**array.sort.py**

import random

# 1. Crie um array de 15 posições com números inteiros e valores aleatórios, não ordenados

array\_inteiros = [random.randint(1, 100) for \_ in range(15)]

# 2. Realize a ordenação dos dados do array utilizando o método “sort”

array\_inteiros.sort()

# 3. Imprima, na tela, o conteúdo do array ordenado

print("Array ordenado crescente:", array\_inteiros)

# 4. Utilize o método “sort” novamente, mas com parâmetros para ordenar de forma decrescente

array\_inteiros.sort(reverse=True)

# 5. Imprima o array ordenado de forma decrescente

print("Array ordenado decrescente:", array\_inteiros)

# 6. Crie um array de strings (nome, dataNascimento, cpf, rg)

array\_strings = ["nome", "dataNascimento", "cpf", "rg"]

# 7. Ordene o array de strings de forma crescente

array\_strings.sort()

# 8. Imprima o array ordenado de forma crescente

print("Array de strings ordenado crescente:", array\_strings)

# 9. Ordene o array de strings de forma decrescente

array\_strings.sort(reverse=True)

# 10. Imprima o array ordenado de forma decrescente

print("Array de strings ordenado decrescente:", array\_strings)

--------------------------------------------------------------------------------------------------------

**bubble.sort.py**

# Método Bubble Sort

def bubbleSort(array):

    # Laço externo para percorrer todo o array

    for i in range(len(array)):

        # Laço interno para comparar os elementos adjacentes

        for j in range(0, len(array) - i - 1):

            # Verifica se o elemento atual é maior que o próximo

            if array[j] > array[j + 1]:

                # Se for maior, troca os valores

                temp = array[j]

                array[j] = array[j + 1]

                array[j + 1] = temp

# Array de 15 números inteiros

array = [98, 12, 56, 89, 34, 73, 62, 21, 48, 94, 3, 77, 26, 15, 67]

# Aplicando o método bubbleSort ao array

bubbleSort(array)

# Imprimindo o array ordenado

print("Array ordenado:", array)

--------------------------------------------------------------------------------------------------------

**selection.sort.py**

# Array de 15 números inteiros

array = [23, 45, 12, 78, 34, 56, 90, 67, 18, 9, 29, 100, 3, 62, 81]

# Laço externo para iterar pelos elementos do array

for i in range(len(array)):

    # A variável 'min\_index' recebe o valor de 'i'

    min\_index = i

    # Laço interno para comparar os elementos subsequentes ao atual

    for j in range(i + 1, len(array)):

        # Verifica se o valor na posição 'min\_index' é maior que o valor na posição 'j'

        if array[min\_index] > array[j]:

            # Se sim, atualiza 'min\_index' para a posição 'j'

            min\_index = j

    # Troca os valores: o valor na posição 'i' troca com o valor na posição 'min\_index'

    array[i], array[min\_index] = array[min\_index], array[i]

# Imprime o array ordenado

print("Array ordenado:", array)

--------------------------------------------------------------------------------------------------------

**ler.txt.py**

# Abrir o arquivo 'loremipsum.txt' utilizando o método 'open'

arquivo = open('loremipsum.txt', 'r')

# Ler todo o conteúdo do arquivo e imprimir na tela

conteudo = arquivo.read()

print("Conteúdo completo do arquivo:")

print(conteudo)

# Fechar o arquivo

arquivo.close()

# Reabrir o arquivo para ler novamente

arquivo = open('loremipsum.txt', 'r')

# Ler e imprimir apenas a primeira linha

primeira\_linha = arquivo.readline()

print("\nPrimeira linha do arquivo:")

print(primeira\_linha)

# Fechar o arquivo novamente

arquivo.close()

# Reabrir o arquivo para ler os primeiros 3 caracteres

arquivo = open('loremipsum.txt', 'r')

# Ler e imprimir os 3 primeiros caracteres

tres\_primeiros = arquivo.read(3)

print("\nOs 3 primeiros caracteres do arquivo:")

print(tres\_primeiros)

# Fechar o arquivo

arquivo.close()

# Utilizando a instrução 'with' para abrir e ler o conteúdo do arquivo

with open('loremipsum.txt', 'r') as arquivo:

    conteudo\_com\_with = arquivo.read()

print("\nConteúdo do arquivo usando 'with':")

print(conteudo\_com\_with)

--------------------------------------------------------------------------------------------------------

**escrever.txt.py**

# Abrir (ou criar) o arquivo 'texto.txt' no modo de escrita ('w')

arquivo = open('texto.txt', 'w')

# Criar uma lista vazia

texto = list()

# Usar o método 'append' para adicionar frases à lista

texto.append("Eu sou fã de One Piece\n")

texto.append("Eu gosto muito de jogar Destiny 2\n")

texto.append("Quero acabar logo a faculdade\n")

# Escrever o conteúdo da lista no arquivo

for linha in texto:

    arquivo.write(linha)

# Fechar o arquivo após a escrita

arquivo.close()

print("Arquivo 'texto.txt' criado e preenchido com sucesso.")

--------------------------------------------------------------------------------------------------------

**kdd.py**

import time

# Função Bubble Sort

def bubble\_sort(arr):

    n = len(arr)

    for i in range(n):

        for j in range(0, n-i-1):

            if arr[j] > arr[j+1]:

                arr[j], arr[j+1] = arr[j+1], arr[j]

# Função Selection Sort

def selection\_sort(arr):

    for i in range(len(arr)):

        min\_idx = i

        for j in range(i+1, len(arr)):

            if arr[min\_idx] > arr[j]:

                min\_idx = j

        arr[i], arr[min\_idx] = arr[min\_idx], arr[i]

# Função para ler o arquivo txt

def ler\_arquivo(arquivo\_nome):

    palavras = []

    try:

        with open(arquivo\_nome, 'r', encoding='utf-8') as arquivo:

            for linha in arquivo:

                # Remover quebras de linha e separar palavras por espaços

                linha = linha.strip()

                palavras.extend(linha.split())

    except FileNotFoundError:

        print(f"Erro: O arquivo '{arquivo\_nome}' não foi encontrado.")

    return palavras

# Função para salvar palavras ordenadas no arquivo

def salvar\_arquivo(arquivo\_nome, palavras):

    with open(arquivo\_nome, 'w', encoding='utf-8') as arquivo:

        for palavra in palavras:

            arquivo.write(palavra + '\n')

# Lendo o conteúdo do arquivo loremipsum.txt

arquivo\_txt = 'loremipsum.txt'  # Nome do arquivo de entrada

palavras = ler\_arquivo(arquivo\_txt)

# Verificar se há palavras no arquivo lido

if not palavras:

    print(f"O arquivo '{arquivo\_txt}' está vazio ou não foi possível ler o conteúdo.")

else:

    # Ordenação com Bubble Sort

    palavras\_bubble = palavras.copy()

    start\_time = time.time()

    bubble\_sort(palavras\_bubble)

    end\_time = time.time()

    print(f"Bubble Sort - Tempo de execução: {end\_time - start\_time:.6f} segundos")

    # Ordenação com Selection Sort

    palavras\_selection = palavras.copy()

    start\_time = time.time()

    selection\_sort(palavras\_selection)

    end\_time = time.time()

    print(f"Selection Sort - Tempo de execução: {end\_time - start\_time:.6f} segundos")

    # Ordenação com o método nativo sort()

    palavras\_sort = palavras.copy()

    start\_time = time.time()

    palavras\_sort.sort()

    end\_time = time.time()

    print(f"Sort (Método Nativo) - Tempo de execução: {end\_time - start\_time:.6f} segundos")

    # Salvando as palavras ordenadas no arquivo

    arquivo\_saida = 'palavras\_ordenadas.txt'

    salvar\_arquivo(arquivo\_saida, palavras\_sort)

    print(f"As palavras ordenadas foram salvas no arquivo '{arquivo\_saida}'.")