C++
Básico ao Avançado
Plante árvores

Heitor Rodrigues Savegnago

UFABC Rocket Design

2017.3

Estruturas de dados

Estruturas de dados

- Armazenar quantidade indefinida

Estruturas de dados

- Armazenar quantidade indefinida
- Localizar

Estruturas de dados

- Armazenar quantidade indefinida
- Localizar
- Ordenar

Estruturas de dados

- Armazenar quantidade indefinida
- Localizar
- Ordenar
- Qualquer tipo de item pode ser armazenado

Estruturas de dados

- Armazenar quantidade indefinida
- Localizar
- Ordenar
- Qualquer tipo de item pode ser armazenado
- É possível criar várias versões de estruturas de dados

Estruturas de dados

- Armazenar quantidade indefinida
- Localizar
- Ordenar
- Qualquer tipo de item pode ser armazenado
- É possível criar várias versões de estruturas de dados
- Podem ter vários tipos de regras

Estruturas de dados

- Armazenar quantidade indefinida
- Localizar
- Ordenar
- Qualquer tipo de item pode ser armazenado
- É possível criar várias versões de estruturas de dados
- Podem ter vários tipos de regras
- São muito versáteis

Células

Células

Antes de começar, precisamos conhecer o formato de uma célula:

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento
  ou elementos

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela (não será eliminado)

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela (não será eliminado)
- Da forma que analizaremos:

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela (não será eliminado)
- Da forma que analizaremos:
  - Elemento único

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela (não será eliminado)
- Da forma que analizaremos:
  - Elemento único
  - Conecta-se a duas células

Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela (não será eliminado)
- Da forma que analizaremos:
  - Elemento único
  - Conecta-se a duas células
  - Armazenado na *heap*

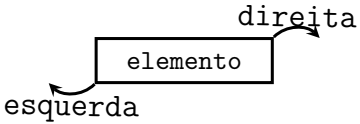
Células

Antes de começar, precisamos conhecer o formato de uma célula:

- Deve ser capaz de armazenar elemento *ou elementos*
- Deve ser capaz de conectar-se a outra célula *ou células*
- Normalmente é armazenada na *heap*
- É recomendada a presença de uma célula sentinela (não será eliminado)
- Da forma que analizaremos:
  - Elemento único
  - Conecta-se a duas células
  - Armazenado na *heap*

```
struct celula
{
  <tipo> elemento;
  celula *direita;
  celula *esquerda;
};
```

Célula



Um depois do outro

Um depois do outro

- Podemos colocar essas células em sequência

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada
  - Árvore

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada
  - Árvore
- A forma que os dados são acessados determina a estrutura de dados

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada
  - Árvore
- A forma que os dados são acessados determina a estrutura de dados
  - Fila

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada
  - Árvore
- A forma que os dados são acessados determina a estrutura de dados
  - Fila
  - Pilha

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada
  - Árvore
- A forma que os dados são acessados determina a estrutura de dados
  - Fila
  - Pilha
  - Lista

Um depois do outro

- Podemos colocar essas células em sequência
- Elas ficam ligadas umas às outras *encadeada*
- A forma que elas são ligadas pode determinar a estrutura de dados
  - Simplesmente encadeada
  - Duplamente encadeada
  - Árvore
- A forma que os dados são acessados determina a estrutura de dados
  - Fila
  - Pilha
  - Lista
  - Árvore

A linha

A linha

- A diferença entre os encadeamentos é que um é simples e outro duplo

A linha

- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*

A linha

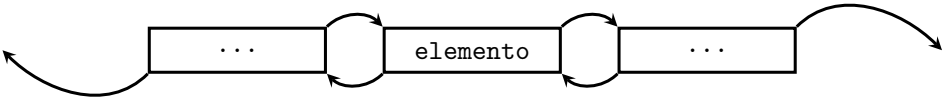
- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*
- Em outras palavas...

A linha

- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*
- Em outras palavas... Um só vai, outro vai e volta

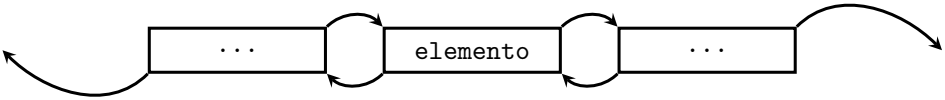
A linha

- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*
- Em outras palavas... Um só vai, outro vai e volta



A linha

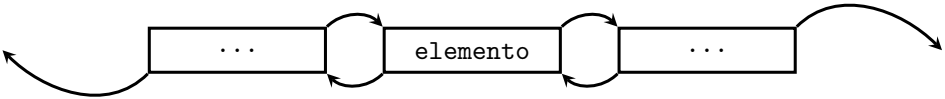
- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*
- Em outras palavas... Um só vai, outro vai e volta



- Olhar para só em cima ou só em baixo, é simplesmente encadeada

A linha

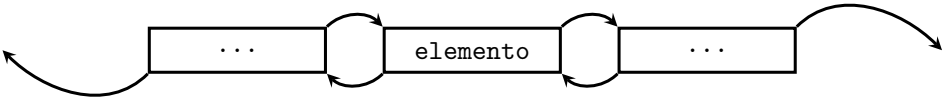
- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*
- Em outras palavas... Um só vai, outro vai e volta



- Olhar para só em cima ou só em baixo, é simplesmente encadeada
- Olhar a estrutura completa, é duplamente encadeada?

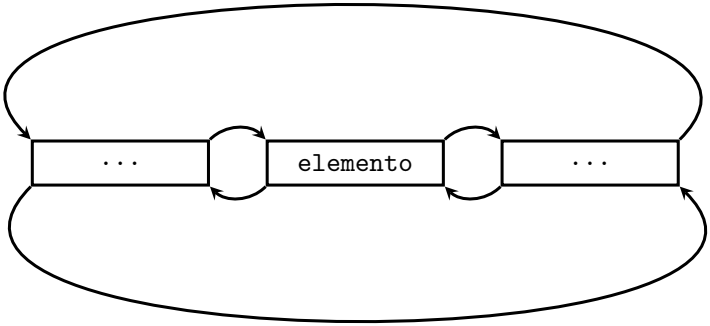
A linha

- A diferença entre os encadeamentos é que um é simples e outro duplo
- *Dã?*
- Em outras palavas... Um só vai, outro vai e volta



- Olhar para só em cima ou só em baixo, é simplesmente encadeada
- Olhar a estrutura completa, é duplamente encadeada?
- *Mas... As pontas estão abertas...*

A linha?



Detalhes de processamento

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros
- É prático utilizar recursão

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros
- É prático utilizar recursão, mas cuidado com o encadeamento

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros
- É prático utilizar recursão, mas cuidado com o encadeamento
- É mais seguro utilizar estrutura de repetição...

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros
- É prático utilizar recursão, mas cuidado com o encadeamento
- É mais seguro utilizar estrutura de repetição...
- Vale a pena criar uma classe para gerenciar as estruturas

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros
- É prático utilizar recursão, mas cuidado com o encadeamento
- É mais seguro utilizar estrutura de repetição...
- Vale a pena criar uma classe para gerenciar as estruturas
  - `insere` e `remove`?

Detalhes de processamento

- Deve existir um ao menos ponteiro para uma célula da estrutura
- É útil fazer duplamente encadeado, mas mais trabalhoso
- Tudo é trabalhado com ponteiros
- É prático utilizar recursão, mas cuidado com o encadeamento
- É mais seguro utilizar estrutura de repetição...
- Vale a pena criar uma classe para gerenciar as estruturas
  - `insere` e `remove`?
  - `insere`, `desempilha` e `desenfila`?

Filas e pilhas (e listas)

Filas e pilhas (e listas)

- São estruturas irmãs

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença:

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`
- `FIFO`: `First In, First Out`

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`
- `FIFO`: `First In, First Out`
- `LIFO`: `Last In, First Out`

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`
- `FIFO`: `First In, First Out`
- `LIFO`: `Last In, First Out`
- Ou seja, a diferença é qual item removemos

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`
- `FIFO`: `First In, First Out`
- `LIFO`: `Last In, First Out`
- Ou seja, a diferença é qual item removemos
- *E as listas?*

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`
- `FIFO`: `First In, First Out`
- `LIFO`: `Last In, First Out`
- Ou seja, a diferença é qual item removemos
- *E as listas?*
- A característica da lista é acessar por índice

Filas e pilhas (e listas)

- São estruturas irmãs
- O processamento é extremamente semelhante
- Grande diferença: `FIFO` e `LIFO`
- Filas são `FIFO`, enquanto pilhas são `LIFO`
- `FIFO`: `First In, First Out`
- `LIFO`: `Last In, First Out`
- Ou seja, a diferença é qual item removemos
- *E as listas?*
- A característica da lista é acessar por índice
- Além escolher onde inserir e onde remover

Algumas folhas

Algumas folhas

- Cada célula é ligada a duas diferentes

Algumas folhas

- Cada célula é ligada a duas diferentes
- Nenuma célula é ligada a outra

Algumas folhas

- Cada célula é ligada a duas diferentes
- Nenuma célula é ligada a outra
- Não é possível existir cíclos

Algumas folhas

- Cada célula é ligada a duas diferentes
- Nenuma célula é ligada a outra
- Não é possível existir cíclos
- O processamento, normalmente, é recursivo

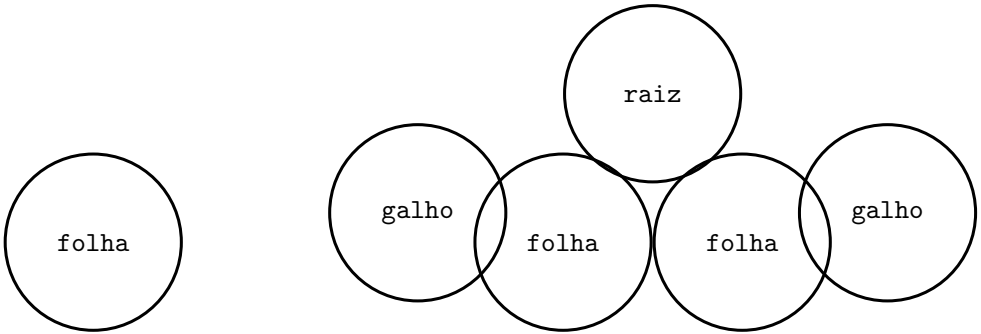
Algumas folhas

- Cada célula é ligada a duas diferentes
- Nenuma célula é ligada a outra
- Não é possível existir cíclos
- O processamento, normalmente, é recursivo
- Amplamente utilizada na computação

Algumas folhas

- Cada célula é ligada a duas diferentes
- Nenuma célula é ligada a outra
- Não é possível existir cíclos
- O processamento, normalmente, é recursivo
- Amplamente utilizada na computação
- *Infelizmente eu não consegui desenhar...*

A árvore



Vamos testar!