NOTA: Esta ficha está dividida em duas partes (Parte I e Parte II), pretende-se que a primeira parte seja seguida com os slides da aula teórico-prática. Na segunda parte pretende-se que o aluno consiga realizar os exercícios pondo em prática a matéria abordada nos slides e praticada na Parte I.

Parte I

Exercício 1

Este primeiro exercício consiste na criação de uma lista ligada (`LinkedList`). A lista ligada deverá possuir as seguintes operações: Add e Remove.
Para que seja possível verificar a integridade da lista ligada crie uma função que imprima todos os elementos da lista.
Atenção: Não confundir com a classe `java.util.LinkedList` (disponível na plataforma de coleções do Java)

Exercício 2

Criar uma implementação de lista ligada que use nós sentinela**.** Esta Implementação deverá também possuir as operações Add e Remove. Para que seja possível verificar a integridade da lista ligada crie uma função que imprima todos os elementos da lista.

Exercício 3

Consegue perceber a diferença entre as duas implementações anteriores? Observe bem as operações implementadas.

Exercício 4

Criar uma lista duplamente ligada (`DoublyLinkedList`) capaz de realizar as seguintes operações:

- Inserir um nó na cabeça.
- Remover o primeiro nó de uma lista.
- Remover o último nó de uma lista.
- Indicar se a lista está vazia ou não.
- Criar uma função para percorrer e imprimir todos os elementos da lista.

Parte II

Exercício 1

Responda às seguintes questões:

- Quais são os componentes básicos que compõem uma lista ligada?
- Para que serve um nó numa lista ligada?
- Qual a diferença entre um *array* e uma lista ligada?
- Qual a diferença entre uma lista ligada e uma lista duplamente ligada?

Exercício 2

Criar as seguintes operações na (`DoublyLinkedList`):

- Devolver um *array* dos elementos.
- Devolver um *array* de todos os elementos até uma data posição.
- Devolver um *array* de todos os elementos depois de uma dada posição.

- Devolver um *array* de todos os elementos entre um intervalo de posições.

Exercício 3

Alterar a `DoublyLinkedList` de forma que esta seja capaz de devolver uma nova `DoublyLinkedList` com apenas os elementos pares (Sempre que for instanciada com tipos inteiros).

Exercício 4

Alterar a `DoublyLinkedList` de forma que esta seja capaz de permitir saber quantos elementos iguais a um dado elemento que é passado por parâmetro estão presentes na `DoublyLinkedList`. Remover todos esses elementos e manter a integridade da `DoublyLinkedList`.