SBVESDD: Estruturas de Dados

Aula 05: Estruturas de Dados Lineares - Listas

Bacharelado em Ciência da Computação
Prof. Dr. David Buzatto

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Lista
Contextualização

- Existem diversas estruturas de dados lineares, sendo que já estudamos três delas: Pilha, Fila e Deque. Essas estruturas possuem características muito específicas que acabam moldando seus comportamentos. Uma Lista pode ser usada para simular cada uma das estruturas citadas, além de permitir a execução de operações mais genéricas, como a inserção e a remoção de itens em qualquer posição da estrutura e não somente em suas extremidades.

Lista
Aplicações

- A lista talvez seja a estrutura de dados linear que será mais utilizada por vocês, visto sua generalidade e aplicabilidade, pois pode ser usada em todos os casos que a Pilha, a Fila e a Deque são utilizadas. Sempre que houver a necessidade de se armazenar dados de forma linear, a Lista provavelmente será a primeira opção a ser considerada.

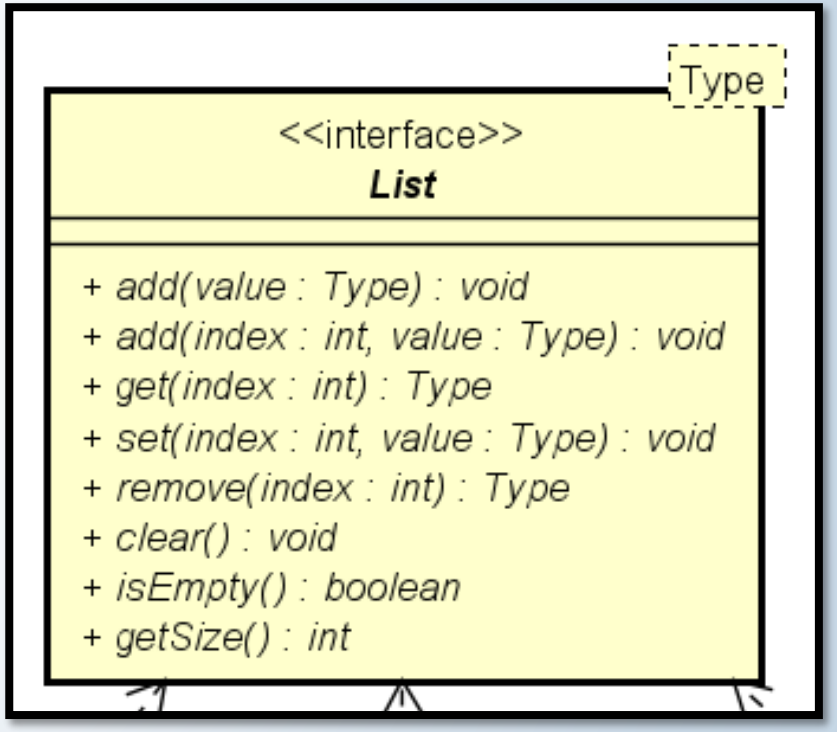
Lista
Implementação

- Iremos estudar três formas de implementar Listas:
  1. Lista redimensionável;
  2. Lista com estrutura encadeada dupla;
  3. Lista circular com estrutura encadeada dupla.

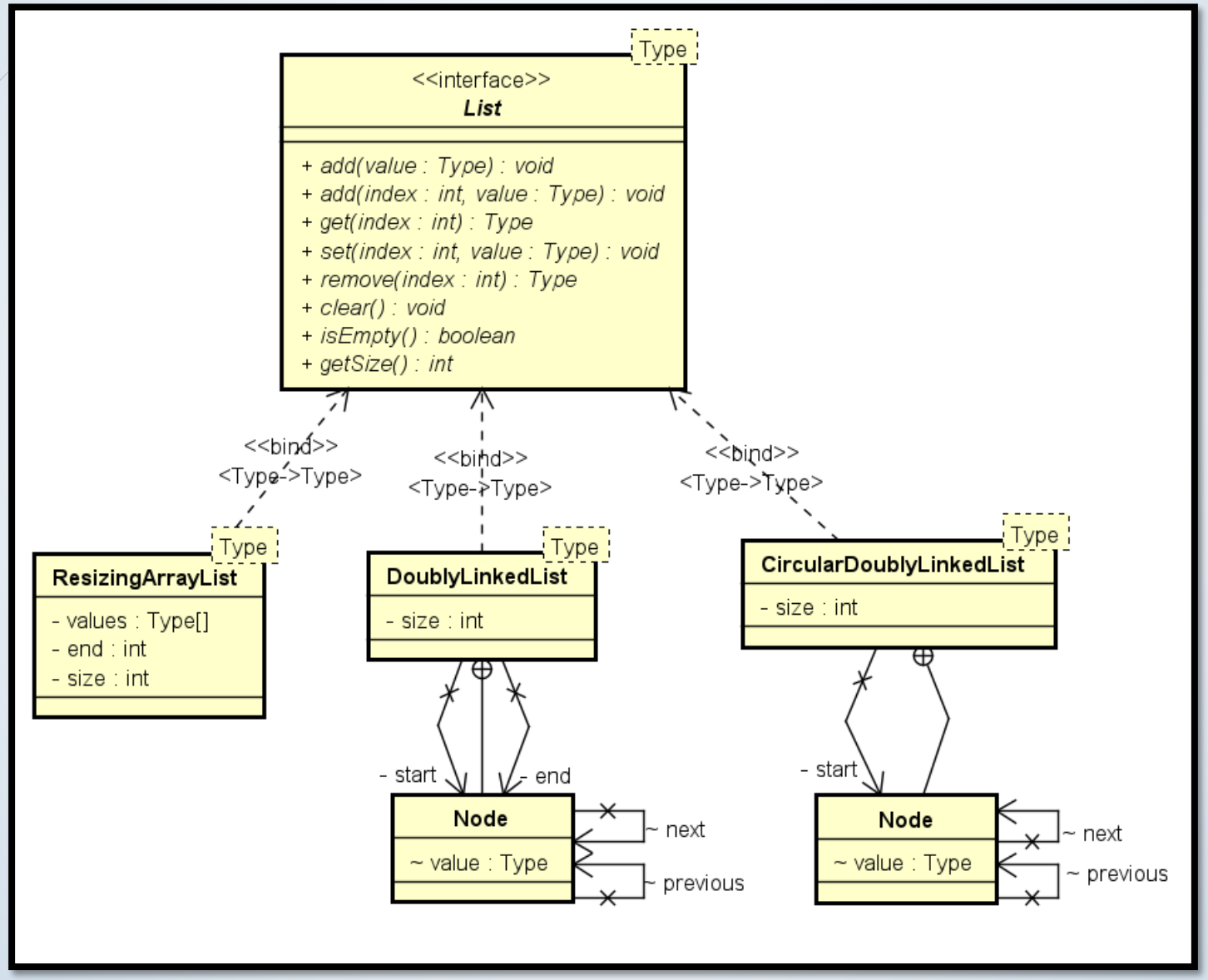
Obs: Poderíamos considerar ainda listas com tamanho fixo e com estruturas encadeadas simples, mas iremos focar nas implementações mais "reais", agora que já entenderam a evolução baseada em melhorias, que foi sendo proposta nas aulas.

Lista
Implementação

- Seguiremos uma API padrão (em inglês):
  - **`add:`** insere um item/elemento no fim ou em uma posição específica;
  - **`get:`** obtém um item/elemento de uma posição específica;
  - **`set:`** altera o valor de um item/elemento da lista em uma posição específica;
  - **`remove:`** remove um item/elemento de uma posição específica;
  - **`clear:`** limpa a lista, removendo todos os elementos;
  - **`isEmpty:`** verifica se a lista está vazia;
  - **`getSize:`** obtém a quantidade de itens/elementos contidos na lista.

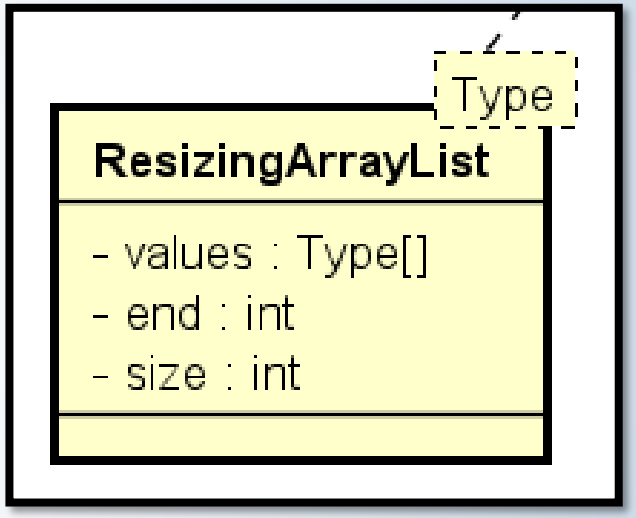


Lista
Implementação



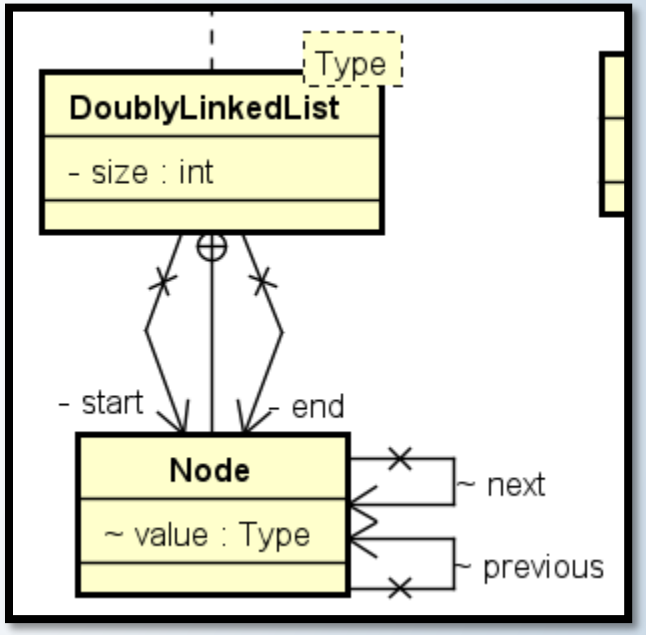
Lista
Implementação: **ResizingArrayList**

- Implementação usando um array como estrutura de armazenamento;
- Implementação com a marcação do início/primeiro para a esquerda e o fim/último para direita (fim do array);
- A quantidade de elementos que podem ser armazenados, após a instanciação da lista, é variável, pois ela crescerá ou diminuirá em função de inserções e remoções;
- **Para pensar:**
  - Qual a ordem de crescimento das operações?
  - Há como melhorar?
  - Mudar a topologia resolve?
  - Usar mapeamento de endereços?



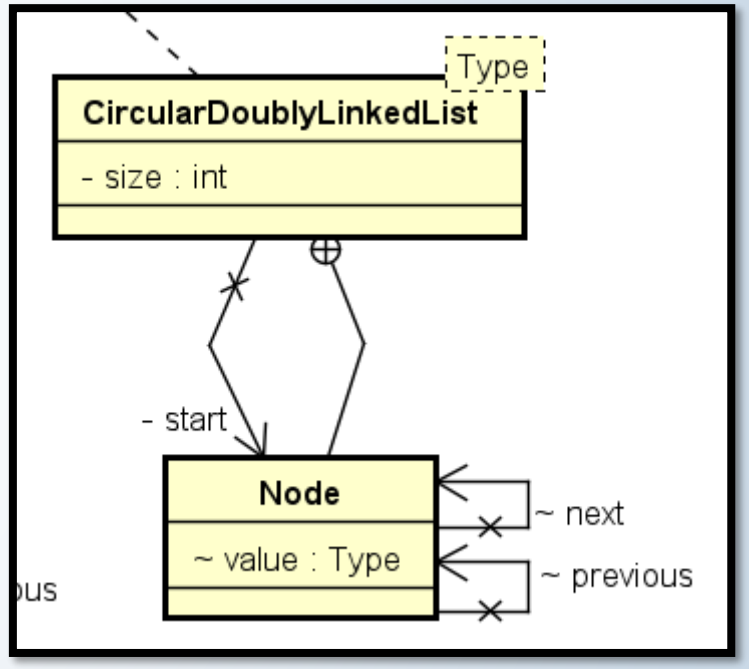
Lista
Implementação: **DoublyLinkedList**

- Implementação usando uma estrutura baseada em nós (**nodes**) com encadeamento duplo;
- O início e o fim da lista são indicados por membros do tipo dos nós;
- A alocação e liberação de espaço para o armazenamento dos elementos tem característica dinâmica;
- **Para pensar:**
  - Qual a ordem de crescimento das operações?
  - Há como melhorar?
  - Precisa melhorar?



Lista
Implementação: **CircularDoublyLinkedList**

- Implementação usando uma estrutura baseada em nós (**nodes**) com encadeamento duplo;
- Em uma lista circular, apenas a marcação do início é necessária;
- A alocação e liberação de espaço para o armazenamento dos elementos tem característica dinâmica;
- **Para pensar:**
  - Qual a ordem de crescimento das operações?
  - Há como melhorar?
  - Precisa melhorar?



Bibliografia

SEDGEWICK, R.; WAYNE, K. **Algorithms**. 4. ed. Boston: Pearson Education, 2011. 955 p.

GOODRICHM M. T.; TAMASSIA, R. **Estruturas de Dados & Algoritmos em Java**. Porto Alegre: Bookman, 2013. 700 p.

CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; STEIN, C. **Algoritmos – Teoria e Prática**. 3. ed. São Paulo: GEN LTC, 2012. 1292 p.