Relatório de Hierarquia de Classes ED Lineares

integrantes : Felipe Rodrigues Ribeiro DRE:119052031
Igor Torres Torres da Costa DRE:119034669

Neste estudo, criamos em Python um grupo de classes que modelam certas estruturas de dados lineares cruciais, tais como pilha, fila, vetor, lista ligada simples e lista duplamente ligada. O foco foi elaborar uma classe mestra que concentra os atributos e métodos partilhados por todas essas estruturas, como determinar sua dimensão, verificar se estão vazias e outras funções essenciais. Com isso, todas as demais classes derivadas seguem esse mesmo modelo, tornando o código mais estruturado e compreensível.

Dentro do código, o vetor dinâmico atua como um array que se expande à medida que adicionamos itens, possibilitando o acesso e a modificação de valores via índices, além de dispor de métodos para inserir ou retirar itens tanto no início quanto no final. A lista ligada simples é composta por nós que apontam para o elemento seguinte, e permite inserir, remover e verificar itens no começo, além de acessar elementos por posição. Por sua vez, a lista duplamente ligada é uma versão mais rica, com nós que conhecem o próximo e o anterior, o que simplifica inserções e remoções tanto no início quanto no final, além de possibilitar a troca de elementos adjacentes e a ordenação da lista.

A pilha foi construída usando a lista ligada simples para armazenar os dados, implementando as operações padrões de empilhar (inserir) e desempilhar (remover) elementos. A fila, em contrapartida, foi edificada com nós ligados, preservando sempre a referência do primeiro e do último item, assegurando que possamos adicionar itens no final e remover do começo de maneira eficaz.

Em todas essas estruturas, adicionei validações para prevenir erros corriqueiros, como tentar acessar posições inexistentes ou remover elementos quando a estrutura se encontra vazia — nestes casos, o código lança exceções para sinalizar o problema. Esta organização torna o código apto para evoluir no futuro, facilitando a inclusão de outras estruturas lineares, já que a classe mestra define uma interface clara para todas elas.

O propósito desta implementação foi equilibrar a simplicidade e a funcionalidade, aproveitando bem os atributos do Python para lidar com referências e garantir que as operações mais habituais sejam executadas de forma rápida e segura. Além disso, o código está preparado para receber melhorias, como funções de ordenação e até iterações customizadas, conforme for necessário.