

NOTA: Esta ficha pretende rever de uma forma geral toda a matéria abordada no conjunto de slides: 2022.ED.Aula01.pdf ao 2022.ED.Aula06.pdf.

Parte I

Exercício 1

Defina e implemente a estrutura de dados **SmackStackADT** recorrendo a uma das suas implementações de *stack* em *array*. A estrutura de dados **SmackStackADT** tem o mesmo funcionamento de uma *stack* normal, no entanto possui mais um comportamento *Smack* que elimina e devolve o último elemento da *stack*. Atenção: Deverá recorrer a herança para definir e implementar esta nova estrutura.

Exercício 2

Uma **Queue** pode ser armazenada em *array* usando dois índices inteiros para guardar a próxima posição livre na cauda e a posição do próximo elemento a ser removido da cabeça. Explique o funcionamento das operações de {**enqueue**, **dequeue**, **empty**, **first**} de modo que não exista nenhum processo de *shift* dos elementos.

Exercício 3

Descreva como é que a {**stack**, **queue**} podem ser representadas com recurso a uma Lista Ligada. Qual é a principal vantagem em relação à implementação em *array*?

Exercício 4

Crie uma implementação de **List** através de uma lista ligada **CircularLinkedList**, podendo recorrer à herança caso seja aplicável, que funcione de forma circular, ou seja, o último elemento da lista deverá apontar para a cabeça e não para **null**.

Exercício 5

Crie uma implementação de **List** através de uma lista duplamente ligada **CircularDoubleLinkedList**, podendo recorrer à herança caso seja aplicável, que funcione de forma circular, ou seja, a referência seguinte da cauda da lista deverá referenciar a cabeça e não **null**, assim como o elemento anterior à cabeça deverá referenciar a cauda da lista em vez de **null**.

Exercício 6

Desenhe uma figura da estrutura ADT {**LinkedList**, **CircularLinkedList**, **DoubleLinkedList**, **CircularDoubleLinkedList**} contendo os elementos numéricos 20, 16, 27, 92. Atenção: Não pode recorrer a nós sentinelas em nenhuma das estruturas.

Exercício 7

Qual é a complexidade de tempo em notação *Big O* das operações de travessia, inserção de um nó no início da

lista, inserção de um nó no fim da lista} para as definições ADT {**LinkedList**, **CircularLinkedList**, **DoubleLinkedList**}?

Exercício 8

Implemente um validador simples de XHTML recorrendo à sua implementação da *Stack*. Para o exercício considere que um documento XHTML só é válido se todas as *tags* abertas forem fechadas e uma *tag* só pode ser fechada se for a que está aberta nesse momento (Ver exemplo). Coloque todas as *tags* (abertura e fecho) aceites numa Lista, e quebre a *string* de entrada por espaços.

Exemplo válido:

```
<body>
  <h1> Titulo </h1>
  <p> Corpo com <a>link</a> </p>
</body>
```

Exemplos não válidos:

```
<body>
  <h1> Titulo </h1>
  <p> Corpo com <a>link</p></a>
</body>
```

```
<body>
  <h1> Titulo </h1>
  <p> Corpo com <a>link</a></p>
```