Lab_02

//----------

Prazo de entrega: 31/03/2025

Valor: 0

Não entrega: -5 valores

//----------

1. Considere estes 3 algoritmos de ordenação:
    a. bubble sort
    b. selection sort
    c. insertion sort

    Implemente, em C, cada um deles com os seguintes protótipos/assinaturas:

    ```
    void bubble_sort(int arr[], int n, int *troca, int *comp);
    void bubble_sort_opt(int arr[], int n, int *troca, int *comp);
    void selection_sort(int arr[], int n, int *troca, int *comp);
    void insertion_sort(int arr[], int n, int *troca, int *comp);
    ```

    Simule cada função com apenas 2 arrays: com 4 e com 10^3 elementos.

    Considere dois arrays de entrada para cada algoritmo:
    - ordenado
    - em ordem reversa

    e armazene o número de trocas e de comparações que cada programa efetua. Utilize um vosso array de 4 elementos para exemplificar e justificar os números de trocas e de comparações.

    Considerando as entradas (array ordenado e ordem reversa) com 10^3 elementos, qual destes algoritmos obteve a melhor performance em termos de tempo de execução? E em termos de trocas e de comparações? Há algum que vocês sugerem como melhor solução a ser usada sempre?

    O grupo deverá escrever um relatório de, no máximo, 5 páginas sobre este experimento. O relatório deverá conter: introdução (breve explicação de cada

algoritmo e as suas complexidades no pior caso), objetivo, resultados, discussão dos resultados e conclusão.

A seção dos resultados deverá ter, obrigatoriamente, a seguinte tabela:

	Array com 10^3 elementos (n=10^3)					
	Array ordenado			Array em ordem reversa		
Algoritmo	Número de trocas	Número de comparações	Tempo de execução	Número de trocas	Número de comparações	Tempo de execução
Bubble Sort						
Bubble Sort com opt						
Selection Sort						
Insertion Sort						