Lista de Exercícios

Processamento da Informação

1. Faça um programa que peça ao usuário 5 valores numéricos e o coloque em uma lista. No final, mostre qual foi o maior e menor valor digitado e as suas respectivas posições na lista.

2. Crie um programa onde o usuário digite 5 valores numéricos e coloque-os em uma lista já na posição correta de inserção (sem usar o sort()). No final mostre a lista ordenada na tela.

3. Crie um programa que vai ler vários números e colocar em uma lista. Depois disso mostre:

- Quantos números foram digitados;
- A lista ordenada em ordem descrecente;
- Se o valor 5 vou digitado e está ou não na lista.

4. Crie um programa onde o usuário digite uma expressão qualquer que use parênteses. Seu aplicativo deverá analisar se a expressão passada está com os parênteses abertos e fechados na ordem correta (obs: lembre-se que uma expressão pode ter mais que um par de parênteses)

5. As mudanças climáticas intensificam a escassez de água potável no mundo. Entre os principais fatores estão: secas prolongadas que reduzem rios e reservatórios; derretimento de geleiras que afeta o abastecimento de regiões montanhosas (muitas populações — ex: Andes, Himalaia — dependem do derretimento sazonal das geleiras para abastecimento de água.); infiltração de água salgada em aquíferos devido à elevação do nível do mar; enxentes contaminam fontes de água com esgoto, lixo e produtos tóxicos; e o aumento da demanda por água em áreas cada vez mais quentes. Com o avanço acelerado das tecnologias de inteligência artificial (IA), surgem também novos desafios ambientais. Modelos como ChatGPT, buscadores inteligentes e geradores de imagem são executados em grandes centros de dados (data centers), que precisam ser mantidos refrigerados 24 horas por dia para não superaquecerem. Para isso, a solução mais comum é o resfriamento com água, muitas vezes retirada de rios, lagos ou aquíferos. Esse processo tem um alto custo ambiental e social, especialmente em regiões com escassez hídrica. Por exemplo:

- O treinamento de um único modelo de IA pode consumir mais de 700 mil litros de água (Fonte: "Making AI Less 'Thirsty': Uncovering and Addressing the Secret Water Footprint of AI Models" (2023))
- Alguns data centers chegam a usar 5 milhões de litros de água por dia.
- Em contrapartida, segundo dados do SNIS (Sistema Nacional de Informações sobre Saneamento, Brasil) e OMS: uma pessoa consome cerca de 150 litros por dia no uso doméstico.

Isso levanta uma questão urgente: será que estamos usando os recursos hídricos de forma justa? Escreva uma função Python chamada ComparaConsumo() que recebe duas listas numéricas: uma com os valores de consumo de água (em litros) por diferentes data centers em um dia e outra com os valores de consumo médio de água (em litros) por famílias em diferentes regiões no mesmo período. A função deve:

1. Calcular a média de consumo de cada lista;

2. Informar qual dos grupos consome mais água em média;

3. Retornar os dois valores médios calculados e uma frase crítica sobre o impacto do maior consumo.

6. Crie um programa que crie uma matriz 3x3 e preencha com valores lidos pelo teclado, mostrando no final:

- a soma de todos os números pares digitados;
- a soma dos valores da terceira coluna;
- o maior valor da segunda linha.

7. Quando falamos sobre indígenas, estamos falando, na verdade, de **centenas de povos diferentes**, com suas **próprias culturas**, modos de vida, histórias e línguas. Muitas línguas indígenas têm raízes comuns — como o Tupi —, mas isso não significa que sejam a mesma língua. Por exemplo, o **Guarani Mbyá**, o **Guarani Kaiowá** e o **Guarani Nhandeva** fazem parte da mesma família linguística, mas são línguas diferentes, faladas por povos distintos.

Crie um programa que pergunte ao usuário quantas **etnias** ele deseja cadastrar e para cada etnia, solicite o nome da etnia e o nome da língua falada. Em seguida armazene os dados em uma lista de listas, onde cada sublista contém uma etnia e sua respectiva língua; Crie e use uma função chamada `mostrar_etnias( )` que recebe a lista como argumento e exiba as informações cadastradas no formato:

```
- Tupinambá fala a língua Tupi
```

8. Crie um programa que leia nome e altura de várias pessoas, armazenando tudo em uma lista composta. Ao final, o programa deve mostrar:

- Quantas pessoas foram cadastradas;
- Uma listagem com as pessoas mais altas;
- Uma listagem com as pessoas mais baixas.

(obs: escolha qual altura será considerada baixa ou alta)

9. Crie um programa em Python que:

- Permita ao usuário cadastrar o **nome de um animal** e o **bioma ao qual ele pertence**.
- Armazene os dados em uma **lista composta**, onde cada sublista deve conter dois elementos: O nome do animal e o nome do bioma correspondente;
- Continue cadastrando até que o usuário diga que não quer mais (usando uma pergunta como: “Deseja cadastrar outro animal? [S/N]”).

Ao final, o programa deve mostrar o **total de animais cadastrados** e listar todos os animais agrupados por bioma, utilizando apenas listas compostas e estruturas de repetição.

10. Os dados de atividade elétrica de neurônios são frequentemente representados como sequências de 0s e 1s ao longo do tempo: **1** representa um disparo (“spike”) do neurônio em um dado instante; **0** representa ausência de atividade. Essas informações são organizadas em

matrizes que ajudam a analisar o comportamento dos neurônios em diferentes momentos e a detectar padrões como *sincronização neural*.

Crie um programa que considere 3 neurônios monitorados em 5 instantes de tempo e peça ao usuário que digite os valores 0 ou 1 para indicar se cada neurônio disparou em cada instante. Armazene os dados em uma **lista composta**, onde cada sublista representa a atividade de um neurônio ao longo do tempo; Em seguida, exiba:

- A matriz digitada (atividade dos neurônios);
- O total de spikes de cada neurônio;
- O instante com maior sincronização neural (caracterizado por mais 1s na mesma coluna).

11. Escreva um programa em Python que solicite ao usuário a **ordem** n da matriz quadrada e os n^2 valores da matriz, preenchendo-a linha por linha. Armazene a matriz como uma **lista composta**, onde cada sublista representa uma linha. Crie uma função chamada `determinante( )` que receba a matriz como argumento e:

- Para $n = 1$, retorne o único elemento da matriz;
- Para $n = 2$, utilize a fórmula do determinante de ordem 2: $ad - bc$;
- Para $n > 2$, aplique a **regra de Laplace**, expandindo a matriz pela primeira linha:

$$\det(A) = \sum_{j=0}^{n-1} (-1)^j \cdot a_{0j} \cdot \det(M_{0j})$$

  onde M_{0j} é a submatriz obtida ao remover a linha 0 e a coluna j.

Exiba a matriz e o valor final do determinante.