Table Of Contents

Introduction

Algorithms are a fundamental concept in computer science, used to solve problems and automate processes in a variety of fields. At its simplest, an algorithm is a set of instructions that a computer program follows to complete a task. These instructions can range from simple, straightforward steps to complex decision-making processes that involve multiple variables and conditions.

The importance of algorithms cannot be overstated, as they are used in virtually every aspect of computer science and technology. From simple arithmetic operations to complex machine learning algorithms, algorithms are essential for solving problems and automating tasks. They are used in a wide range of applications, including web search engines, image recognition software, financial modelling, and more.

In addition to their practical applications, algorithms are also a fascinating and intellectually stimulating subject in their own right. Studying algorithms provides insights into the underlying principles of computer science and how computers can be used to solve problems in innovative and creative ways.

One of the most common types of algorithms used all the time is Sorting algorithm. Sorting algorithms are used to arrange data and information in a logical and efficient manner. A sorting algorithm is a set of instructions that a computer program follows to rearrange a collection of data in a specific order. There are many types of sorting algorithms, each with their own strengths and weaknesses, making them suitable for different use cases and applications.

Sorting algorithms are used in a wide range of applications, from simple tasks such as sorting a list of names alphabetically to more complex tasks such as sorting large datasets for analysis. They are also used in a variety of fields, including computer science, finance, and data analysis.

Sorting algorithms can be broadly classified into two categories: comparison-based sorting algorithms and non-comparison-based sorting algorithms. Comparison-based sorting algorithms compare elements of the collection being sorted and determine their relative order. Examples of comparison-based sorting algorithms include Bubble Sort, Quick Sort, and Merge Sort. Non-comparison-based sorting algorithms do not compare elements but instead make assumptions about the data being sorted. Examples of non-comparison-based sorting algorithms include Counting Sort and Radix Sort.

Selection Sort: Selection sort is a simple and efficient sorting algorithm that works by repeatedly selecting the smallest (or largest) element from the unsorted portion of the list and moving it to the sorted portion of the list. The algorithm repeatedly selects the smallest (or largest) element from the unsorted portion of the list and swaps it with the first element of the unsorted portion. This process is repeated for the remaining unsorted portion of the list until the entire list is sorted.

Objectives

The Selection sort visualization project is a Python program that uses the Tkinter library to create a graphical user interface (GUI) for the Selection sort algorithm. The program consists of a canvas, which is the main visual component of the project, and one widget section on the bottom of the canvas.

The canvas displays a set of boxes representing the data to be sorted. The algorithm is visualized by moving the boxes around to their correct position during the sorting process. The boxes are color-coded to show their current state during the sorting process, such as being swapped or already sorted. The canvas also displays text annotations that describe the current state of the sorting process.

The widget section on the bottom of the canvas is used to modify the algorithm and data. It contains a slider that allow the user to adjust the speed of the sorting process. The user can also manually input data into an input section or generate random data using a generate button. The generate button also creates the boxes on the canvas to represent the generated data. Once the data is generated, the user can start the sorting process by clicking on a start button. An edit button also appears after the generate button is clicked, which opens a new window allowing the user to add, delete, or modify the current generated data. At the bottom of the canvas, there is a list box that displays the sorting or searching process happening in text. The list box shows the current state of the data and the actions being taken by the algorithm during the sorting or searching process. This provides an additional way for the user to follow the algorithmic process.

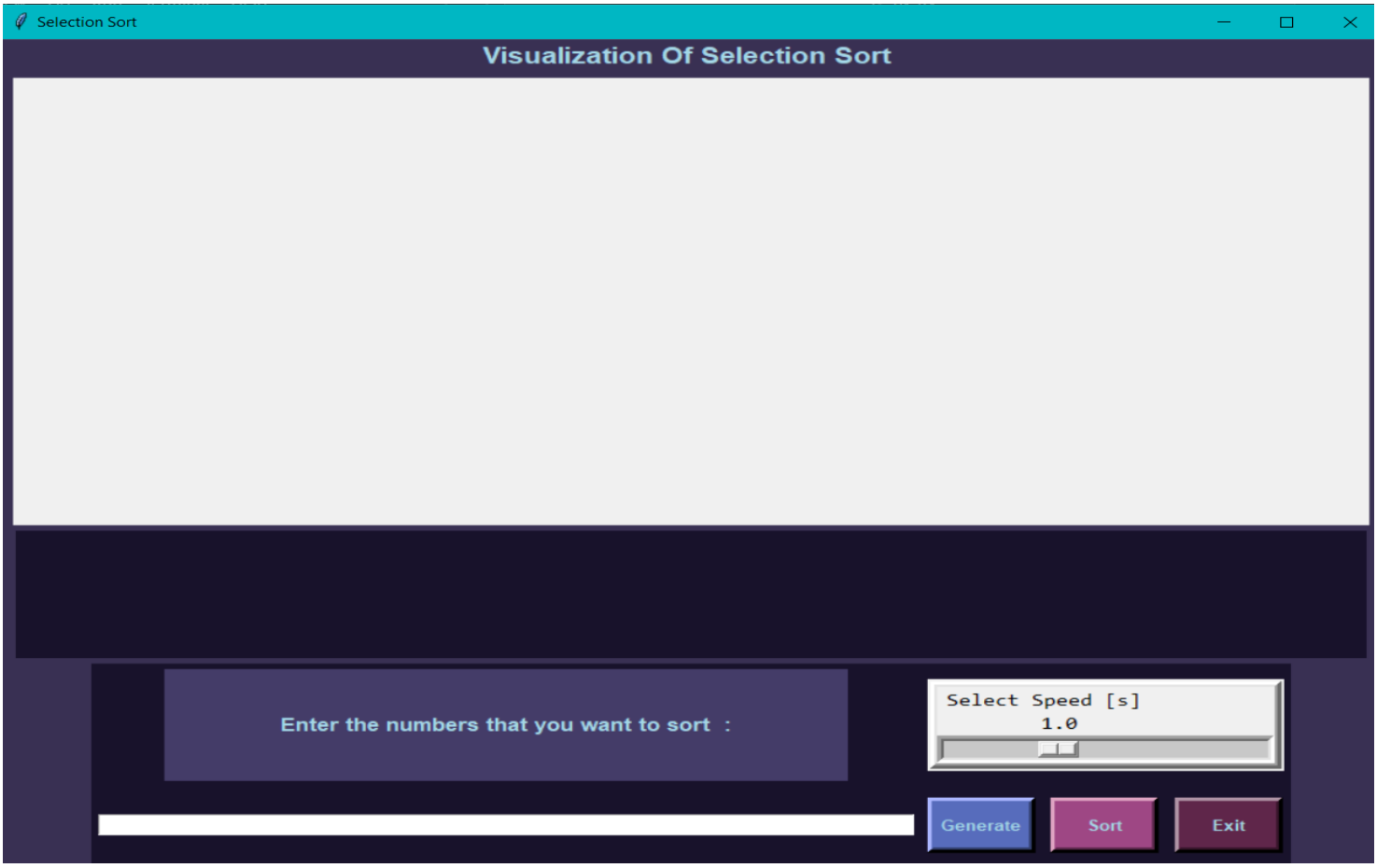
At the bottom of the canvas, there is a list box that displays the sorting or searching process happening in text. The list box shows the current state of the data and the actions being taken by the algorithm during the sorting or searching process. This provides an additional way for the user to follow the algorithmic process.

Overall, the Bubble sort visualization project provides an interactive and intuitive way to understand and visualize the Bubble sort algorithm. It allows the user to modify the data and the algorithm parameters, search for specific elements within the data, and follow the algorithmic process in real-time using visual and text-based feedback.

Description

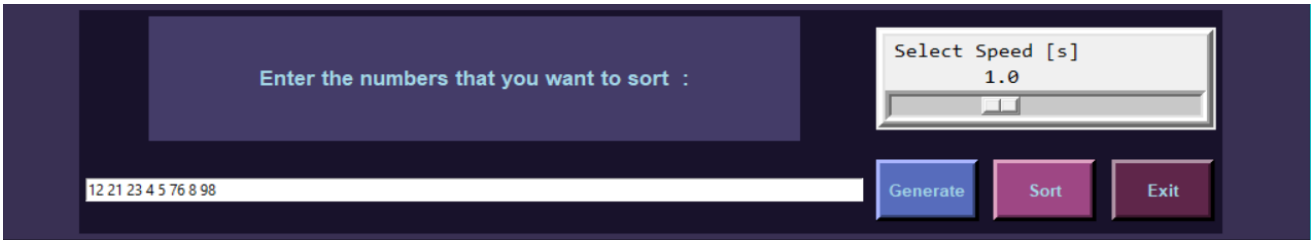
Landing Page:

This is the landing page of our program when we first run the code.



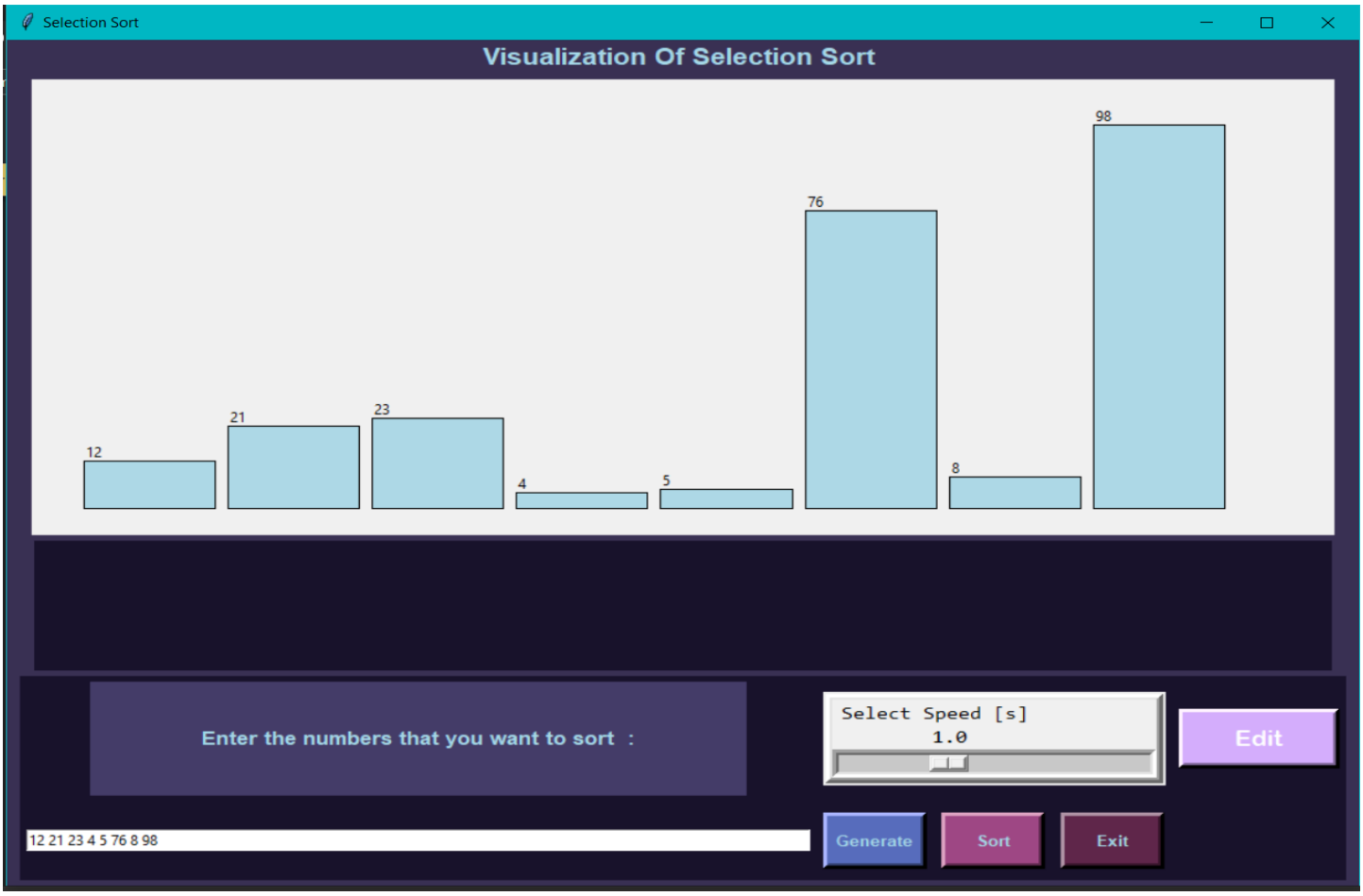
Input Section:

In the input field we can take user input with a space separating them.



Generate Data:

After giving the user input we press generate button to generate the data into canvas.

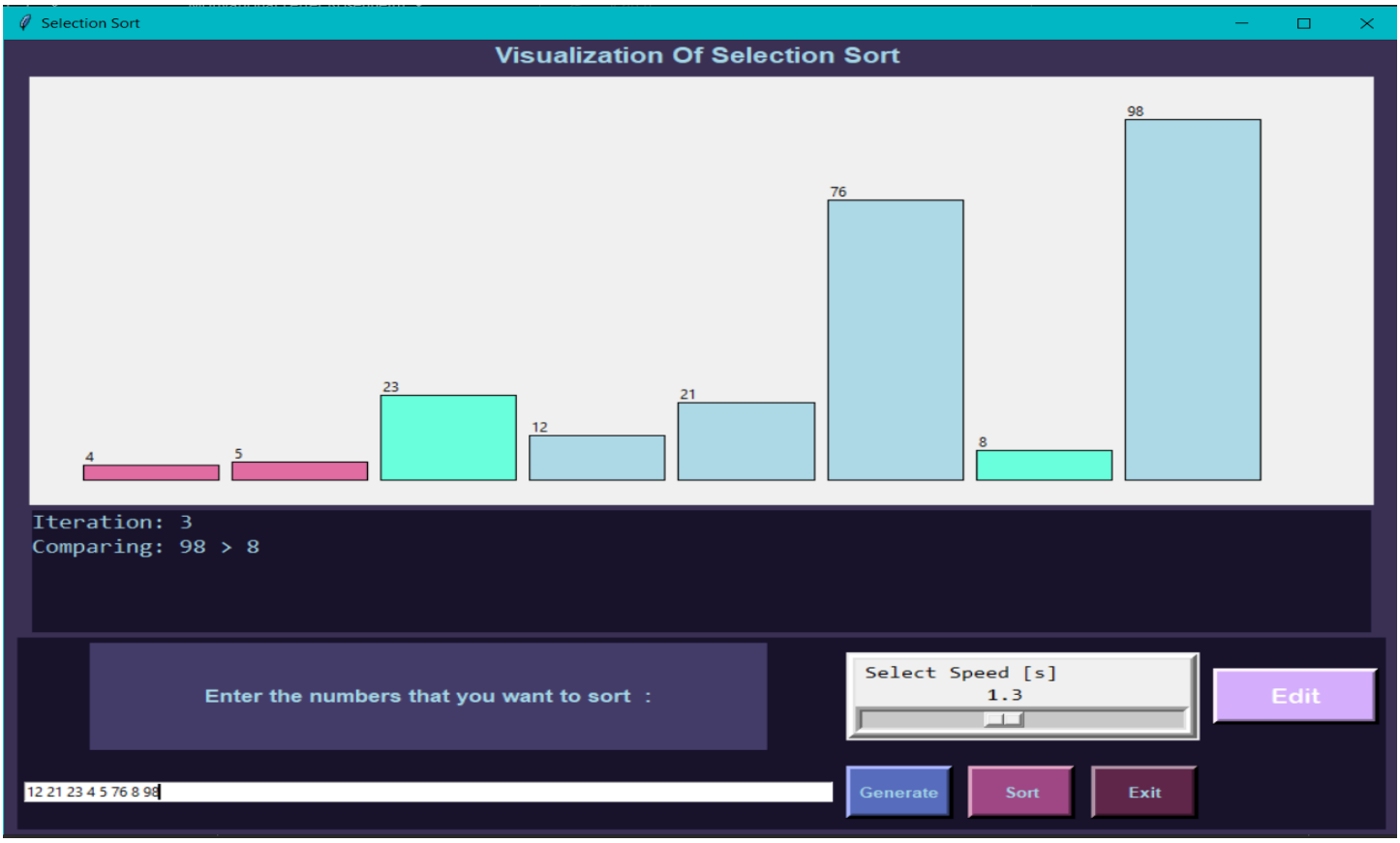


Speed Control:

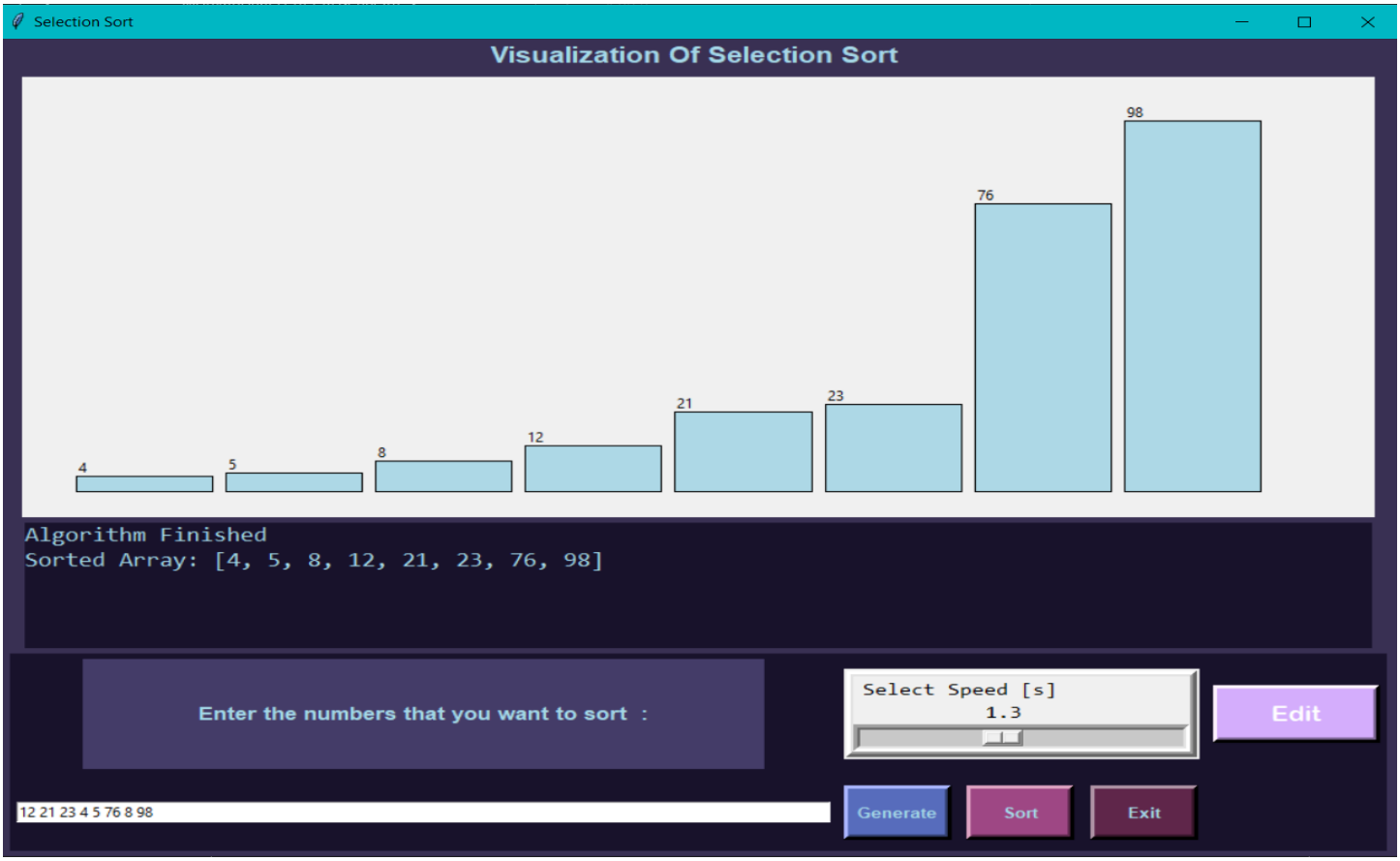
To control the speed of our algorithm we change the select speed slider. If we want the sorting in a slower manner, then we increase the slider or vice versa.

Sorting Data:

To start the sorting process, we press the sort button.

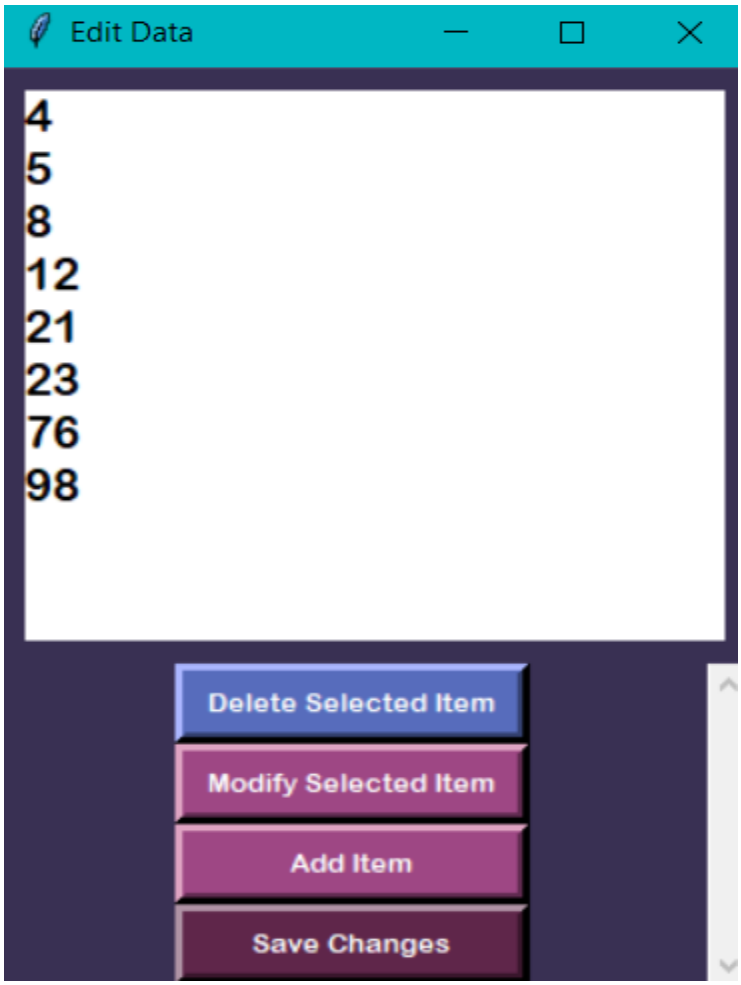


After the sorting is finished,



Edit Data:

For editing the user data, we press the edit button. As we press the button a new window pops up that's called Edit Data. As, we can see there is a text box for showing the data and several buttons for several actions. For example, for deleting an item there is delete button, for modifying an item there is modify button, for adding new items there is add button and finally for saving the changes done in this section there is save change button.



Conclusion

Our project successfully visualized the Selection Sort algorithm using Tkinter and Python, and provided users with a helpful tool for understanding how the algorithm works. By breaking down the algorithm into smaller steps and creating visual aids, we were able to make it easier for users to follow along and gain a deeper understanding of the sorting process. Our project also demonstrated the importance of visualizing algorithms for better understanding and provided a framework for future research and improvement.

For now, the visualizer only includes the bubble sort algorithm. But our team has worked hard to make such a framework that any sorting algorithm can be added and modified according to its need. And we will make sure to improve it further down the road as we add more complex algorithm into the mix. By building on our findings and continuing to innovate in this area, we can help to make complex algorithms more accessible and understandable to a wider audience.