# **Project Documentation: Sorting Algorithm Visualizer**

**## Course Information**

- \*\*Course Name\*\*: Design and Algorithm

- \*\*Course Code\*\*: CSCI 208

- \*\*Instructor\*\*: Dr. Shereen Aly Mohamed Taie

- \*\*University\*\*: Nile University

**## Team Members**

- Omar Elhossiny - 221001028

- Marwan Ibrahim - 221001906

- Zeyad Shawky - 221001447

- Yousef Ahmed - 221001899

- Hassan Karam - 221002161

**## Project Overview**

### Introduction

The Sorting Algorithm Visualizer is a Python-based GUI application designed to help students and enthusiasts understand and visualize the working of various sorting algorithms. By providing real-time graphical representations of sorting processes, the visualizer aims to enhance comprehension of algorithm behavior and efficiency.

**### Project Objectives**

- To create an educational tool that demonstrates the execution of sorting algorithms.

- To visualize the steps and processes involved in sorting arrays.

- To provide a clear understanding of time complexities associated with each sorting algorithm.

**### Key Features**

- \*\*Visual Representation\*\*: Displays the sorting process step-by-step for better understanding.

- \*\*Multiple Algorithms\*\*: Supports Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, and Quick Sort.

- \*\*User Interaction\*\*: Allows users to start the sorting process and observe the algorithm's execution.

- \*\*Time Complexity Display\*\*: Shows the time complexity of each algorithm on the screen.

**### Technologies and Tools Used**

- \*\*Python\*\*: The core programming language used to develop the application.

- \*\*Tkinter\*\*: Python's standard GUI (Graphical User Interface) toolkit used for creating the application window and user interface.

- \*\*Sorting Algorithms\*\*: Implementations of various sorting algorithms including Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, and Quick Sort.

**### Implementation Details**

- \*\*GUI Setup\*\*: The graphical user interface is created using Tkinter. It includes buttons for starting different sorting algorithms and a canvas for displaying the sorting process.

- \*\*Array Generation\*\*: A random array of numbers is generated to be sorted. The size of the array can be adjusted.

- \*\*Sorting Visualization\*\*: Each sorting algorithm is implemented to visualize its process. The array is represented as bars whose heights correspond to array values. The bars are updated in real-time to reflect the sorting progress.

- \*\*Time Complexity Display\*\*: The time complexity (Best, Average, and Worst cases) of the selected sorting algorithm is displayed on the screen for educational purposes.

**### Usage Instructions**

1. \*\*Launch the Application\*\*: Run the Python script to open the Sorting Algorithm Visualizer.

2. \*\*Generate Array\*\*: The application starts with a randomly generated array.

3. \*\*Select Algorithm\*\*: Click on the button corresponding to the sorting algorithm you want to visualize (Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort).

4. \*\*Observe Sorting\*\*: Watch the sorting process in real-time as the bars representing the array elements rearrange themselves.

5. \*\*Time Complexity\*\*: View the time complexity of the algorithm on the screen.

**### Sorting Algorithms**

1. \*\*Bubble Sort\*\*:

- \*\*Best Case\*\*: O (n)

- \*\*Average Case\*\*: O (n²)

- \*\*Worst Case\*\*: O (n²)

2. \*\*Selection Sort\*\*:

- \*\*Best Case\*\*: O (n²)

- \*\*Average Case\*\*: O (n²)

- \*\*Worst Case\*\*: O (n²)

3. \*\*Insertion Sort\*\*:

- \*\*Best Case\*\*: O (n)

- \*\*Average Case\*\*: O (n²)

- \*\*Worst Case\*\*: O (n²)

4. \*\*Merge Sort\*\*:

- \*\*Best Case\*\*: O (n log n)

- \*\*Average Case\*\*: O (n log n)

- \*\*Worst Case\*\*: O (n log n)

5. \*\*Quick Sort\*\*:

- \*\*Best Case\*\*: O (n log n)

- \*\*Average Case\*\*: O (n log n)

- \*\*Worst Case\*\*: O (n²)

**### Message to the Doctor**

Dear Dr. Shereen Aly Mohamed Taie,

We are pleased to present our project, the Sorting Algorithm Visualizer, developed for the Design and Algorithm course (CSCI 208) at Nile University. This tool aims to provide an interactive and educational experience, helping users visualize and understand the inner workings of various sorting algorithms.

We hope this project meets your expectations and serves as a valuable learning resource for future students. We look forward to your feedback and suggestions.

Thank you for your guidance and support throughout this course.

Sincerely,

- Omar Elhossiny

- Marwan Ibrahim

- Zeyad Shawky

- Yousef Ahmed

- Hassan Karam