Introdução às Estruturas de Dados

Filas e Pilhas

Prof. Paulo Viniccius Vieira
paulo.vieira@faculdadeimpacta.com.br

Agenda

- Estruturas de Dados
- Fila
  - Definição de Fila
  - Operações de Fila
  - Exemplo de Fila
- Pilha
  - Definição de Pilha
  - Operações de Pilha
  - Exemplo de Pilha
- Exercícios de Implementação de Fila e Pilha

Estrutura de Dados

- **Estruturas de Dados** são utilizadas para armazenar e manipular um conjunto de dados.
- A Estruturas de Dados diferem umas das outras pelo **relacionamento** e **manipulação** de seus dados.
  - Exemplos de operações:
    - criação da Estrutura de Dado
    - inclusão de um novo elemento
    - remoção de um elemento
    - acesso a um elemento

Estrutura de Dados

- Diferentes tipos de Estrutura de Dados são adequadas a diferentes tipos de aplicações ou problemas

- Exemplos de Estruturas de Dados:
  - Listas
  - Tuplas
  - Dicionários
  - Filas
  - Pilhas
  - Árvores

Fila

- Uma ***fila*** *(queue)* define uma ***estrutura de dados*** que armazena uma sequência de dados onde o primeiro elemento inserido será o primeiro a ser removido
  - Estrutura do tipo **FIFO** - ***First In First Out***

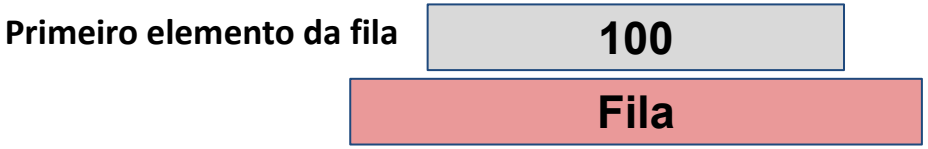
- A remoção é restrita ao primeiro elemento da sequência
- A inserção é restrita ao final da sequência

Fila

- Simulação:
  - Inicialmente, a fila está vazia.

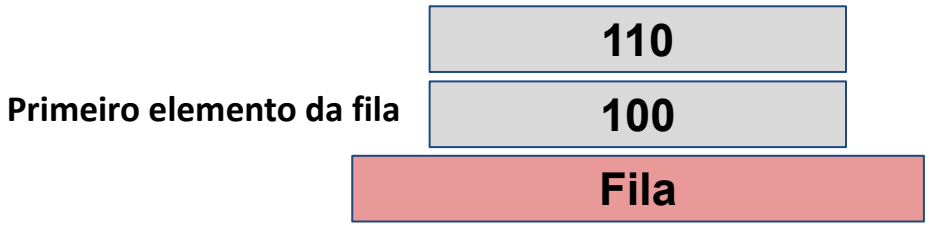
Fila

- Simulação:
  - Inicialmente, a fila está vazia.
  - Adicionar um elemento (100).



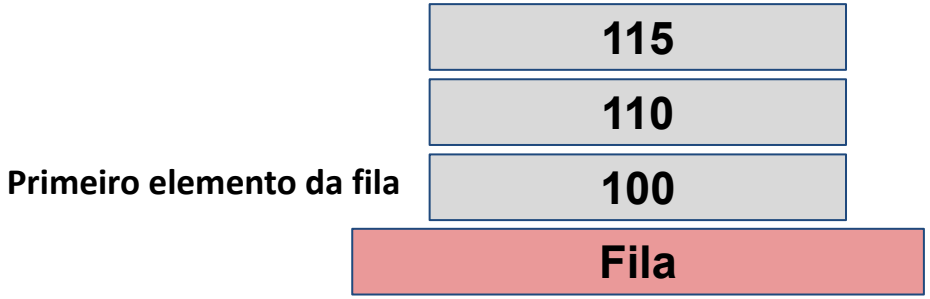
Fila

- Simulação:
  - Inicialmente, a fila está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).



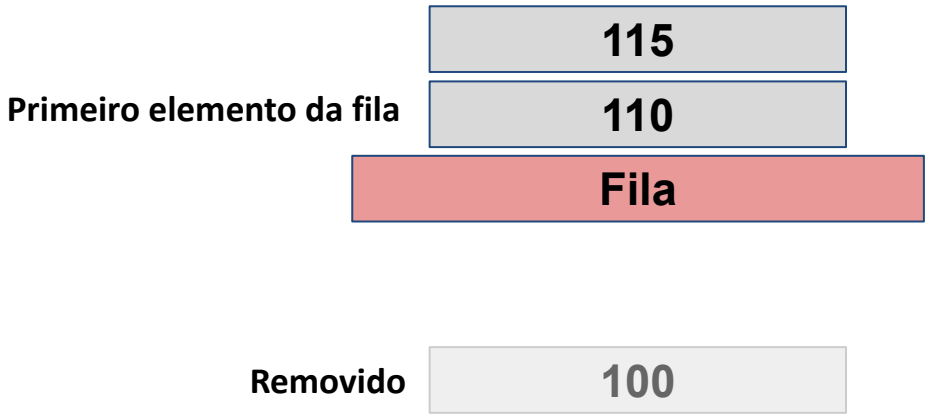
Fila

- Simulação:
  - Inicialmente, a fila está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).



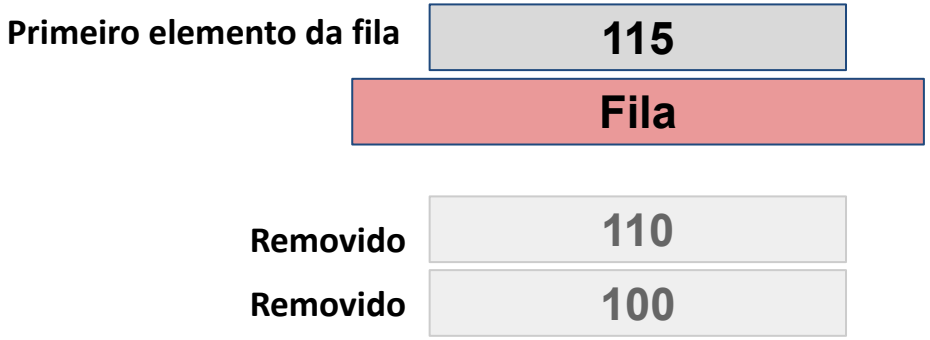
Fila

- Simulação:
  - Inicialmente, a fila está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).
  - Remover elemento.



Fila

- Simulação:
  - Inicialmente, a fila está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).
  - Remover elemento.
  - Remover elemento.



Fila

- Aplicações:
  - Filas são utilizadas em situações em que desejamos preservar a ordem de entrada dos elementos
    - Sistema de atendimento de clientes
    - Filas de banco
    - Filas de processos de software esperando para usar algum recurso
    - Filas de impressão

Fila

- Exemplo:
  - Sistema de atendimento de clientes por senha
    - um cliente solicita uma senha e espera para ser atendido;
    - a senha do cliente entra em uma fila de espera, seguindo a ordem dos clientes que solicitaram senha antes;
    - as senhas são chamadas até que a fila esteja vazia

Fila

- Uma fila suporta as seguintes operações:
  - **Enfileirar** (enqueue):
    - Insere um item no final da fila
  - **Desenfileirar** (dequeue):
    - Remove e retorna o item do início da fila
  - **Primeiro** (first):
    - Retorna mas não remove o item do início da fila
  - **Vazia** (is_empty):
    - Retorna um valor *booleano* informando se a fila está vazia
  - **Tamanho** (size):
    - Retorna a quantidade de itens na fila

Fila

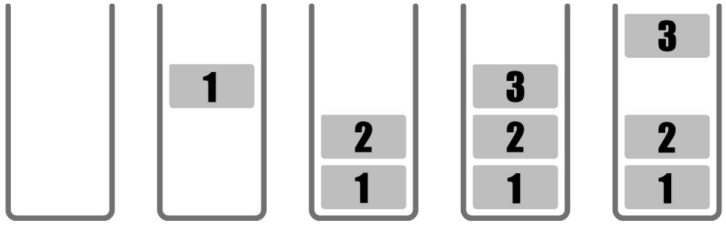
- A forma mais simples de implementar uma fila em *Python* é a partir de uma lista
  - Utiliza as funções:
    - **`append`** para enfileirar
    - **`pop(0)`** para desenfileirar

```
fila = []                  # []        cria uma fila vazia
fila.append(2)             # [2]       insere
fila.append(4)             # [2,4]     insere
fila.append(9)             # [2,4,9]   insere
item = fila.pop(0)         # [4,9]     remove e retorna o item
fila.append(5)             # [4,9,5]   insere
primeiro = fila[0]         # 4         retorna primeiro elemento
tamanho = len(fila)        # 3         retorna tamanho da fila
```

Pilha

- Uma ***pilha*** *(stack)* define uma ***estrutura de dados*** que armazena uma sequência de dados na qual os dados que foram inseridos primeiros serão os últimos a serem removidos
  - Estrutura do tipo **LIFO** - ***Last In First Out***

- A inserção e a remoção de itens ocorrem no final da pilha, chamada **topo**.
- A extremidade oposta é chamada de base.



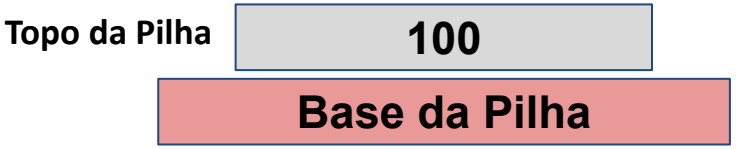
Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.

Base da Pilha

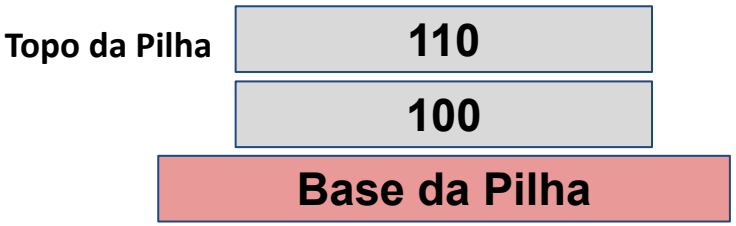
Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.
  - Adicionar um elemento (100).



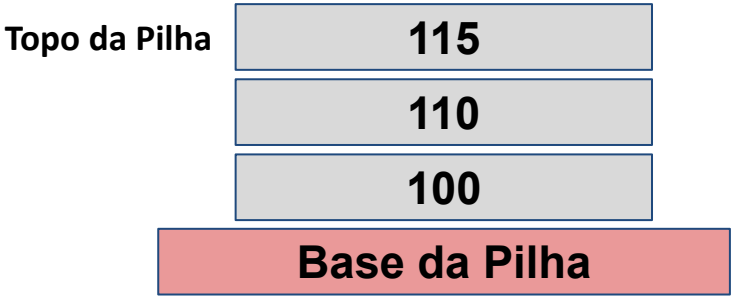
Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).



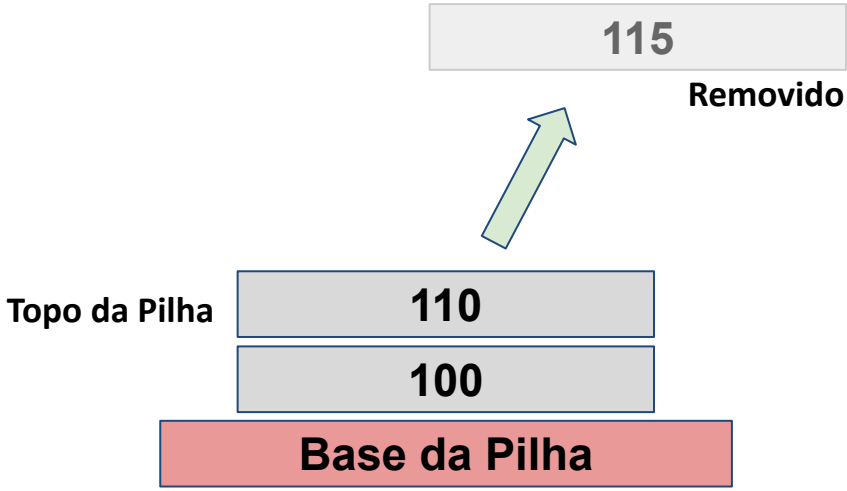
Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).



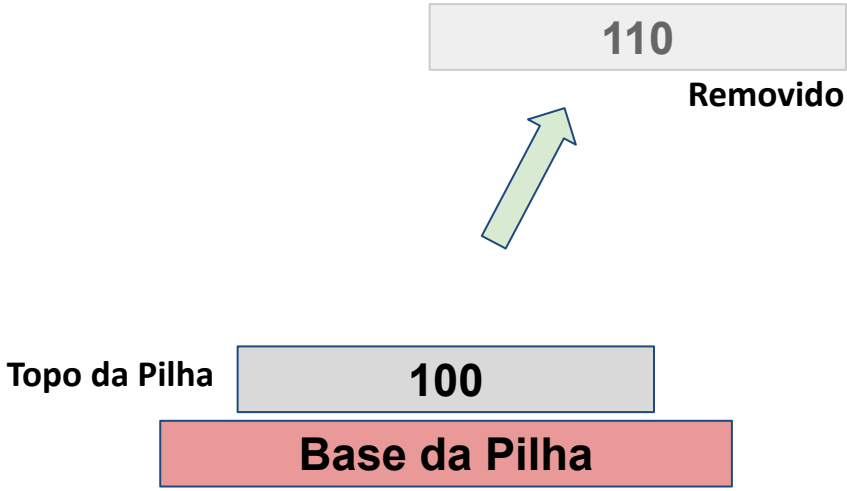
Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).
  - Remover elemento.



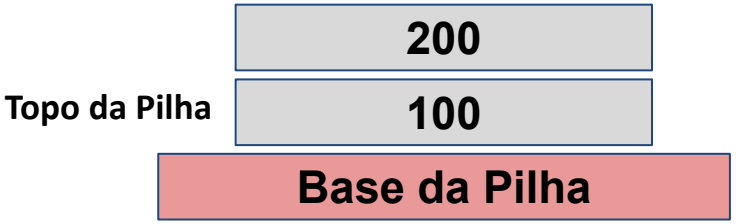
Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).
  - Remover elemento.
  - Remover elemento.



Pilha

- Simulação:
  - Inicialmente, a pilha está vazia.
  - Adicionar um elemento (100).
  - Adicionar um elemento (110).
  - Adicionar um elemento (115).
  - Remover elemento.
  - Remover elemento.
  - Adicionar um elemento (200).



Pilha

- Aplicações:
  - Pilhas são muito comuns em sistemas computacionais. Dentre as várias soluções possíveis que a pilha permite podemos citar:
    - Operações como desfazer e refazer em aplicações
    - Controle de navegação em browsers
    - Análise de expressões aritméticas

Pilha

- Uma pilha suporta as seguintes operações:
  - **Empilhar** (push):
    - Insere um item no topo da pilha
  - **Desempilhar** (pop):
    - Remove e retorna o item do topo da pilha
  - **Topo** (top):
    - Retorna mas não remove o item do topo da pilha
  - **Vazia** (is_empty):
    - Retorna um valor *booleano* informando se a pilha está vazia
  - **Tamanho** (size):
    - Retorna a quantidade de itens na pilha

Pilha

- A forma mais simples de implementar uma pilha em *Python* é a partir de uma lista. Considere que o primeiro item é a base.
  - **`append`** para empilhar
  - **`pop()`** para desempilhar

```
pilha = []                  #             cria uma pilha vazia
pilha.append(2)             # [2]         insere
pilha.append(4)             # [2,4]       insere
pilha.append(9)             # [2,4,9]     insere
pilha.append(7)             # [2,4,9,7]   insere
item = pilha.pop()          # [2,4,9]     remove e retorna o item
item = pilha.pop()          # [2,4]       remove e retorna o item
pilha.append(5)             # [2,4,5]     insere
topo = pilha[-1]            # 5           retorna o item do topo
tamanho = len(pilha)        # 3           retorna tamanho da pilha
```