**Exercícios de C#**

**1. Sistema de Contas Bancárias**

Crie uma classe ContaBancaria com métodos para **depositar, sacar e transferir** dinheiro entre contas.

* Use **herança** para criar ContaCorrente e ContaPoupanca.
* Implemente **exceções personalizadas** para saldo insuficiente.

**2. Manipulação de Arquivos JSON**

Crie um programa que lê um arquivo JSON contendo uma lista de produtos e permite:

* Buscar um produto por nome.
* Atualizar o preço de um produto.
* Adicionar um novo produto e salvar no arquivo.

**3. API REST com .NET**

Crie uma API em **ASP.NET Core** para gerenciar um CRUD de usuários:

* Use **Entity Framework** com um banco de dados SQL.
* Adicione autenticação via **JWT**.
* Exponha endpoints para GET, POST, PUT, DELETE.

**4. Gerador de Relatórios com LINQ**

Dado um array de objetos representando vendas (Produto, Quantidade, Data, Valor):

* Filtre vendas de um determinado mês.
* Calcule a média de vendas por produto.
* Gere um relatório ordenado pelo total vendido.

**5. Multi-threading e Performance**

Crie um programa que simula o processamento de pedidos em um e-commerce usando **Tasks e Threads**.

* Os pedidos devem ser processados em paralelo.
* Deve haver um controle para evitar **condição de corrida**.

**Exercícios de Python**

**1. Sistema de Gestão de Estoque**

Crie um programa que gerencia um estoque de produtos.

* Cada produto tem nome, quantidade, preço.
* Adicione funcionalidades para cadastrar, vender e listar produtos.
* Salve os dados em um arquivo **CSV ou banco SQLite**.

**2. Web Scraping**

Use a biblioteca BeautifulSoup para extrair dados de um site de notícias.

* Pegue o **título, data e link** das últimas notícias.
* Salve os dados em um arquivo JSON.

**3. API com Flask**

Crie uma API em **Flask** que permite cadastrar e consultar livros em um banco MySQL.

* Implemente autenticação via **JWT**.
* Use **SQLAlchemy** para modelagem de dados.
* Faça testes automatizados com pytest.

**4. Algoritmos de Busca e Ordenação**

Implemente os algoritmos **Merge Sort** e **Busca Binária** e use-os para organizar e buscar dados em uma lista grande.

**5. Paralelismo com Multiprocessing**

Crie um programa que processa grandes quantidades de dados (exemplo: um log de 1GB).

* Use **multiprocessing** para dividir a carga entre núcleos do processador.
* Meça o tempo de execução antes e depois da otimização.