# COLEÇÕES E LISTAS

Prof. Arthur Araruna ⟨ararunaufc@gmail.com⟩

QXD0010 – Estrutura de Dados – 2020.1

UFC – Campus Quixadá

# TAD Coleção

# TAD COLEÇÃO

Antes de falarmos do TAD *Lista*, vamos mencionar outro TAD muito importante, *Coleção*.

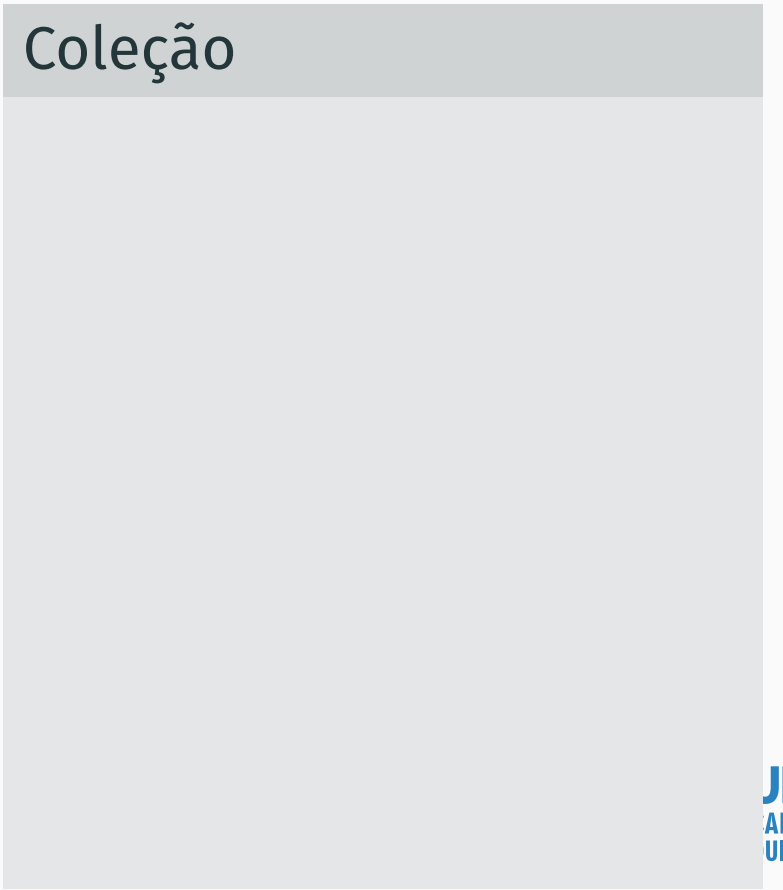
- Agrupamento de entidades, **elementos**, de forma lógica.
- Operações necessárias
  - Inserir elemento
  - Remover elemento
  - Verificar pertinência de elemento
  - Obter tamanho

UFC CAMPUS QUIXADÁ

# TAD COLEÇÃO

Antes de falarmos do TAD *Lista*, vamos mencionar outro TAD muito importante, *Coleção*.

- Agrupamento de entidades, **elementos**, de forma lógica.
- Operações necessárias
  - Inserir elemento
  - Remover elemento
  - Verificar pertinência de elemento
  - Obter tamanho



# TAD COLEÇÃO — OPERAÇÕES NECESSÁRIAS

## Inserir

Fornecemos um elemento.

Caso bem sucedido:

- O elemento fará parte da coleção
- O tamanho aumenta em 1 unidade

Caso mal sucedido: nenhuma propriedade é alterada.

UFC CAMPUS QUIXADÁ

# TAD COLEÇÃO — OPERAÇÕES NECESSÁRIAS

## Remover

Fornecemos um elemento.

Caso bem sucedido:

- O elemento deixará de fazer parte da coleção
- O tamanho diminui em 1 unidade

Caso mal sucedido: nenhuma propriedade é alterada.

Observações: Não se pode remover um elemento que não faz parte da coleção.

UFC CAMPUS QUIXADÁ

# TAD COLEÇÃO — OPERAÇÕES NECESSÁRIAS

## Testar

Fornecemos um elemento.

Resposta: Verdadeiro ou Falso (o elemento faz parte ou não da coleção)

## Tamanho

Resposta: a quantidade de elementos que fazem parte da coleção.

UFC CAMPUS QUIXADÁ

# TAD Lista

# TAD LISTA

- *Lista* é uma especialização de *Coleção*
  - portanto possui mesmas regras
- elementos logicamente dispostos em sequência
- a cada elemento corresponde um **índice**
  - corresponde à posição relativa do elemento
  - número natural
  - identificam unicamente um elemento

UFC CAMPUS QUIXADÁ

# TAD LISTA — VISUALIZAÇÃO LÓGICA

*Coleção*

*Lista*

UFC CAMPUS QUIXADÁ

# TAD LISTA

Essa noção de *índices* nos permite aumentar as operações

- Inserir elemento em um índice
- Remover elemento em um índice
- Obter elemento em um índice
- Obter índice de um elemento

UFC CAMPUS QUIXADÁ

# TAD LISTA — OPERAÇÕES NECESSÁRIAS

## Inserir por índice

Fornecemos um elemento e um índice.

Caso bem sucedido:

- O elemento fará parte da coleção e corresponderá ao índice
- O tamanho aumenta em 1 unidade

Caso mal sucedido: nenhuma propriedade é alterada.

UFC CAMPUS QUIXADÁ

# TAD LISTA — OPERAÇÕES NECESSÁRIAS

## Remover por índice

Fornecemos um índice.

Caso bem sucedido:

- O elemento que correspondia ao índice deixará de fazer parte da coleção,
- O tamanho diminui em 1 unidade.

Caso mal sucedido: nenhuma propriedade é alterada.

Observações: Não existe índice maior que o tamanho.

UFC CAMPUS QUIXADÁ

# TAD LISTA — OPERAÇÕES NECESSÁRIAS

## Obter por índice

Fornecemos um índice.

Caso bem sucedido: temos o elemento correspondente.

Caso mal sucedido: nenhuma propriedade é alterada.

Observações: Não existe índice maior que o tamanho.

## Obter por elemento

Fornecemos um elemento.

Caso bem sucedido: temos o elemento índice correspondente.

Caso mal sucedido: nenhuma propriedade é alterada.

Observações: Elementos não-colecionados não possuem índice.

JFC
AMPUS
UIXADÁ

**Observe**

Vetores simples do C suportam todas essas regras. Podemos considerá-los como um tipo de *Lista*.

**Observe**

Veja que nessas regras não há exigências sobre **como** esses elementos devem ser armazenados, ou quais dados precisamos armazenar. Exigimos apenas que os conceitos possam ser representados.

UFC CAMPUS QUIXADÁ

# TAD LISTA — ONDE SÃO INTERESSANTES?



UFC CAMPUS QUIXADÁ

# TAD LISTA — ONDE SÃO INTERESSANTES?



Podemos modelar usando apenas as garantias fornecidas por *Listas*.

UFC CAMPUS QUIXADÁ

# TAD LISTA — ONDE SÃO INTERESSANTES?

Qualquer contexto onde simplesmente colecionar elementos é suficiente.

- Estruturas concretas de *Listas* são geralmente as *Coleções* mais simples.
- Índices podem não ser necessários, mas geralmente temos algum ganho.

UFC CAMPUS QUIXADÁ

# TAD LISTA — NA PRÁTICA

**Em *C++*, onde encontramos *Listas*?**

Os tipos `vector` e `list` na biblioteca padrão do *C++* são implementações de *Lista*.

UFC CAMPUS QUIXADÁ

# TAD LISTA — NA PRÁTICA

## Em *C++*, onde encontramos *Listas*?

Os tipos `vector` e `list` na biblioteca padrão do *C++* são implementações de *Lista*.

## Exercício

Tente descobrir, dos métodos dessas classes, quais deles implementam as operações que citamos.

UFC CAMPUS QUIXADÁ