Projeto 1: Sistema de Gestão de Estoque Inteligente

1. Visão Geral:

O objetivo é desenvolver um sistema para gerenciar o inventário de produtos de uma loja. O projeto integrará a Orientação a Objetos (Encapsulamento e Herança) para modelar os produtos de forma segura e organizada, com Algoritmos de Ordenação e Busca para permitir consultas e relatórios eficientes sobre o estoque.

2. Parte I: Modelagem de Dados com Encapsulamento e Herança

Primeiro, vamos modelar os diferentes tipos de produtos que a loja pode ter, garantindo a integridade de seus dados.

Objetivos:

- Criar uma estrutura de classes lógica e extensível para diferentes categorias de produtos.
- Proteger os atributos dos produtos (ex: preço, quantidade) contra valores inválidos.

Tarefas Sugeridas:

1. Classe Base Produto (Encapsulamento):
    - Crie uma classe base Produto com atributos "privados": _nome, _codigo (um identificador único, como código de barras), _preco e _quantidade_estoque.
    - Getters (@property): Crie getters para todos os atributos, permitindo o acesso de leitura.
    - Setters (@setter): Implemente setters com validações para preco e quantidade_estoque, impedindo que recebam valores negativos.
    - Crie métodos para manipular o estoque, como adicionar_estoque(quantidade) e remover_estoque(quantidade), que encapsulam a lógica de alteração da quantidade.
2. Classes Específicas (Herança):
    - Crie subclasses que herdam de Produto para representar categorias específicas. O __init__ de cada uma deve usar super() para inicializar os atributos da classe pai.
    - Eletronico(Produto): Adiciona atributos específicos como _marca e _garantia_meses.
    - Alimento(Produto): Adiciona o atributo _data_validade.

- Vestuario(Produto): Adiciona atributos como _tamanho (P, M, G) e _cor.
- Polimorfismo: Sobrescreva um método exibir_detalhes() em cada subclasse para que, além das informações comuns do produto, ele mostre também os detalhes específicos da categoria.

3. Parte II: Gerenciamento do Estoque com Estruturas de Dados e Algoritmos

Agora, criaremos a classe que gerencia a coleção de todos os produtos, implementando os algoritmos de busca e ordenação como suas funcionalidades centrais.

Objetivos:

- Centralizar a lógica de gerenciamento do estoque em uma única classe.
- Aplicar algoritmos de ordenação para gerar relatórios (ex: listar produtos do mais barato ao mais caro).
- Utilizar algoritmos de busca para encontrar produtos de forma eficiente.

Tarefas Sugeridas:

1. Classe Estoque:
    - Crie uma classe Estoque que terá um atributo _produtos, uma lista (list) para armazenar todas as instâncias de Produto (e suas subclasses).
    - Crie um método adicionar_produto(produto) para incluir novos produtos no inventário.
2. Implementando Algoritmos de Ordenação:
    - Crie um método ordenar_produtos(criterio='nome', algoritmo='selection').
    - O criterio pode ser 'nome', 'preco' ou 'quantidade_estoque'.
    - O algoritmo pode ser 'bubble', 'insertion' ou 'selection'. Dentro do método, use if/elif para selecionar qual função de ordenação chamar.
    - Implemente os algoritmos de Bubble Sort, Insertion Sort e Selection Sort como métodos privados (ex: _bubble_sort(lista, chave)) dentro da classe Estoque. Eles receberão a lista de produtos e a chave de ordenação.
    - Exemplo de uso: meu_estoque.ordenar_produtos(criterio='preco', algoritmo='bubble')

3. Implementando Algoritmos de Busca:
    - Busca Sequencial: Crie um método buscar_por_nome(nome) que use a busca sequencial para encontrar um produto pelo nome. Este método funciona bem em listas não ordenadas.
    - Busca Binária: Crie um método buscar_por_codigo(codigo). Para que a busca binária funcione, o método deve primeiro garantir que a lista de produtos esteja ordenada pelo código. Em seguida, ele aplica o algoritmo de busca binária para encontrar o produto de forma muito mais eficiente. Isso cria uma conexão direta entre a necessidade de ordenação e a eficiência da busca.

4. Entregável Final do Projeto:

Um script Python (.py ou .ipynb) que contenha:

1. As definições das classes Produto, Eletronico, Alimento e Vestuario.
2. A definição da classe Estoque, com todos os métodos de adição, ordenação (contendo as três implementações) e busca (sequencial e binária).
3. Uma seção de "Demonstração do Sistema" que:
    - Crie várias instâncias de diferentes tipos de produtos.
    - Adicione todos os produtos ao Estoque.
    - Demonstre a busca sequencial por um nome de produto.
    - Demonstre a busca binária por um código. Mostre o estado da lista (ordenada por código) antes da busca.
    - Demonstre a ordenação do estoque por preço usando bubble sort e exiba o resultado.
    - Demonstre a ordenação por quantidade em estoque usando insertion sort e exiba o resultado.
    - Demonstre a ordenação por nome usando selection sort e exiba o resultado.