Lab 6.1 - Listas e Funções

Objetivos:

- ☐ Compreender o uso de *listas, acesso de índices, inserção e remoção de elementos*
- ☐ Compreender o uso de funções, parâmetros, retornos, módulos

Exercício 1 – Aquecimento

A. Escreva uma função *categorizar_por_indice_par* que receba uma lista de números inteiros. Essa função deve retornar uma lista com duas listas internas. A da esquerda contento todos os números de índice par, a segunda com os números de índice ímpar.
B. Agora escreva uma função semelhante chamada *categorizar_por_paridade*. Essa função recebe uma lista de números inteiros e retorna uma lista com duas listas internas. A lista da esquerda contém todos os elementos pares e a da direita os ímpares. (Perceba que agora se refere aos elementos em si, e não seus índices).
C. Chame ambas as funções no programa principal, e imprima sua diferença.

Exercício 2 – Filtro de palavras

Crie uma função *filtrar_palavras* que recebe uma *string* e uma lista de *strings*. O primeiro parâmetro contém o texto original, já o segundo uma lista de palavras bloqueadas. Portanto, retorne uma nova string trocando cada uma dessas palavras por “ *** ” .

Entrada:
texto = "O jogo de Minecraft é muito divertido, mas alguns consideram muito viciante."
palavras_bloqueadas = ["Minecraft", "viciante"]

Saída:
"O jogo de *** é muito divertido, mas alguns consideram muito ***."

Exercício 3 – Sistema de Inventário Simples

Crie um sistema para gerenciar o inventário de uma loja.

Etapas:

- **Estrutura de Dados:** Crie uma lista de listas. Cada lista interna representa um item no inventário e deve conter o nome do produto, o preço e a quantidade em estoque (ex: [['Teclado', 150.00, 10], ['Mouse', 80.00, 25]]).
- **Crie as Funções de Gerenciamento:**
    - adicionar_produto(inventario, nome, preco, quantidade): Adiciona um novo produto ao inventario.
    - atualizar_quantidade(inventario, nome, nova_quantidade): Encontra o produto pelo nome e atualiza sua quantidade em estoque.
    - buscar_produto(inventario, nome): Retorna as informações completas de um produto. Se não for encontrado, retorne None.
- **Crie a Função de Exibição:**
    - listar_inventario(inventario): Imprime todos os produtos do inventário de forma organizada, mostrando nome, preço e quantidade.

Exercício 4 – Aplicativo de Análise de Notas de Alunos

Crie um programa para analisar e gerenciar as notas de uma turma.

Etapas:

- **Armazene as notas em uma Lista:** Crie uma lista chamada notas e preencha-a com 10 notas aleatórias (inteiras ou decimais) de 0 a 100.
- **Crie as Funções de Análise:**
    - calcular_media(lista_notas): Uma função que recebe a lista e retorna a média das notas.
    - encontrar_maior_nota(lista_notas): Uma função que retorna a maior nota da lista.
    - encontrar_menor_nota(lista_notas): Uma função que retorna a menor nota da lista.
- **Crie a Função de Exibição:**
    - mostrar_resumo(lista_notas): Uma função que imprime um resumo formatado das notas. Ela deve chamar as funções de análise e mostrar a média da turma, a maior nota e a menor nota de forma clara.

Exercício 5 – Jogo de exploração

Imagine-se criando um simples jogo de aventura baseado em texto! Vamos escrever um código em Python que simula entrar e sair de salas em um calabouço.

Etapas:

1. **Crie uma Lista de Salas:**
    - Defina uma lista para armazenar os nomes das salas dos calabouços.
2. **Rastreie sua exploração:**
    - Inicie outra lista, vazia por agora, para representar as salas visitadas.
3. **Explore os calabouços:**
    - Use um *loop* para simular a exploração do calabouço. Dentro do loop:
        - Imprima a sala atual (acesse o último elemento da lista de salas visitadas)
        - Provenha opções para o jogador escolher a próxima sala do jogo para explorar (de uma lista de salas)
        - Baseado na escolha do jogador, adicione ao final a nova sala em salas visitadas.
    - Caso o jogador decida sair, remova a última sala visitada (simulando voltar).
4. **Imprima um Resumo da Exploração:**
    - Depois do loop de exploração, imprima um resumo das salas visitadas na ordem que elas foram exploradas.

Extra:

- Expanda o programa para incluir descrições de cada sala. Imprima a descrição com o nome da sala quando o jogador entrar na nova sala.
- Adicione uma condição para verificar se o jogador está tentando explorar uma sala já visitada. Informe ao jogador que ele já visitou essa sala e provenha opções para explorar uma sala diferente ou sair.

Extra:

- Modifique o programa para aceitar temperatura em Fahrenheit e converter para Celsius antes de armazenar na lista. Você pode encontrar a fórmula de conversão online.