Matias Mesa - Facundo Padilla - Sebastian Castillo - Javier Olivares

Informe del Trabajo Práctico – Algoritmos de Ordenamiento

En este trabajo práctico se estudiaron distintos métodos que permiten ordenar una lista de valores paso a paso.
El objetivo fue comprender cómo cada algoritmo toma decisiones para comparar elementos, moverlos y dejar todo ordenado.
Aunque todos llegan al mismo resultado, el modo en que cada uno trabaja es distinto, y es importante entender esas diferencias.

Durante el trabajo se implementaron tres algoritmos: **Bubble Sort**, **Selection Sort** y **Insertion Sort**. Cada uno de ellos fue adaptado para funcionar de manera "fraccionada", realizando un avance pequeño por cada paso, lo que permite observar con claridad cómo progresa el ordenamiento.

Bubble Sort

Bubble Sort recorre la lista comparando elementos vecinos. Cuando encuentra dos elementos fuera de orden, los intercambia. En cada pasada completa, el elemento más grande "sube" al final de la lista. El proceso se repite desde el principio hasta que todo queda ordenado.

Decisiones y dificultades

- Hubo que controlar en qué momento terminar la pasada para que no siga comparando en posiciones ya ordenadas.

- Un error común fue usar `i == i+1` en vez de `i = i + 1`, lo que impedía que avanzara la pasada.

Selection Sort

Selection Sort busca el elemento más pequeño de la parte no ordenada de la lista y lo coloca en el lugar correcto.
En cada pasada se fija en toda la zona restante hasta encontrar el mínimo.

Decisiones y dificultades

- Fue importante mantener min_idx siempre dentro del rango y reiniciarlo al comenzar cada pasada.

- Hubo que coordinar los índices i y j para que el visualizador mostrará todas las comparaciones.

Insertion Sort

Insertion Sort toma el elemento de la posición i y lo mueve hacia la izquierda hasta que encuentre su lugar correcto. El movimiento se realiza mediante intercambios entre elementos vecinos, de a un paso por vez. Básicamente “Inserta” al elemento en su posición correcta.

Decisiones y dificultades

- El desafío principal fue evitar comparaciones inválidas (por ejemplo, acceder a índices negativos).

- Hubo que asegurarse de que el algoritmo avanzara de forma correcta entre pasos y no quedará detenido en una sola comparación.

- Otro punto importante fue hacer que la última comparación también se muestre, incluso cuando el elemento ya está ordenado.

El trabajo permitió comprender cómo distintos algoritmos resuelven el mismo problema utilizando estrategias diferentes. La implementación paso a paso obligó a prestar atención a cada comparación y movimiento, permitiendo visualizar claramente el funcionamiento interno de cada técnica. Los algoritmos fueron implementados correctamente y cada uno presenta comportamientos particulares que enriquecen la comprensión del ordenamiento.