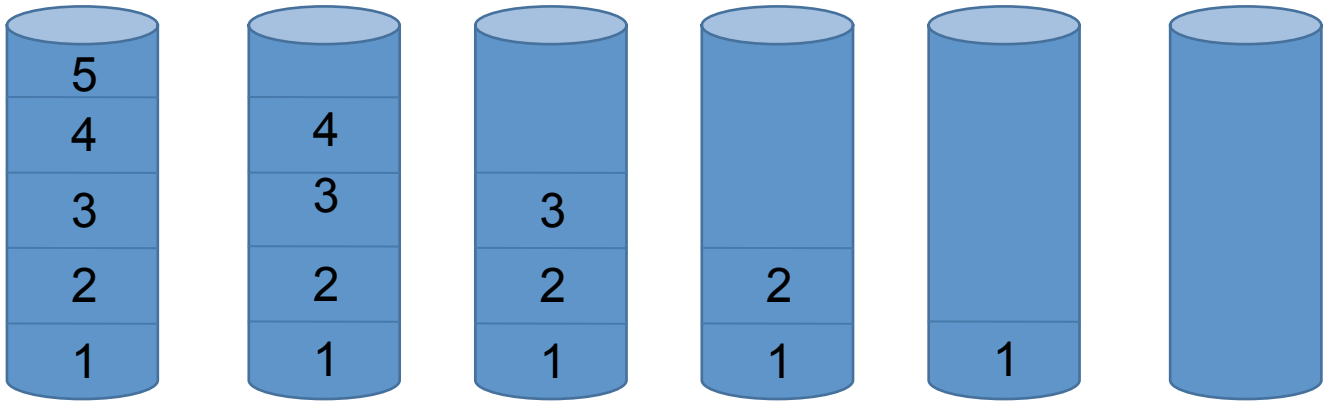
Empilhar:

1	2	3	4	5



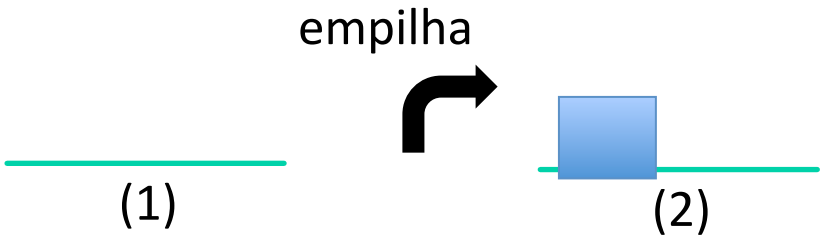
Desempilhar:

5	4	3	2	1



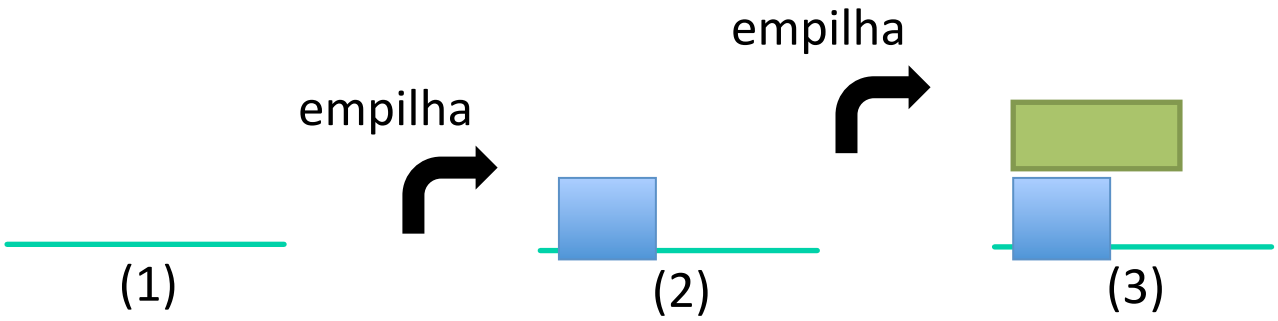
Pilhas

Comportamento



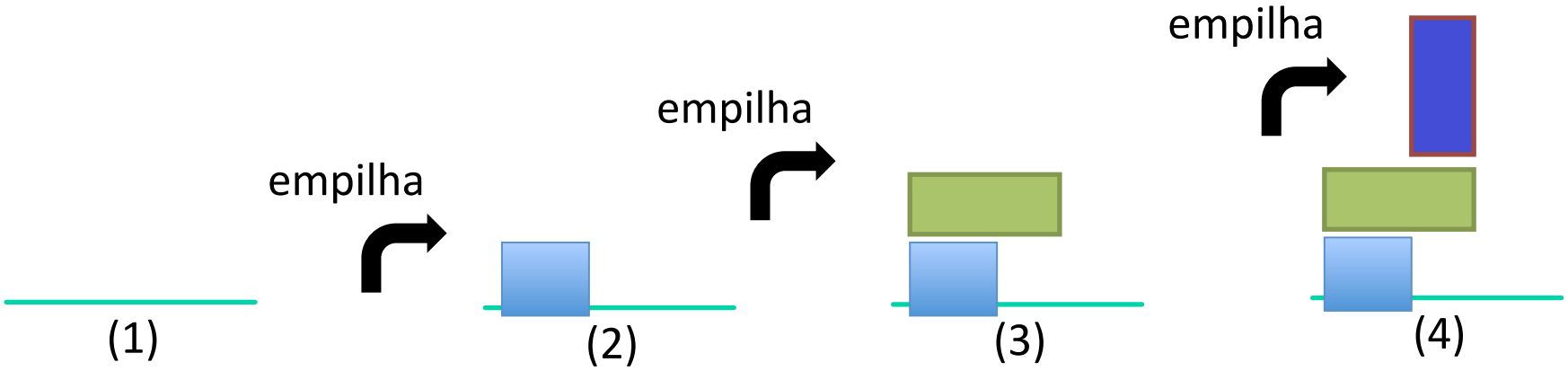
Pilhas

Comportamento



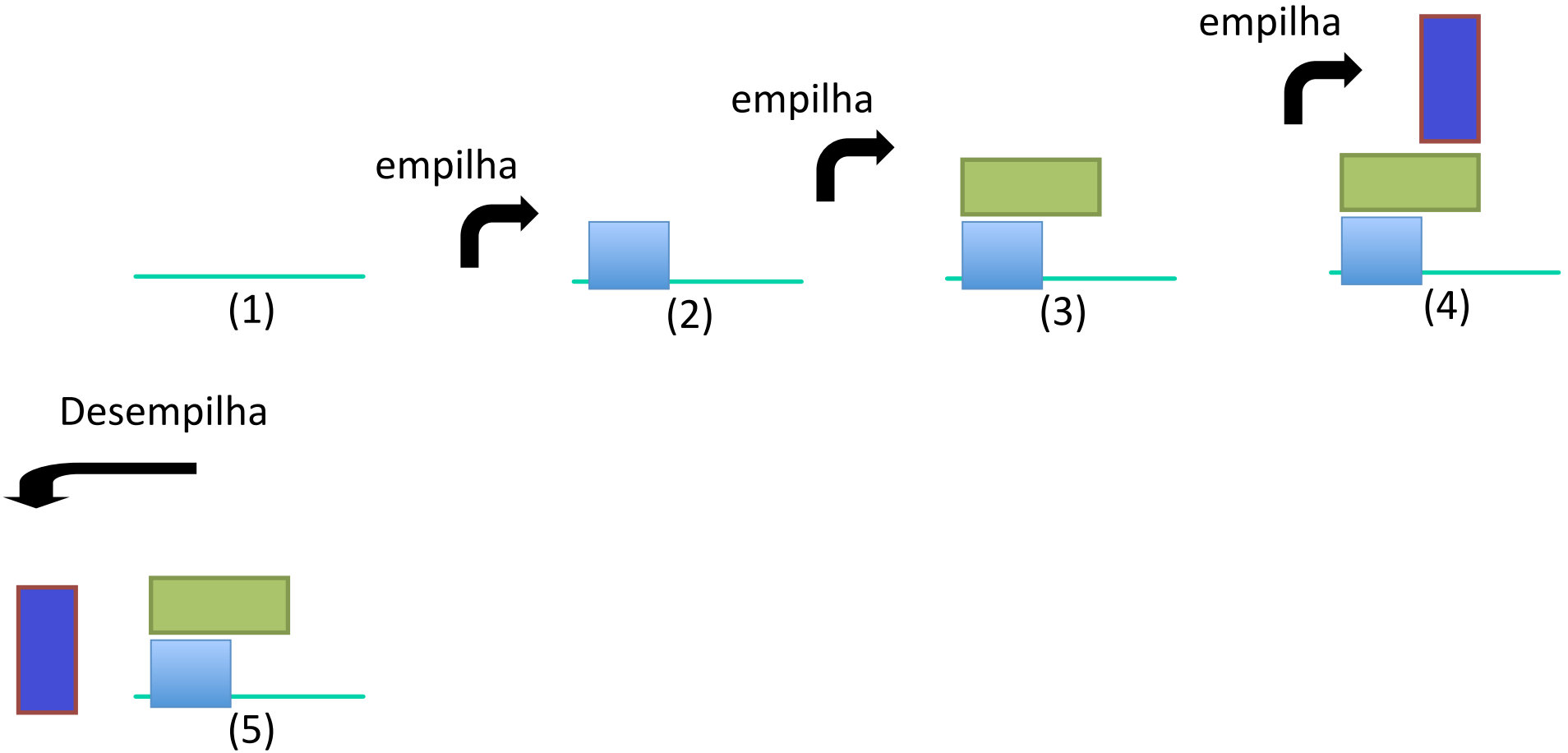
Pilhas

Comportamento



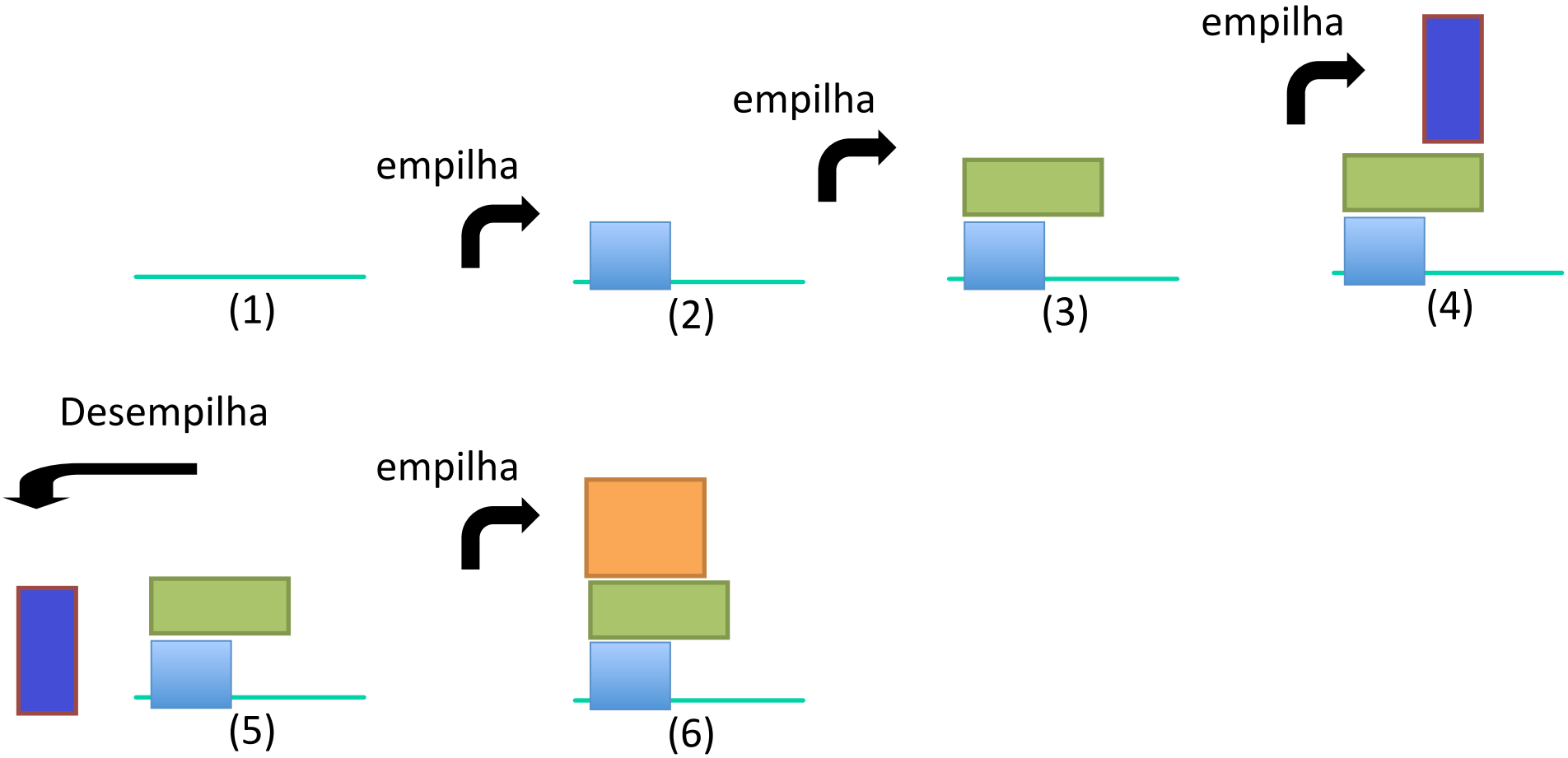
Pilhas

Comportamento



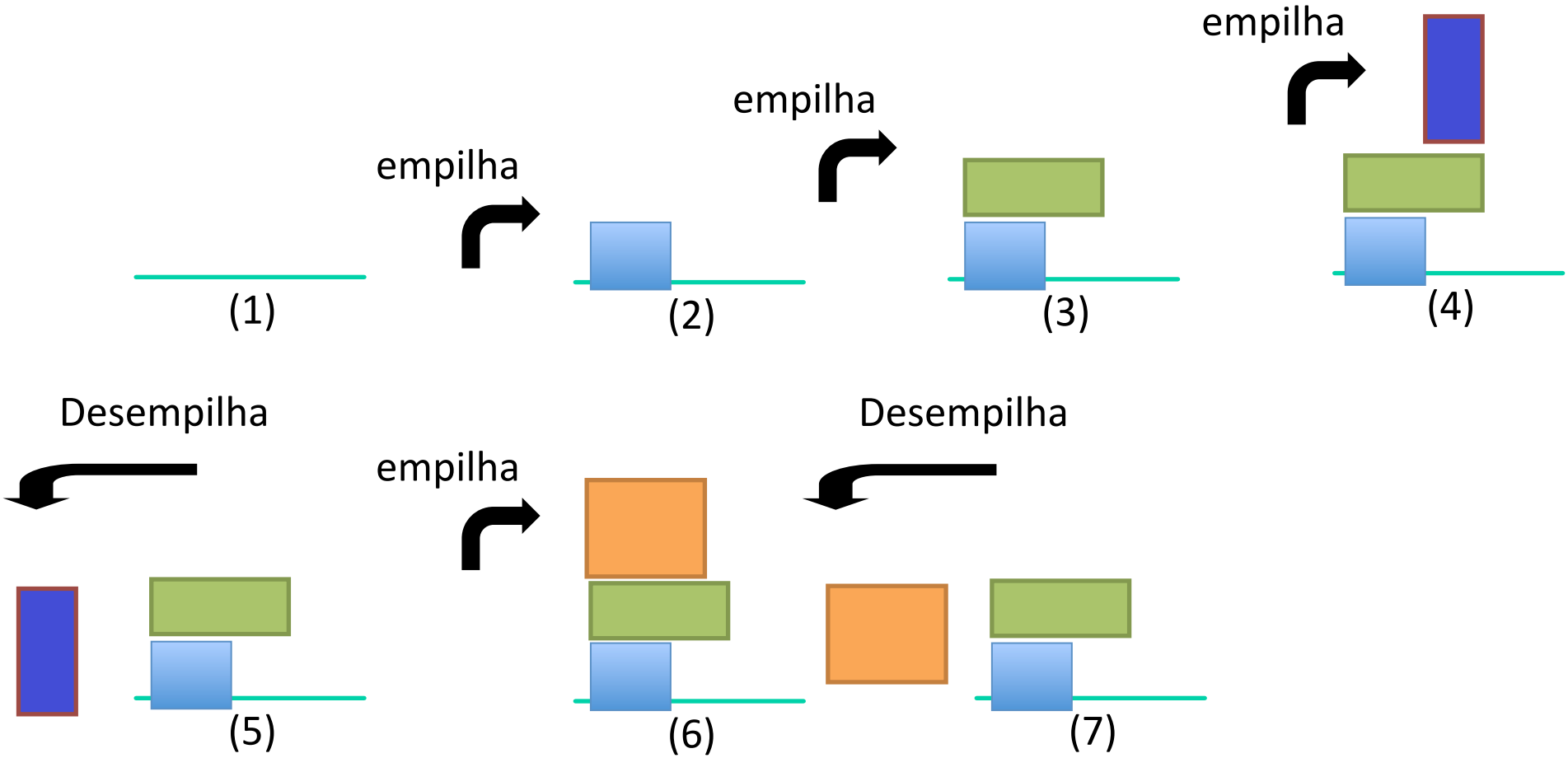
Pilhas

Comportamento



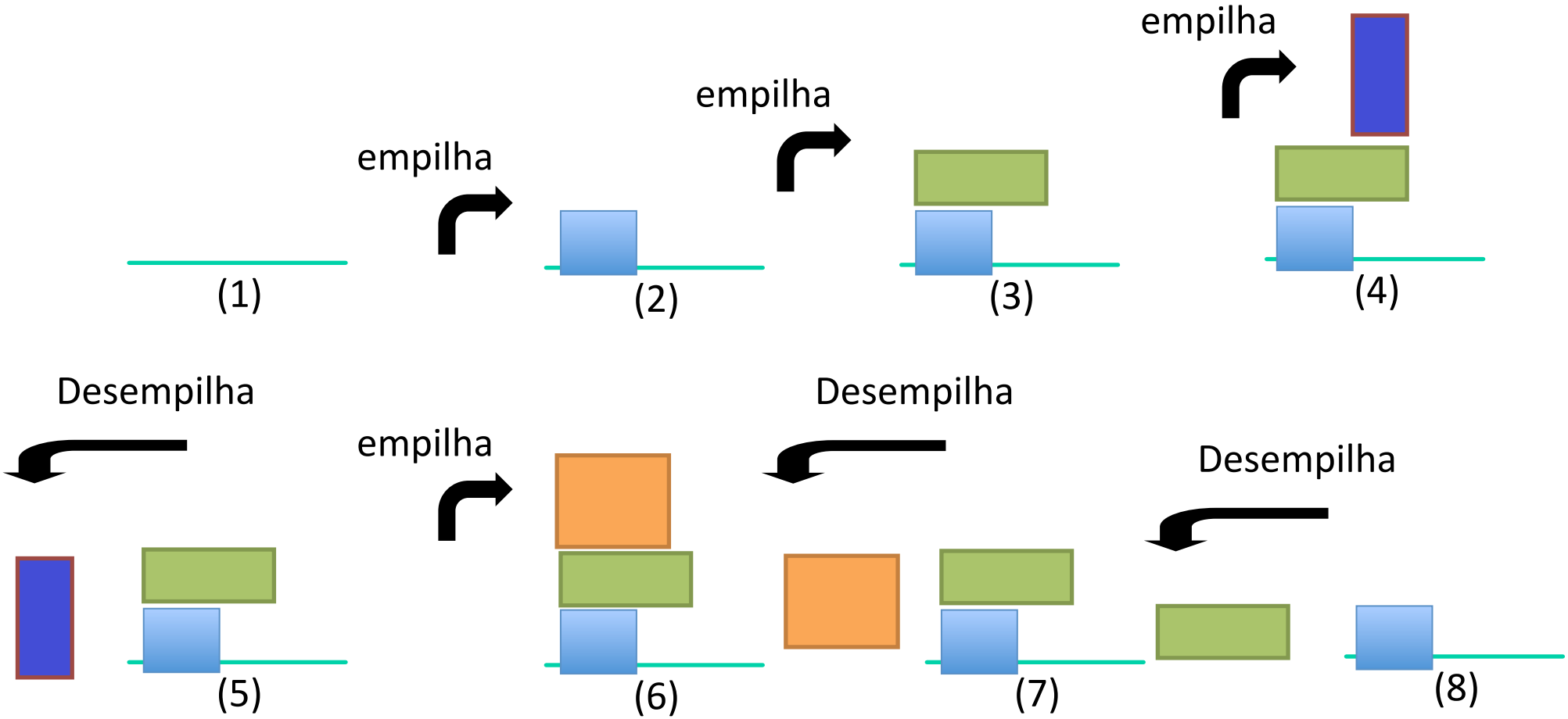
Pilhas

Comportamento



Pilhas

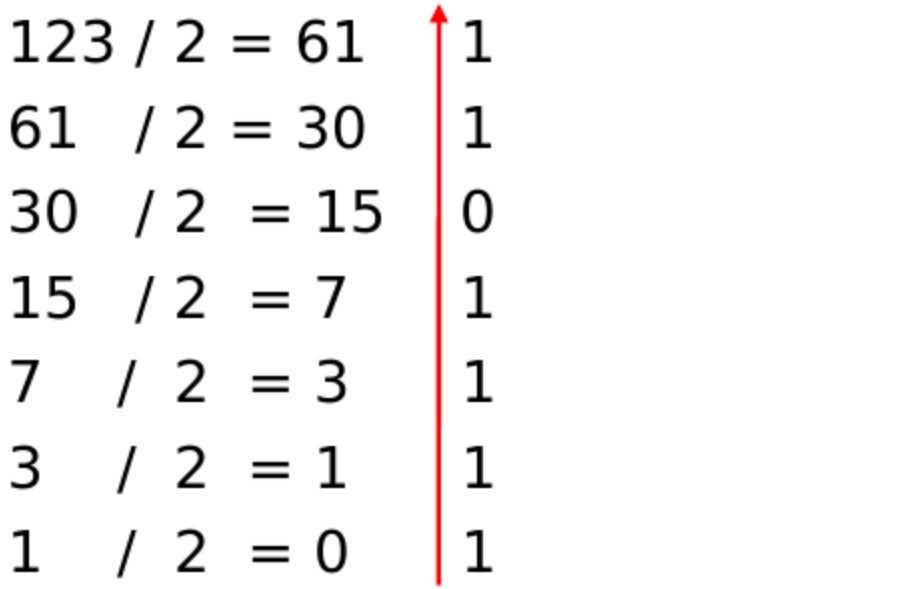
Comportamento



Pilhas

Exemplo: conversão de um número em decimal para binário

- Converta o número $123_{(10)}$ para binário
  - Método das divisões sucessivas.



Resultado: $123_{(10)} = 1111011_{(2)}$

Observação: Percebam o comportamento de **pilha**: o último resto a ser calculado é o primeiro resto utilizado na composição do número binário.

Pilhas

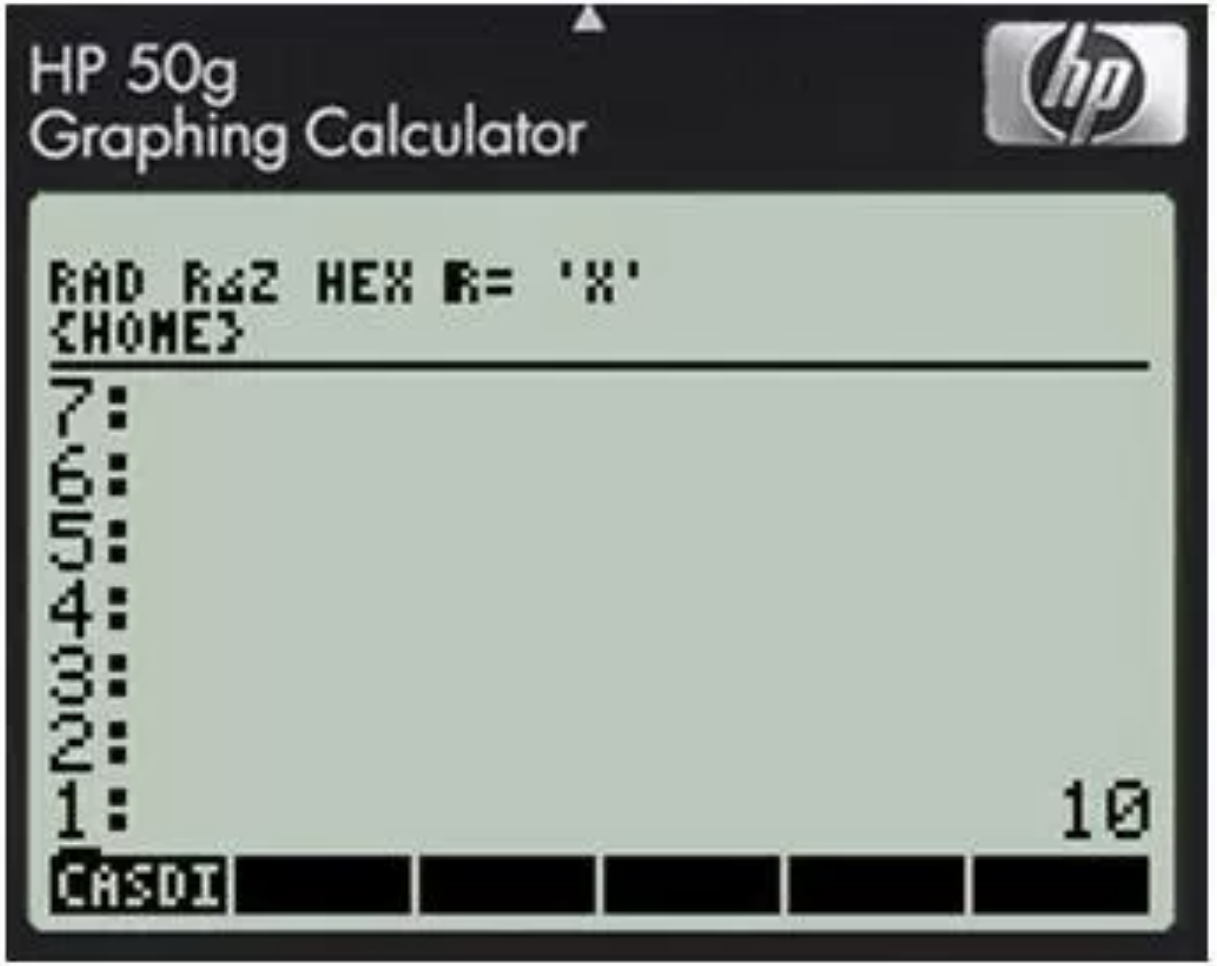
Exemplo: controle de chamada de funções

- Chamada de funções

Observação: Percebam o comportamento de **pilha**: a última função a ser chamada é a primeira função a ter a execução completada.

Pilhas

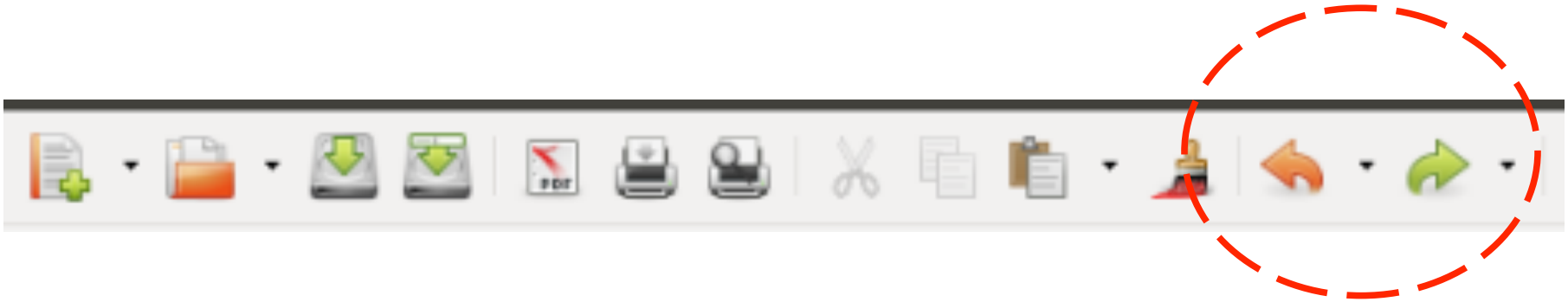
Exemplo: avaliação de expressão pré-fixa



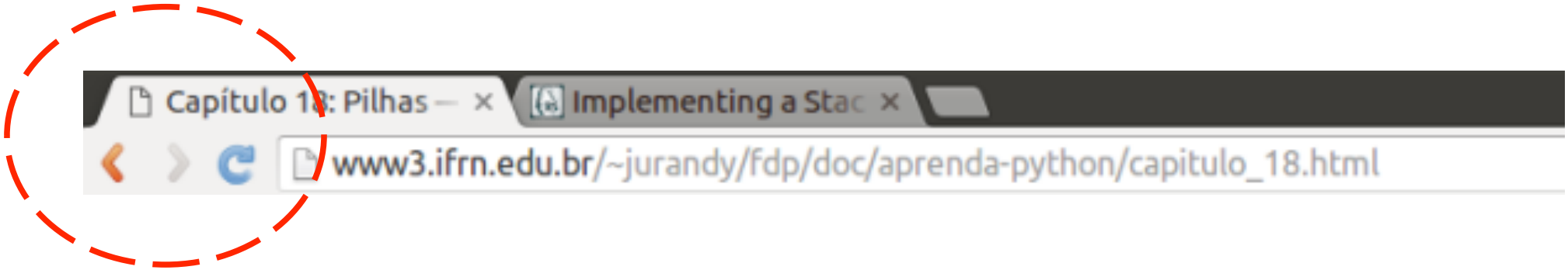
Pilhas

Exemplo: cotidiano

- Undo/Redo no editor de textos



- Ir para frente / ir para a trás no navegador



Pilhas

Como um conteito, uma abstração

- As pilhas são conceitos e ou abstrações.

Pilhas

Como um conteito, uma abstração

- As pilhas são conceitos e ou abstrações.
- As strings, listas, dicionários e tuplas são abstrações “nativas”do Python.

Pilhas

Como um conteito, uma abstração

- As pilhas são conceitos, abstrações e ou tipos de dados.
- As strings, listas, dicionários e tuplas são abstrações “nativas”do Python.
- As pilhas precisamos criar.

Pilhas

Como um conteito, uma abstração

- As pilhas são conceitos, abstrações e ou tipos de dados.
- As strings, listas, dicionários e tuplas são abstrações “nativas”do Python.
- As pilhas precisamos criar.
- Então, precisamos usar os elementos que a linguagem fornece para poder criar novas abstrações e ou tipos de dados.

Pilhas

Como um conteito, uma abstração

- As pilhas são conceitos, abstrações e ou tipos de dados.
- As strings, listas, dicionários e tuplas são abstrações “nativas”do Python.
- As pilhas precisamos criar.
- Então, precisamos usar os elementos que a linguagem fornece para poder criar novas abstrações e ou tipos de dados.
- Em Python, são através de classes.

Criando novos tipos de dados

Definindo classes

- Em Python, classes possuem:
  - Um identificador
  - Um conjunto de atributos (características, estado das suas **instâncias** ou **objetos**)
  - Um conjunto de métodos (operações)

- Em Python, a criação de classes é realizada utilizando a palavra reservada *class*.

```
Class Cachorros:
    pass
```

Criando novos tipos de dados

Definindo atributos

- Atributos são características que estarão presentes em todos os objetos **instanciados** a partir da classe.

```
Class Cachorros:
  cobertura = 'pelos'
  alimento = 'carne'
  patas = 4
  habitat = 'domestico'
  nome = 'Rex'
```

```
Class Galinhas:
  cobertura = 'penas'
  alimento = 'graos'
  patas = 2
  habitat = 'domestico'
  bico = 'pequeno'
```

Criando novos tipos de dados

Definindo atributos

- Atributos são características estarão presentes em todos os objetos **instanciados** a partir da classe.

```
Class Cachorros:
  cobertura = 'pelos'
  alimento = 'carne'
  patas = 4
  habitat = 'domestico'
  nome = 'Rex'
```

```
Class Galinhas:
  cobertura = 'penas'
  alimento = 'graos'
  patas = 2
  habitat = 'domestico'
  bico = 'pequeno'
```

```
Snoopy = Cachorros()
Lala = Galinhas()
print Snoopy.cobertura #sairá na tela 'pelos'
print Lala.alimento #sairá na tela 'graos'
Print Snoopy.bico #ERRO - não existe atributo bico
```

Criando novos tipos de dados

Definindo métodos

- A classe Cachorro tem um problema: todos os objetos do tipo Cachorro têm o mesmo nome “Rex”
  - No mundo real, cada cachorro possui um nome.

```
Class Cachorros:
   cobertura = 'pelos'
   alimento = 'carne'
   patas = 4
   habitat = 'domestico'
   nome = 'Rex'
```

Criando novos tipos de dados

Definindo métodos

- Métodos são funções definidas dentro da classe.
- Eles definem ações que serão executadas em uma instância (objeto) da classe
- O primeiro método de uma classe, chamado no momento da instanciação dos objetos é o **Construtor**.

```
Class <nome da classe>:
  def __init__ (self):
     <commandos>
```

Criando novos tipos de dados

Definindo métodos

- O primeiro método de uma classe, chamado no momento da instanciação dos objetos é o **Construtor**.

```
Class Cachorros:
   cobertura = 'pelos'
   alimento = 'carne'
   patas = 4
   habitat = 'domestico'
   def __init__ (self, nome):
       self.nome = nome
```

```
d1 = Cachorros('Dog1')
print d1.nome
```

Voltando a Pilha ...

Pilhas

Atributos, estado

- No caso da pilha, o estado é definido por uma coleção que permite armazenar os elementos e removê-los em uma dada ordem.

Pilhas

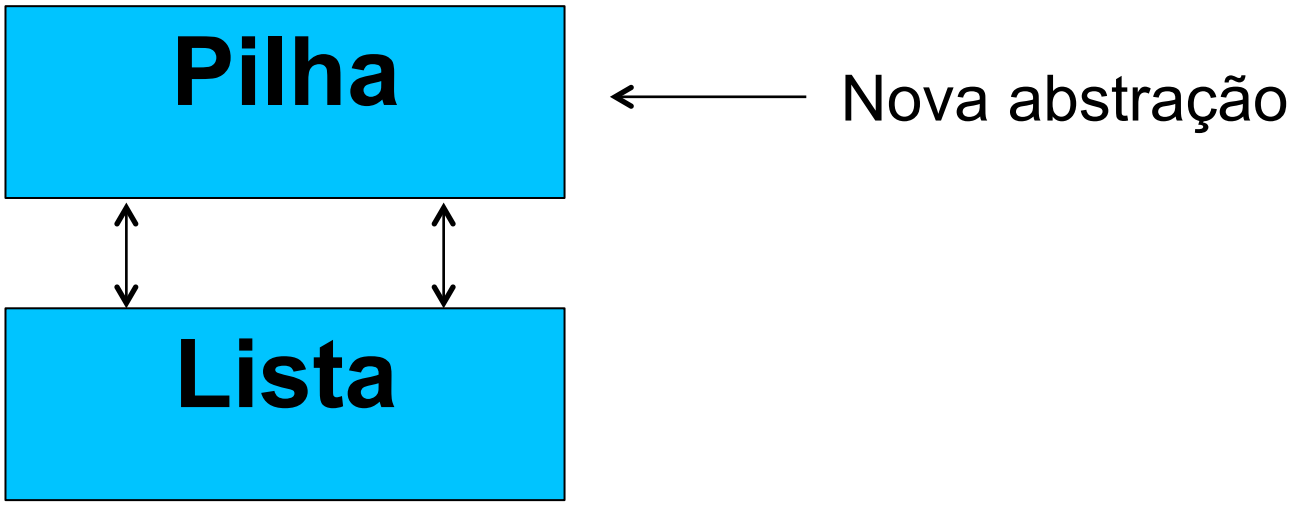
Atributos, estado

- No caso da pilha, o estado é definido por uma coleção que permite armazenar os elementos e removê-los em uma dada ordem.
- A lista nativa do Python tem essa flexibilidade.

Pilhas

Atributos, estado

- No caso da pilha, o estado é definido por uma coleção que permite armazenar os elementos e removê-los em uma dada ordem.
- A lista nativa do Python tem essa flexibilidade.



Pilhas

Operações: descrição

- As operações (métodos) comumente utilizadas em pilhas são:

  - **Empilhar**: insere um novo item no topo da estrutura.
  - **Desempilhar**: remove o item do topo da estrutura, retornando-o.
  - **Vazia**: verifica se a pilha está vazia.
  - **Topo**: retorna o item no topo da estrutura, sem removê-lo.
  - **Tamanho**: retorna a quantidade de item na estrutura.

Pilhas

Implementação por meio de listas do Python

```
1  class Pilha:
2      def __init__(self):
3          self.items = []
4
5      def Vazia(self):
6          return self.items == []
7
8      def Empilhar(self, item):
9          self.items.append(item)
10
11     def Desempilhar(self):
12         return self.items.pop()
13
14     def Topo(self):
15         return self.items[len(self.items)-1]
16
17     def Tamanho(self):
18         return len(self.items)
19
```

Pilhas

Implementação por meio de listas do Python

```
1  s=Pilha()
2
3  print(s.Vazia())
4  s.Empilhar(4)
5  s.Empilhar('dog')
6  print(s.Topo())
7  s.Desempilhar(    )
8  print(s.Tamanho())
9  print(s.Vazia())
10 s.Empilhar(8.4)
11 print(s.Desempilhar())
12 print(s.Desempilhar())
13 print(s.Tamanho())
14
```

Pilhas

Aplicação: verificação de parênteses

Pilhas podem ser utilizadas para verificar se os parênteses em uma expressão está balanceado.

Exemplo:

((3+4 + (4*9) **ERRO: Faltam dois parênteses fechando!**

Como?

Para cada carácter da expressão faça

1. Se encontrou um "(" empilha
2. Se encontrou um ")" então
    1. Se a pilha estiver vazia: expressão invalida
    2. Caso contrario desempilhe
3. Se no final pilha estiver vazia, expressão válida.

Pilhas

Aplicação: avaliação de expressões aritméticas

- Notação **Infixa:**
  - A+B*C
  - É ambigua
  - Necessidade de precedência de operadores ou do uso de parênteses
- Notação **Prefixa (Polonesa):**
  - -*AB/CD = (A*B)-(C/D)
  - Operadores precedem operandos
  - Dispensa o uso de parênteses
- Notação **Posfixa (Polonesa reversa):**
  - AB*CD/- = (A*B)-(C/D)
  - Operadores sucedem operandos
  - Dispensa o uso de parênteses

Pilhas

Aplicação: avaliação de expressões aritméticas

- Expressões na notação **Posfixa** podem ser avaliadas utilizando uma pilha.
  - A expressão é avaliada da esquerda para a direita.
  - Os operandos são empilhados.
  - Os operadores fazem com que dois operandos sejam desempilhados, o cálculo seja realizado e o resultado seja empilhado.

Pilhas

Aplicação: avaliação de expressões aritméticas

- Expressões na notação **Posfixa** podem ser avaliadas utilizando uma pilha.

  - Exemplo: 62/34*+3-

Símbolo	Ação	Pilha
6	empilhar	$P[6]$
2	empilhar	$P[2,6]$
/	desempilhar, aplicar operador e empilhar	$P[(6/2)] = P[3]$
3	empilhar	$P[3,3]$
4	empilhar	$P[4,3,3]$
*	desempilhar, aplicar operador e empilhar	$P[(3*4),3] = P[12,3]$
+	desempilhar, aplicar operador e empilhar	$P[(3+12)] = P[15]$
3	empilhar	$P[3,15]$
−	desempilhar, aplicar operador e empilhar	$P[(15-3)] = P[12]$
	final, resultado no topo da pilha	$P[12]$

Observação: Neste exemplo, o topo da pilha é representado pela posição mais à esquerda de P.

Pilha

Exercícios em laboratório

- Escreva uma função em Python que receba uma sequência de parênteses e colchetes e retorne True se ela está bem formada ou False, caso contrário.

- Escreva um programa em Python que leia uma expressão na notação Posfixa e realize sua avaliação.

Material complementar

http://www3.ifrn.edu.br/~jurandy/fdp/doc/aprenda-python/capitulo_18.html

http://interactivepython.org/runestone/static/pythonds/BasicDS/ImplementingaStackinPython.html

Dúvidas?