Exercício 1: TAD para Manipulação de Polinômios

Descrição:
Implemente um TAD chamado Polinomio, que representa polinômios na forma $P(x) = a_nx^n + a_{n-1}x^{n-1} + \dots + a_0$. O TAD deve permitir a criação, manipulação e análise de polinômios.

Interface do TAD:
- Construtores:
  - criar_polinomio(coeficientes: list[int]) return Polinomio:
    Cria um polinômio a partir de uma lista de coeficientes, onde o índice indica o grau (ex.: [3, 2, 1] representa $3x^2 + 2x + 1$).

- Analisadores:
  - avaliar(p: Polinomio, x: float) return float: Avalia o polinômio $P(x)$ em um valor x.
  - grau(p: Polinomio) return int: Retorna o grau do polinômio.
  - coeficiente(p: Polinomio, n: int) return int: Retorna o coeficiente do termo de grau n.

- Modificadores:
  - adicionar_termo(p: Polinomio, coeficiente: int, grau: int) return Polinomio: Adiciona um termo ao polinômio.
  - derivar(p: Polinomio) return Polinomio: Retorna o polinômio derivado.
  - somar(p1: Polinomio, p2: Polinomio) return Polinomio: Retorna a soma de dois polinômios.

Atividade:

Implemente o TAD Polinomio em Python e realize os seguintes testes:

1. Crie o polinômio $P(x) = 2x^2 + 3x + 1$.
2. Avalie $P(x)$ em $x = 2$.
3. Adicione o termo $4x^3$ e calcule o novo grau.
4. Derive o polinômio e mostre o resultado.

****

Exercício 2: TAD para Controle de Estoque (Semelhante ao exercício da segunda lista de dicionários)

Descrição:
Implemente um TAD chamado Estoque, que gerencia produtos de um armazém. Cada produto possui um nome, uma quantidade e um preço unitário.

Interface do TAD:
- Construtores:
  - criar_estoque() return Estoque: Cria um estoque vazio.
  - adicionar_produto(e: Estoque, nome: str, quantidade: int, preco: float) return Estoque: Adiciona um novo produto ao estoque.

- Analisadores:
  - listar_produtos(e: Estoque) return list[str]: Retorna uma lista com os nomes de todos os produtos no estoque.
  - quantidade_produto(e: Estoque, nome: str) return int: Retorna a quantidade de um produto específico.
  - valor_total_estoque(e: Estoque) return float: Retorna o valor total de todos os produtos no estoque.

- Modificadores:
  - atualizar_quantidade(e: Estoque, nome: str, nova_quantidade: int) return Estoque: Atualiza a quantidade de um produto específico.
  - remover_produto(e: Estoque, nome: str) return Estoque: Remove um produto do estoque.

Atividade:

Implemente o TAD Estoque em Python e realize os seguintes testes:
1. Crie um estoque com os produtos:
   - "Arroz", 20 unidades, R$5.00/unidade.

   - "Feijão", 10 unidades, R$7.00/unidade.
2. Liste todos os produtos no estoque.
3. Atualize a quantidade de "Arroz" para 30 unidades.
4. Calcule o valor total do estoque.
5. Remova o produto "Feijão" e liste os produtos restantes.

Bons estudos!!