Lista de Exercícios: Busca Sequencial e Busca Binária

Nível 1 – Fundamentos

1. **Busca Linear Simples**
   - Dado um vetor de números inteiros e um número alvo, use **busca sequencial** para verificar se o número está presente.
   - Extra: informe o índice se encontrar.
2. **Contar Ocorrências (Busca Linear)**
   - Conte quantas vezes um número aparece na lista usando busca sequencial.
3. **Maior Número (Busca Linear)**
   - Use busca sequencial para encontrar o **maior número** em uma lista.
4. **Menor Número (Busca Linear)**
   - Similar ao anterior, mas encontre o **menor valor**.
5. **Verificar Elemento (Busca Binária)**
   - Dada uma **lista ordenada**, implemente a **busca binária** para verificar se o elemento existe.

Nível 2 – Aplicações Práticas

6. **Busca de Nome em Lista**
   - Peça ao usuário para digitar nomes e depois procure um nome específico usando **busca linear**.
7. **Busca Binária Recursiva**
   - Implemente a versão recursiva da **busca binária** em uma lista ordenada.
8. **Comparar Tempo de Execução**
   - Compare o tempo de execução da **busca linear** e **busca binária** em uma lista com 1 milhão de elementos. Use o módulo `time`.
9. **Encontrar Primeira Ocorrência (Busca Binária Modificada)**
   - Dada uma lista ordenada com elementos repetidos, use busca binária modificada para encontrar o **índice da primeira ocorrência** de um número.
10. **Localizar Intervalo de Índices**

- Encontre o **intervalo (início e fim)** de um número que aparece mais de uma vez usando busca binária (ex: [1,2,2,2,3,4] > número 2 > índices 1 a 3).

Nível 3 – Desafios

11. **Busca em Lista de Dicionários**

- Dada uma lista de dicionários representando pessoas (`{"nome": "Ana", "idade": 25}`), implemente uma busca linear para encontrar a pessoa com nome "João".

12. **Jogo: Adivinhe o Número (Busca Binária)**

- O computador escolhe um número entre 1 e 100. O jogador tenta adivinhar, e o computador responde se é maior ou menor. Use **lógica de busca binária** para resolver com o menor número de tentativas.

13. **Buscar Produtos por Preço**

- Dada uma lista de produtos com preços, implemente uma busca para encontrar todos os produtos com um determinado preço.

14. **Implementar sua própria função `index()`**

- Crie uma função `meu_index(lista, valor)` que funcione como o método `list.index()`, usando busca sequencial.

Extras

- Para todos os exercícios de **busca binária**, lembre-se de **ordenar a lista** primeiro, se necessário.
- Use **funções** para modularizar o código (ex: `def busca_binaria(lista, alvo):`)
- Para comparar desempenho, use `time.time()` ou `timeit`.