

SSC0304 - Introdução à Programação para Engenharias

# Listas e Vetores

Prof.: Leonardo Tórtoro Pereira

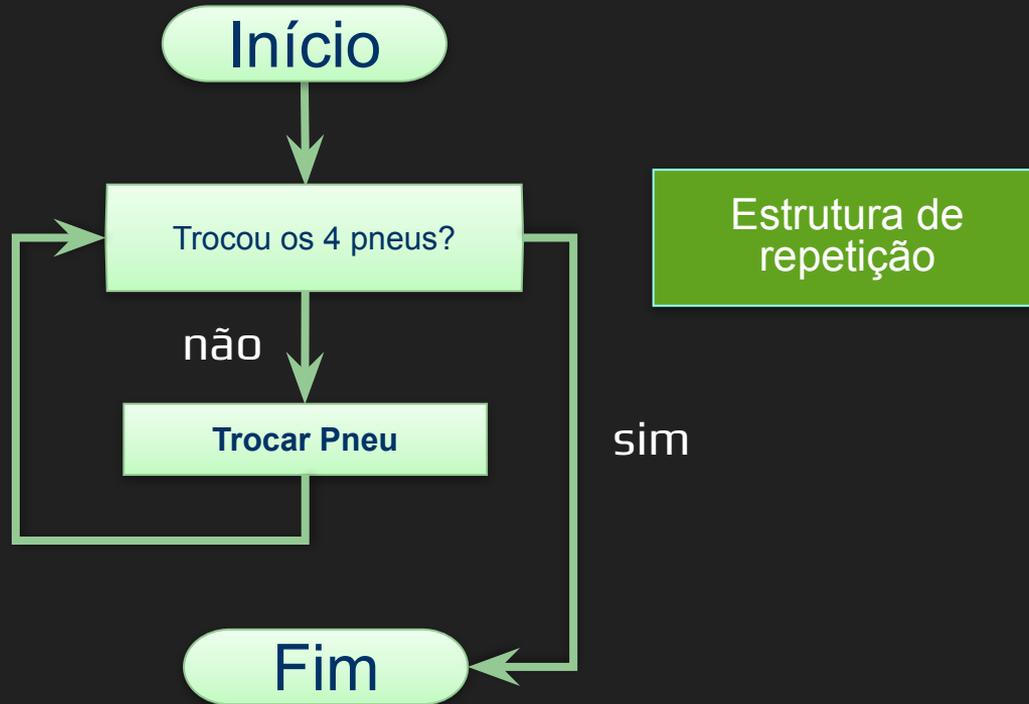
[leonardop@usp.br](mailto:leonardop@usp.br)

Baseado no material dos profs Fernando S. Osório e Claudio F.M. Toledo

Na aula passada...



# Algoritmo para trocar um pneu



# Loops

## Entry Controlled

for

```
for( initialization ; condition; updation)
{
}
```

while

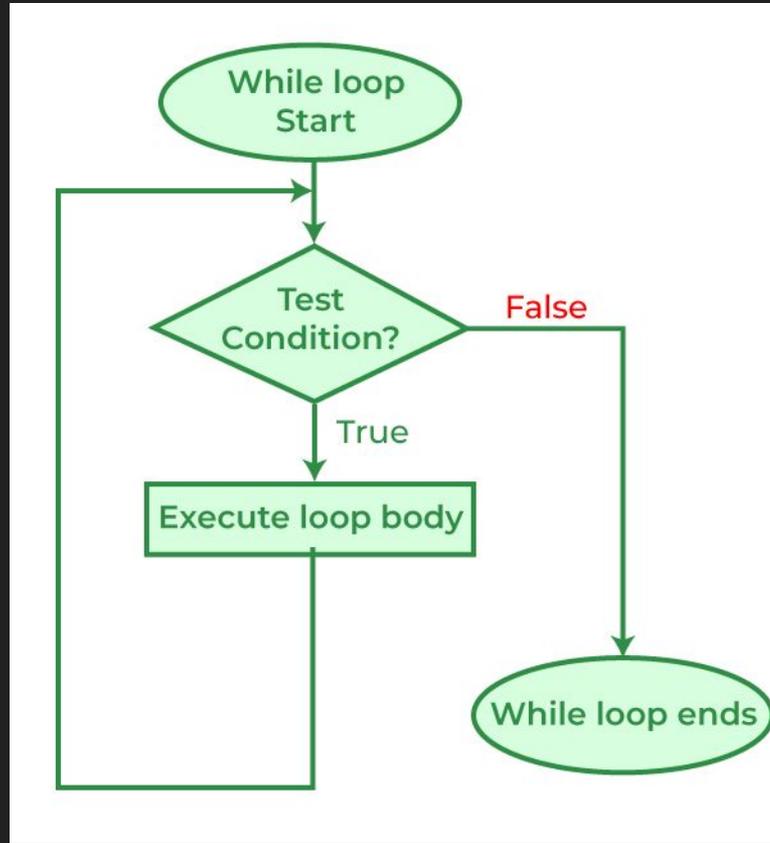
```
while( condition )
{
}
```

## Exit Controlled

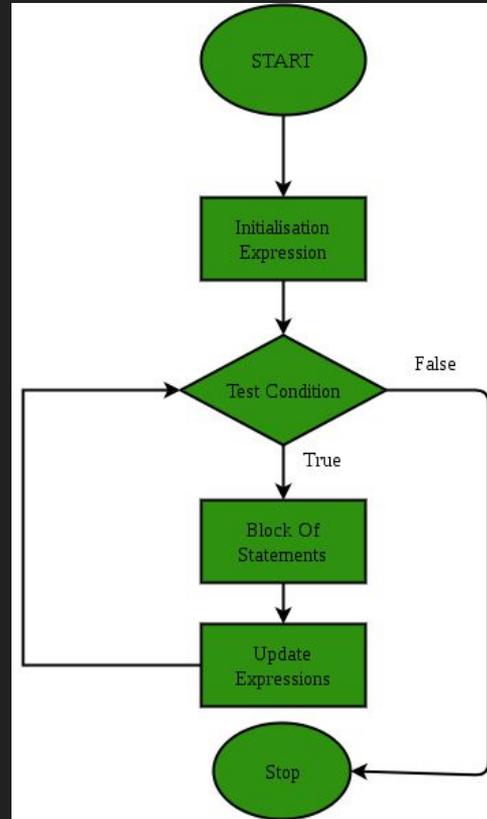
do-while

```
do
{
}while( condition )
```

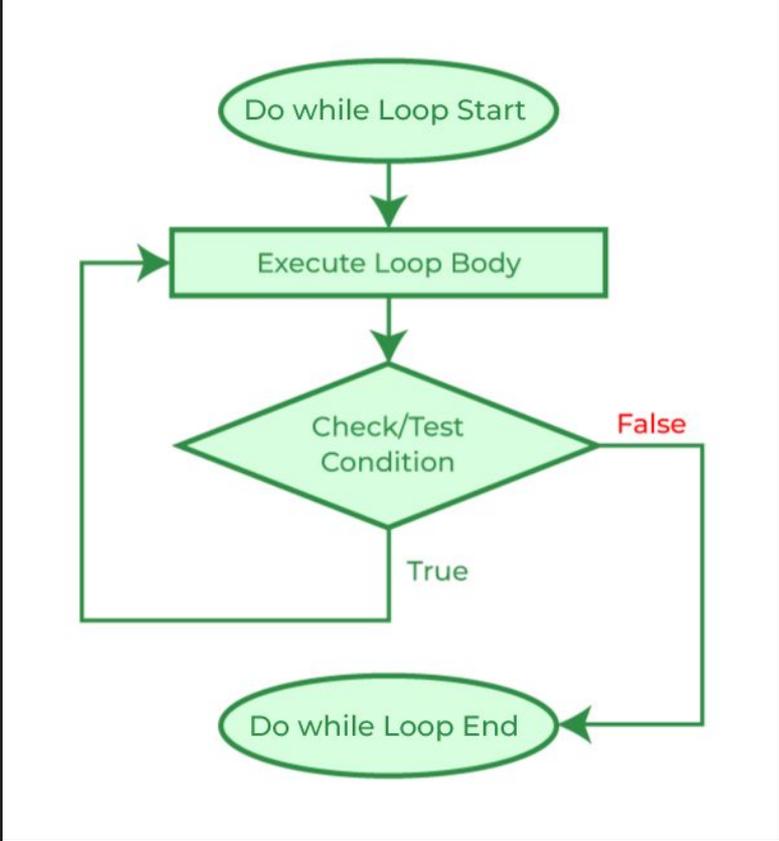




Fonte: <https://www.geeksforgeeks.org/cpp-loops/>



Fonte: <https://www.geeksforgeeks.org/loops-in-c-and-cpp/>



Fonte: <https://www.geeksforgeeks.org/loops-in-c-and-cpp/>

O que vamos aprender hoje?



# Objetivos

- Conhecer algumas das coleções de dados de python
- Conhecer Arrays (vetores)
- Observar as maiores diferenças entre vetores e listas
- Conhecer listas e suas aplicações como filas e pilhas

# Tópicos da Aula

- Coleções de Dados
- Arrays
- Diferenças entre Arrays e Listas
- Listas
- Métodos de listas
- Pilhas
- Filas

# Coleções de Dados

# Coleções de Dados

- Listas, Tuplas, Conjuntos, e Dicionários são heterogêneos
  - ◆ Aceitam vários tipos de dados
  - ◆ Menor custo para inserir e deletar
  - ◆ Maior custo para indexar
  - ◆ Maior gasto de memória

# Coleções de Dados

- Python tem 4 tipos *built-in* para coleção de dados
  - ◆ List
  - ◆ Tuple
  - ◆ Set
  - ◆ Dictionary
- String é um array (vetor)
- É possível criar outros Arrays com a biblioteca Array

# Coleções de Dados

- Strings e Arrays são homogêneos
  - ◆ Aceitam um único tipo de dado
  - ◆ Maior custo para inserir e deletar
  - ◆ Menor custo para indexar
  - ◆ Menor gasto de memória

# Exemplos

# Arrays

# Arrays

- Coleção de itens armazenado em espaços sequenciais de memória
  - ◆ Podem ser acessado *aleatoriamente* com o índice do vetor
- Usados para armazenar tipos de elementos similares
  - ◆ Todos os elementos precisam ser do mesmo tipo de dado
- Seu tamanho geralmente é *fixo!*

# Arrays

→ Para criar Arrays, é preciso usar a biblioteca Array

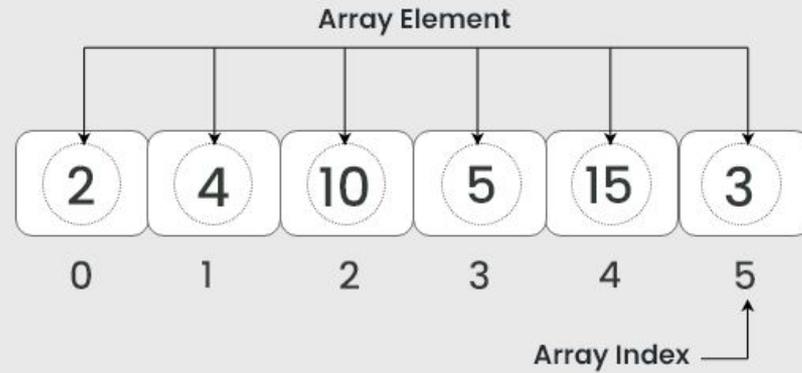
```
import array as arr

a = arr.array('i', [1, 2, 3])
print(type(a))
for i in a:
    print(i, end=" ")
```



# Array

## Data Structure



Fonte: <https://www.geeksforgeeks.org/array-data-structure/>

## Memory Location

200	201	202	203	204	205	206	■	■	■
U	B	F	D	A	E	C	■	■	■
0	1	2	3	4	5	6	■	■	■

Index

Fonte: <https://www.geeksforgeeks.org/what-is-array/>

# Tipos permitidos para um Array

Type code	C Type	Python Type	Minimum size in bytes
'b'	signed char	int	1
'B'	unsigned char	int	1
'u'	wchar_t	Unicode character	2
'h'	signed short	int	2
'H'	unsigned short	int	2
'i'	signed int	int	2
'I'	unsigned int	int	2
'l'	signed long	int	4
'L'	unsigned long	int	4
'q'	signed long long	int	8
'Q'	unsigned long long	int	8
'f'	float	float	4
'd'	double	float	8

# Exemplos

# Diferenças entre Arrays e Listas

Lista	Array (Vetor)
Elementos podem ser de tipos diferentes	Elementos precisam ser do mesmo tipo
Não precisa importar módulo explicitamente	Precisa importar módulo explicitamente
Não suporta diretamente operações aritméticas	Suporta operações aritméticas (numpy)
Melhor para sequência pequena de dados	Melhor para sequência maior de dados
Maior flexibilidade para modificar dados	Menor flexibilidade (add e del por elemento)
Acesso a elementos não precisa de loop	Acesso a elementos precisa de loop
Alto consumo de memória	Consumo mais baixo de memória
Lista aninhada pode ter tamanho variável	Array aninhado deve ter mesmo tamanho
Muitas funções prontas	Precisa importar módulos para operações

# Listas

# Listas

- Listas possuem vários métodos
- Podem ser usadas como pilhas e filas
  - ◆ Estruturas de dados interessantes
  - ◆ Pilha - First In Last Out (FILO)
  - ◆ Filas - First In First Out (FIFO)
- Podem ser percorridos via compreensão de listas

## Conceito de Lista Linear

- Uma Lista Linear é uma sequência de componentes de um mesmo tipo
- É uma sequência de zero ou mais itens  $x_1, x_2, \dots, x_n$  na qual  $x_i$  é de um determinado tipo e  $n$  representa o tamanho da lista

# Conceito de Lista Linear

- Sua principal propriedade estrutural diz respeito às posições relativas dos itens
- ◆ Se  $n \geq 1$ ,  $x_1$  é o primeiro item e  $x_n$  é o último (considerando que a indexação inicia a partir de 1)
  - ◆ Em geral,  $x_i$  precede  $x_{i+1}$  para  $i = 1, 2, \dots, n - 1$  e  $x_i$  sucede  $x_{i-1}$  para  $i = 2, 3, \dots, n$

# Operações

- As operações mais frequentes em listas são a busca, a inclusão e a remoção de um determinado elemento
- ◆ Outras operações: alteração de um elemento da lista, combinação de duas ou mais listas lineares em uma única, ordenação dos nós segundo um determinado campo, determinação do primeiro ou último nó da lista, etc.

# Aplicação

- Diversos tipos de aplicações requerem uma lista
- ◆ Informações sobre funcionários de uma empresa
  - ◆ Notas de compras
  - ◆ Itens de estoque
  - ◆ Notas de alunos
  - ◆ Lista telefônica

# Aplicação

→ Diversos tipos de aplicações requerem uma lista

- ◆ Lista de tarefas
- ◆ Gerenciamento de memória
- ◆ Simulações em geral
- ◆ Compiladores, etc.

## Casos particulares de listas

- **Deque (Double Ended QUEUE):** se as inserções e remoções são permitidas apenas nas extremidades da lista
- **Pilha (stack):** se as inserções e remoções são realizadas somente em um extremo
- **Fila (queue):** se as inserções são realizadas em um extremo e remoções em outro

# Métodos

## Métodos de Listas

- Adicionar itens ao final ou em uma posição específica
- Remover itens por valor ou por posição
- Remover todos os itens
- Retornar índice de item por valor
- Contar elementos
- Ordenar
- Reverter
- Copiar

# Exemplos

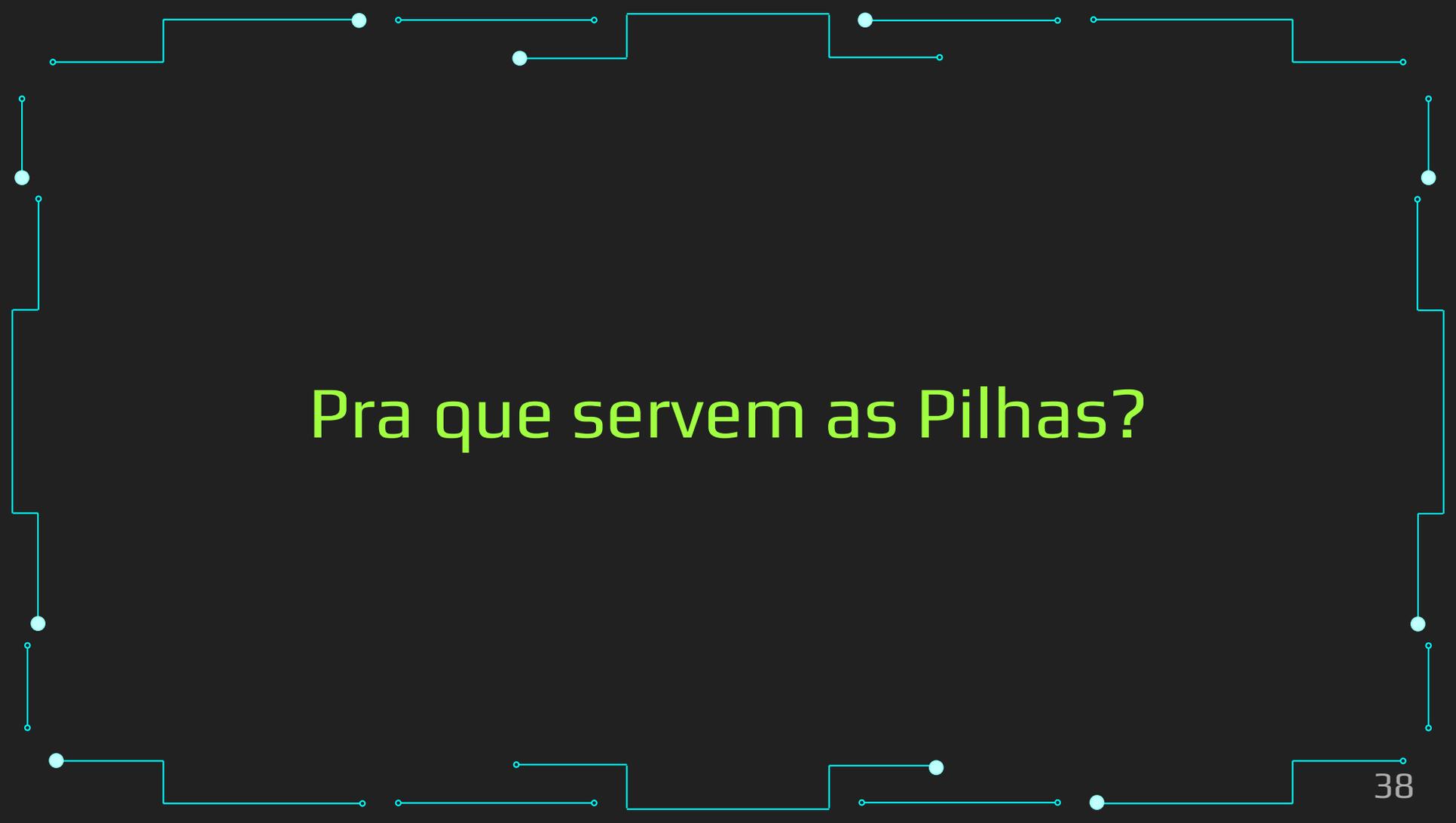
# Pilhas

# O quê são?



# Exemplos do mundo real



A decorative border in cyan color with white dots at the corners and midpoints of the sides, framing the central text.

Pra que servem as Pilhas?

## Para que servem?

- Auxiliam em problemas práticos em computação
  - ◆ O botão “back” de um navegador web ou a opção “undo” de um editor de textos
  - ◆ Controle de chamada de procedimentos
  - ◆ Estrutura de dados auxiliar em alguns algoritmos como a busca em profundidade

# Pilhas

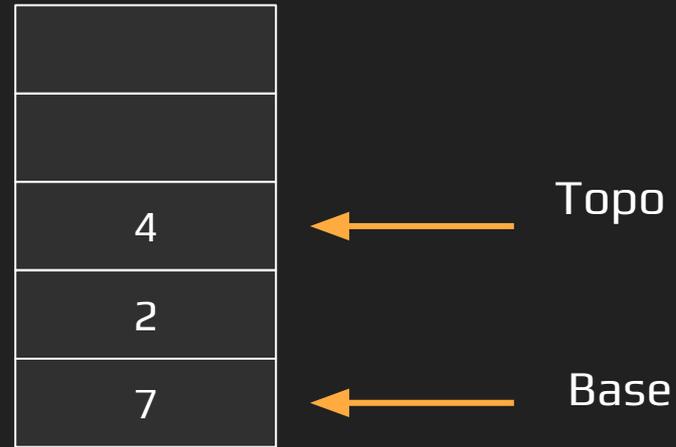
- Também conhecida como *stack*
- Estrutura de dados nas quais o último elemento inserido é o primeiro a ser retirado
  - ◆ Só é permitido acesso a 1 único item por vez
    - O último que foi inserido

# Pilhas

- Pilha é construída a partir da base até o topo
- ◆ Topo = nome dado ao último elemento inserido
- Qualquer operação com a pilha ocorre no topo

# Pilhas

- Por seu tipo de acesso, é uma estrutura LIFO
- ◆ Last-In First-Out



# Pilhas

- Usando as listas de python:
  - ◆ Adicionar apenas com `append()`
  - ◆ Remover apenas com `pop()`

# Exemplo

# Filas

# Exemplo de Aplicação

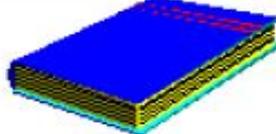
- Problema: automação de uma biblioteca
  - ◆ Todos os livros devem ser cadastrados
  - ◆ O sistema deve informar se um livro está disponível ou não nas estantes
  - ◆ Caso o livro não esteja disponível, o usuário pode aguardar em uma fila de espera
  - ◆ Quando o livro for devolvido, o primeiro da fila de espera pode retirá-lo
- Sua tarefa: desenvolver esse sistema

# Modelagem do Problema

→ 1º passo: abstração

- ◆ Identificar os elementos do mundo real que são relevantes para a solução do problema
  - Quais são eles?

# Modelagem do Problema

fila de espera para o livro	livros do acervo	disponível?
 <p>último ---&gt; &lt;--- 1º</p>	 <p>trigonometria</p>	não
 <p>último ---&gt; &lt;--- 1º</p>	 <p>química inorgânica</p>	não
fila vazia!	 <p>estruturas de dados</p>	sim

# Modelagem do Problema

→ Elementos relevantes

- ◆ Um cadastro de livros
- ◆ Indicação da disponibilidade dos livros
- ◆ Uma fila de espera para cada livro, com indicação da ordem das pessoas
- ◆ Primeiro e último da fila
- ◆ Cadastro de pessoas: nome, endereço e telefone

# Modelagem do Problema

→ 2º passo: quais são as operações possíveis nas filas?

- ◆ Entrar na fila
  - Quem entra, entra onde?
- ◆ Sair da fila
  - Quem sai, sai de onde?
- ◆ Outras?

# Fila (Queue)

→ O que é?

- ◆ *É uma estrutura para armazenar um conjunto de elementos, que funciona da seguinte forma*
  - Novos elementos sempre entram no fim da fila
  - O único elemento que se pode retirar da fila em um dado momento é seu primeiro elemento

# Fila (Queue)

→ Para que serve?

- ◆ Modelar situações em que é preciso armazenar um conjunto ordenado de elementos, no qual o primeiro elemento a entrar no conjunto será também o primeiro elemento a sair do conjunto, e assim por diante
  - Política FIFO: first in, first out

# Aplicações

→ Exemplos de aplicações de filas

- ◆ Filas de espera e algoritmos de simulação
- ◆ Controle por parte do sistema operacional de recursos compartilhados, tais como impressoras
- ◆ Buffers de Entrada/Saída
- ◆ Estrutura de dados auxiliar em alguns algoritmos como a busca em largura

# Fila

- Em Python, apesar de ineficiente, pode-se usar lista como fila
  - ◆ Todos os elementos precisam ser deslocados
  - ◆ Append + pop ao começo da lista
- Pode-se usar deque do módulo *collections*

# Exemplo

# Referências

# Referências

1. <https://www.learnpython.org/>
2. <https://www.w3schools.com/python/>
3. <https://panda.ime.usp.br/cc110/static/cc110/index.html>
4. [https://www.youtube.com/playlist?list=PLcoJJSvnDgcKpOi\\_UeneTNTIV0igRQwcn](https://www.youtube.com/playlist?list=PLcoJJSvnDgcKpOi_UeneTNTIV0igRQwcn)
5. <https://docs.python.org/3/tutorial/datastructures.html>
6. <https://www.geeksforgeeks.org/python-lists/>
7. <https://www.geeksforgeeks.org/python-data-types/>
8. <https://www.geeksforgeeks.org/difference-between-list-and-array-in-python/>
9. [https://www.w3schools.com/python/python\\_strings.asp](https://www.w3schools.com/python/python_strings.asp)
10. <https://docs.python.org/3/library/array.html>
11. <https://numpy.org/doc/stable/user/quickstart.html>