

”

E-fólio A | Folha de resolução para E-fólio

UNIDADE CURRICULAR: Estruturas de Dados e Algoritmos Fundamentais

CÓDIGO: 21046

DOCENTE: Paulo Pombinho

A preencher pelo estudante

NOME: João Paulo Alves Correia Mendes Pires

N.º DE ESTUDANTE: 2203810

CURSO: Licenciatura em Engenharia Informática

DATA DE ENTREGA: 08/04/2026

RELATÓRIO:

Na elaboração do trabalho comecei por definir as classes base `INode`, `IStack` e `IQueue`. A classe `INode` representa cada elemento da lista ligada, contendo o valor armazenado e um apontador para o nó seguinte. Esta estrutura permite flexibilidade e controlo sobre a memória. Para a pilha, utilizei um único apontador (`ptop`) para o topo, bem como um contador (`n`) para registar o número de elementos. Já na fila, utilizei dois apontadores (`pfirst` e `plast`), permitindo acesso direto ao início e ao fim da estrutura, o que garante eficiência nas operações de inserção.

Na implementação tive em conta a eficiência das operações. A inserção e remoção na pilha foram implementadas com complexidade $O(1)$, dado que apenas implicam a manipulação do topo da lista. Na fila, a utilização do apontador para o último elemento (`plast`) permite que a operação de inserção (`enqueue`) também seja realizada em tempo constante, evitando a necessidade de percorrer a lista.

Relativamente ao processamento do input, optei pela leitura linha a linha utilizando `getline`, conforme sugerido no enunciado. Para interpretar os comandos e os seus argumentos, utilizei um `stringstream`, permitindo uma análise dos dados independentemente do número de espaços entre elementos. Implementei também um mecanismo para ignorar linhas vazias e comentários. A implementação dos comandos foi feita diretamente no `main`, através de uma estrutura de decisão baseada em `if / else if`. Esta opção permite garantir que apenas um comando é executado por linha e facilita a leitura e manutenção do código. Para cada operação potencialmente inválida, incluí verificações prévias, assegurando que o programa não executa operações que comprometam a sua estabilidade.

Nos comandos específicos (`transfer_stack_queue` e `reverse_queue`), garanti que a quantidade pedida era válida e que a transferência respeitava a ordem correta dos elementos. No segundo caso, utilizei uma pilha auxiliar, conforme exigido, explorando a propriedade LIFO para inverter a ordem dos elementos da fila.

Por fim, dei especial atenção à gestão de memória, utilizando `new` e `delete` de forma controlada e garantindo que todas as estruturas são corretamente libertadas através dos métodos `clear` e dos destrutores.