Algorithm Visualizer

Submitted in partial fulfillment of the requirements

of the degree.

BACHELOR OF ENGINEERING

IN INFORMATION TECHNOLOGY

By

Ankita Pilvilkar	**22101B2003**
Vinaya Radekar	**22101B2004**
Anam Ansari	**22101B2005**
Arisha Rakhangi	**22101B2006**

Supervisor

Prof. Rohit Barve

Vidyalankar Institute of Technology

Accredited A+ by NAAC

Department of Information Technology

Vidyalankar Institute of Technology

Vidyalankar Educational Campus,

Wadala(E), Mumbai - 400 037

University of Mumbai

(AY 2022-23)

CERTIFICATE

This is to certify that the Mini Project entitled " **Algorithm Visualizer "** is a bonafide work of **Ankita Pilvilkar(22101B2003), Vinaya Redekar(22101B2004), Anam Ansari(22101B2005), Arisha Rakhangi(22101B2006)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering"** in **"Information Technology" .**

Prof.Rohit Barve

Supervisor

Dr. Vipul Dalal

Head of Department

Dr. S. A. Patekar

Principal

Mini Project Approval

This Mini Project entitled “**Algorithm Visualizer”** by **Ankita Pilvilkar(22101B2003), Vinaya Redekar(22101B2004), Anam Ansari(22101B2005), Arisha Rakhangi(22101B2006)** is approved for the degree of **Bachelor of Engineering** in **Information Technology.**

Examiners

1...
(Internal Examiner Name & Sign)

2...
(External Examiner name & Sign)

Date:

Place:

Contents

Abstract

Learning about algorithms and different system concepts related to programming is a difficult task and often becomes complex for students to grasp. If ever a concept is being learned using visualization techniques it becomes easy to learn and remember. There are many algorithm visualizers built for it, though students may interpret them in a wrong way and that is why it should be appropriate and correct.

Algorithms Visualizations contribute to improving computer science education. The method of teaching and learning algorithms is commonly complex to understand the problem. Visualization is a helpful technique for learning in any engineering course. In this report, an e-learning tool for Sorting Algorithms visualization is described. For example, In sorting the animation tool would represent information as a bar and once choosing a data-ordering and algorithms, the user will run an automatic animation or can step through it at their own pace.

Acknowledgments

I would like to take this opportunity to extend our heartfelt gratitude to our esteemed professor, Prof. Rohit Barve, for providing us with the invaluable opportunity to undertake the project on "Algorithm Visualizer". His unwavering guidance and mentorship helped us gain precious knowledge and conduct extensive research.

I would also like to express our sincere appreciation to our Head of Department for his/her support and encouragement throughout the project.

Finally, we would like to express our appreciation to our parents for their unrelenting support and encouragement.

STUDENT NAME

Ankita Pilvilkar(22101B2003)

Vinaya Radekar(22101B2004)

Anam Ansari(22101B2005)

Arisha Rakhangi(22101B2006)

List of Figures

Chapter 1
Introduction

1.1 Introduction

Algorithm visualizer is a Python project that allows you to create interactive visualizations of various algorithms. It can be used to help students and learners understand how different algorithms work and how they can be implemented. The project provides a graphical interface where you can select an algorithm, set its parameters, and visualize its execution in real-time. The algorithms are implemented using Python code, and the visualization is done using a library like Pygame or Matplotlib. The project is a great tool for anyone who wants to learn about algorithms and how they work, and for teachers who want to demonstrate the concepts to their students in a more engaging and interactive way.

An algorithm visualizer project can be used for educational purposes, to help students learn computer science concepts such as sorting algorithms. It can also be used for research purposes, to help researchers analyze and compare different algorithms.

One key feature of an algorithm visualizer project is interactivity. Users can manipulate the input parameters of the algorithm and see how the algorithm behaves differently with different inputs. Additionally, users can pause the algorithm's progress to examine the intermediate steps, which is useful for understanding the algorithm's inner workings.

Another important feature of an algorithm visualizer project is customization. Users should be able to choose from different algorithms to visualize. They should also be able to adjust the speed of the algorithm's progress and pause and resume the visualization as needed.

1.2 Motivation

1. Enhance understanding: The primary motivation of an algorithm visualizer is to enhance understanding and make algorithms more accessible to a wider audience. By visualizing how an algorithm works, users can better understand its inner workings and gain a deeper understanding of how it operates.

2. Learning aid: Algorithm visualizers can be an effective learning aid for students and professionals who are studying computer science or programming. They can help users to better understand complex algorithms and to practice implementing them in a visual and interactive way.

1.3 Problem Statement & Objectives

Problem Statement

The problem at hand is the difficulty faced by individuals in understanding and comprehending complex algorithms. Algorithms serve as the backbone of numerous computer science and programming concepts, but their abstract nature and intricate workings often pose a significant challenge for learners. Traditional textual explanations and code snippets can be insufficient in providing a comprehensive understanding of algorithmic processes, hindering effective learning and application.
The existing educational resources and tools available often lack an interactive and visual platform that allows learners to observe and analyze algorithms in action. Without a visual representation that showcases the step-by-step execution and data manipulation of algorithms, students may struggle to grasp the underlying concepts, identify patterns, and optimize algorithmic solutions efficiently.

Objectives

- Providing a visual understanding of how an algorithm works, making it easier to understand for beginners and experts alike.
- Facilitating the debugging process by allowing users to visualize the algorithm's behavior and identify any issues or errors.
- Providing a tool for testing and benchmarking algorithms, helping users to compare the performance of different algorithms under different conditions.
- Encouraging experimentation and innovation by allowing users to modify and customize existing algorithms or create their own.

1.4 Organization of the Report

User Interface: Designing of an intuitive and user-friendly interface that allows users to interact with the algorithm visualizer, which includes inputting data, selecting algorithms, adjusting visualization settings, and viewing the step-by-step execution of the algorithm.

Algorithm Selection: Collection of algorithms for users to choose from. Allows users to select an algorithm and visualize its execution.

Step-by-Step Execution: Implementing a mechanism to show the step-by-step execution of the algorithm. This involves highlighting the current state of the data structures, animating the movement of elements, or displaying textual explanations of the actions being performed.

Customization and Interaction: Allow users to customize the visualization according to their preferences.

Chapter 2
Literature Survey

2.1 Survey of Existing/Similar System

Over the years, there have been many studies and papers on the use of sorting algorithms as visual aids. Some are comprehensive views on how to create animations and perform statistical analysis, and others focus on different techniques aimed for increased understanding of a similar animation. By similar, I mean that between two studies, the animation used may be similar, but the difference in analysis was geared to how the algorithm was used. I would like to say upfront that John T. Stasko was a prominent figure in researching animations and most of my sources include papers in which he contributed.

The paper “Algorithm Animation,” by A. Kerren and J. Stasko [3] is a step-by-step guide to analyzing the environment, means, and available coding methods to use a sorting animation. Many different types of software are listed to be used for animation, one of which was BALSA, which pioneered the interesting event approach [3, p. 3]. BALSA was created by Marc Brown and the interesting event approach was coined to determine what part of the sorting algorithm was significant to both clearly see and understand how the algorithm performs. Additional software that followed this principle were Zeus, CAT, Tango, and Samba, to name a few [3, p. 2]. These developments prove the continuing interest in creating animation tools. For example, the interesting event approach I took focused on animating the movement of data as a visual description of the algorithm.

For a direct analysis of how students respond to sorting animation, the paper “Do Algorithm Animations Assist Learning? An Empirical Study and Analysis” [6] provides an in-depth view. A post-test study was used to gather information on comparing the results of students who only had textbook resources to those who had a textbook as well as an animation for assistance. The post-test was the same for each group of students, which covered a comprehensive view of the topic.

2.2 Limitation Existing/Similar system or research gap

In programming, there are many ways to solve a problem. However, the effectiveness of the available methods varies. Some methods are much better than others at giving accurate answers.

Algorithms are used to find the best possible way to solve a problem, based on data storage, sorting and processing, and machine learning. In doing so, they improve the efficiency of a program. Algorithms are used in all areas of computing. Because it is a fantastic way of automating computer decisions.

The current situation wherein Data Structures Algorithms are taught Manually which makes it difficult to visualize the concept, this project can be helpful. Students cannot visualize how swapping of elements takes place in internal passes of various algorithms.

2.3 Mini Project Contribution

A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of the project. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem statement.

When you write a literature review in respect of your project, you have to write the researches made by various analysts - their methodology (which is basically their abstract) and the conclusions they have arrived at. You should also give an account of how this research has influenced your thesis. Descriptive papers may or may not contain reviews, but analytical papers will contain reviews.

Chapter 3

Proposed System

3.1 Introduction

An Algorithm Visualizer is one of its kind which has its aim to provide systematic visualization of algorithms and data structures for a better understanding of the concept. Since the beginning of the learning process Data Structures, it has been the hardest challenge to visualize DSA. Algorithm Visualizer is a GUI based python program to visualize common Sorting Algorithms. The project uses the Tkinter Library in python. The project file contains a python script (sort_visual.py). This is a simple GUI based project which is easy to understand and use. It's an app-based approach that provides a single and most effective platform to visualize the Data Structure and Algorithms.

3.2 Architecture/ Framework

The proposed system involves the simulation of the different types of Sorting algorithms. The technology used is python. Libraries imported were Tkinter, winsound, time, etc. The below diagram shows the working of the project, like principal parts and functions.

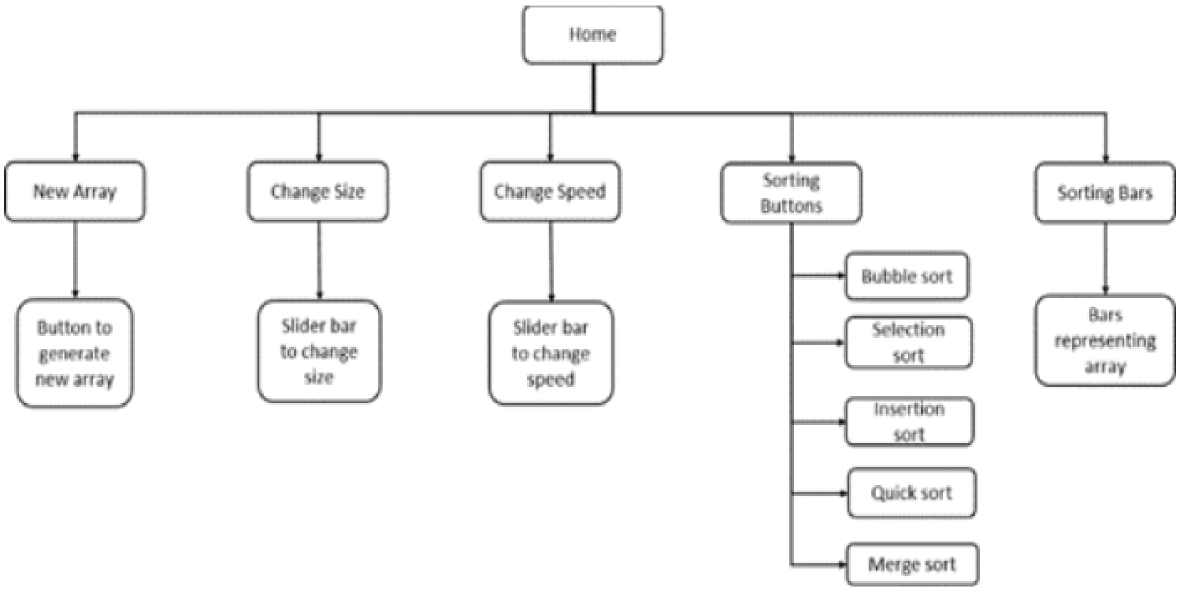


Fig.3.2.1: Block diagram for Algorithm Visualizer

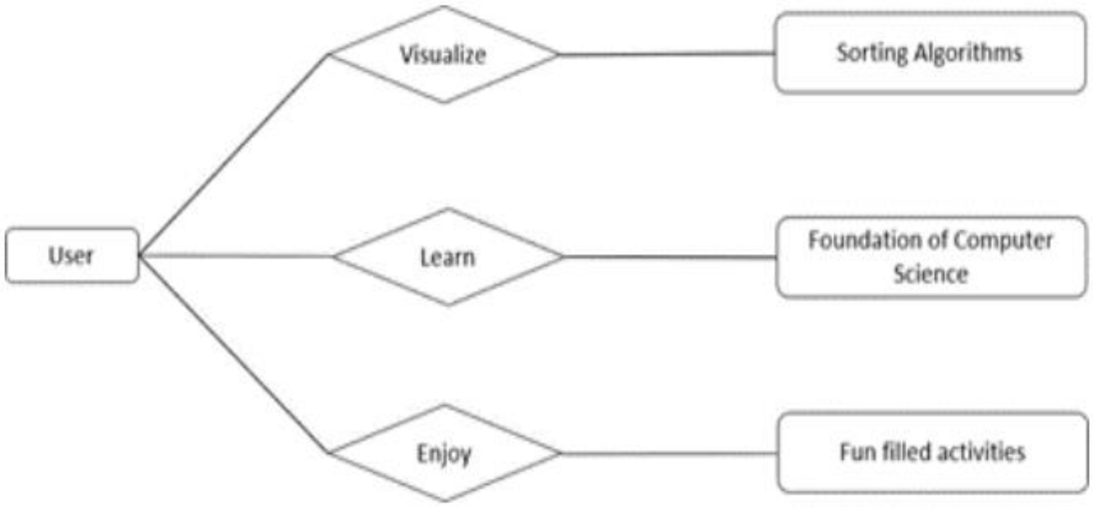


Fig.3.2.2: ER model for Algorithm Visualizer

3.3 Algorithm and Process Design

★ **Algorithms covered include:**

- ➔ Selection sort
- ➔ Bubble sort
- ➔ Insertion sort
- ➔ Merge sort
- ➔ Quicksort

★ **Process Design:**

1. Created a new window using the Tk window object.

2. Created a Canvas on the window and buttons with required commands.

3. Randomized Data Generator function to add bars using canvas.create_rectangle() function.

4. Swaps in the algorithms were animated using the window.after() function.

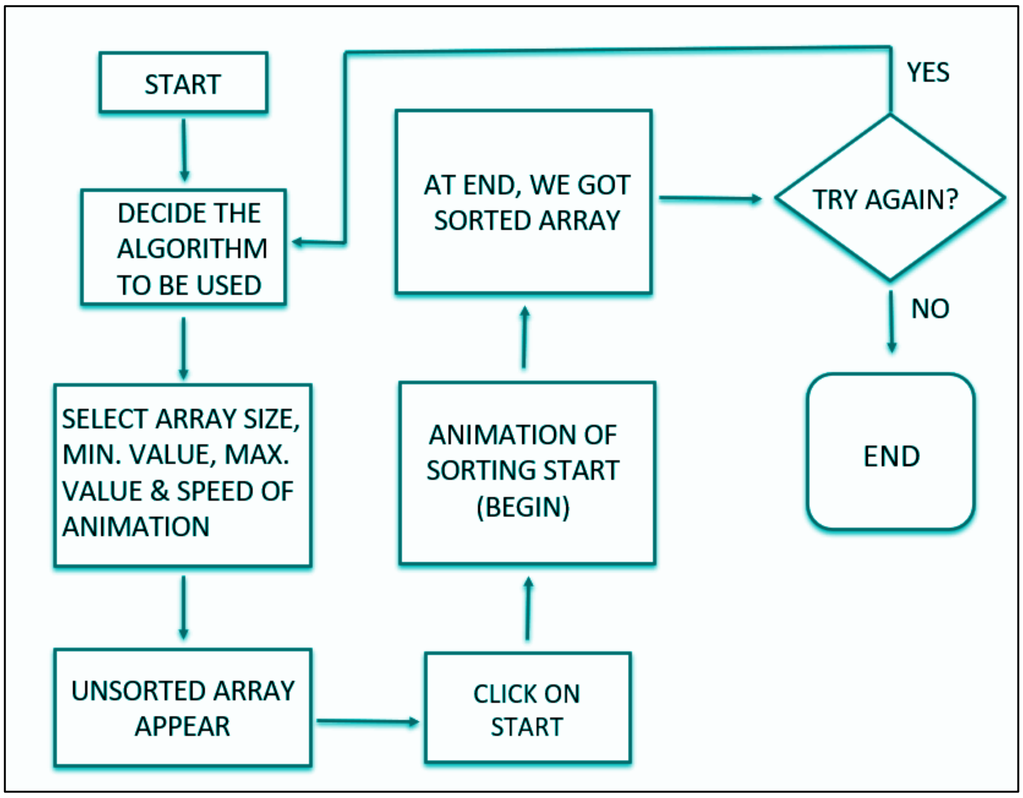

START
DECIDE THE ALGORITHM TO BE USED
SELECT ARRAY SIZE, MIN. VALUE, MAX. VALUE & SPEED OF ANIMATION
UNSORTED ARRAY APPEAR
CLICK ON START
ANIMATION OF SORTING START (BEGIN)
AT END, WE GOT SORTED ARRAY
TRY AGAIN?
YES
NO
END

Fig 3.3.1. IMPLEMENTATION(FLOW-CHART)

3.4 Details of Hardware and software

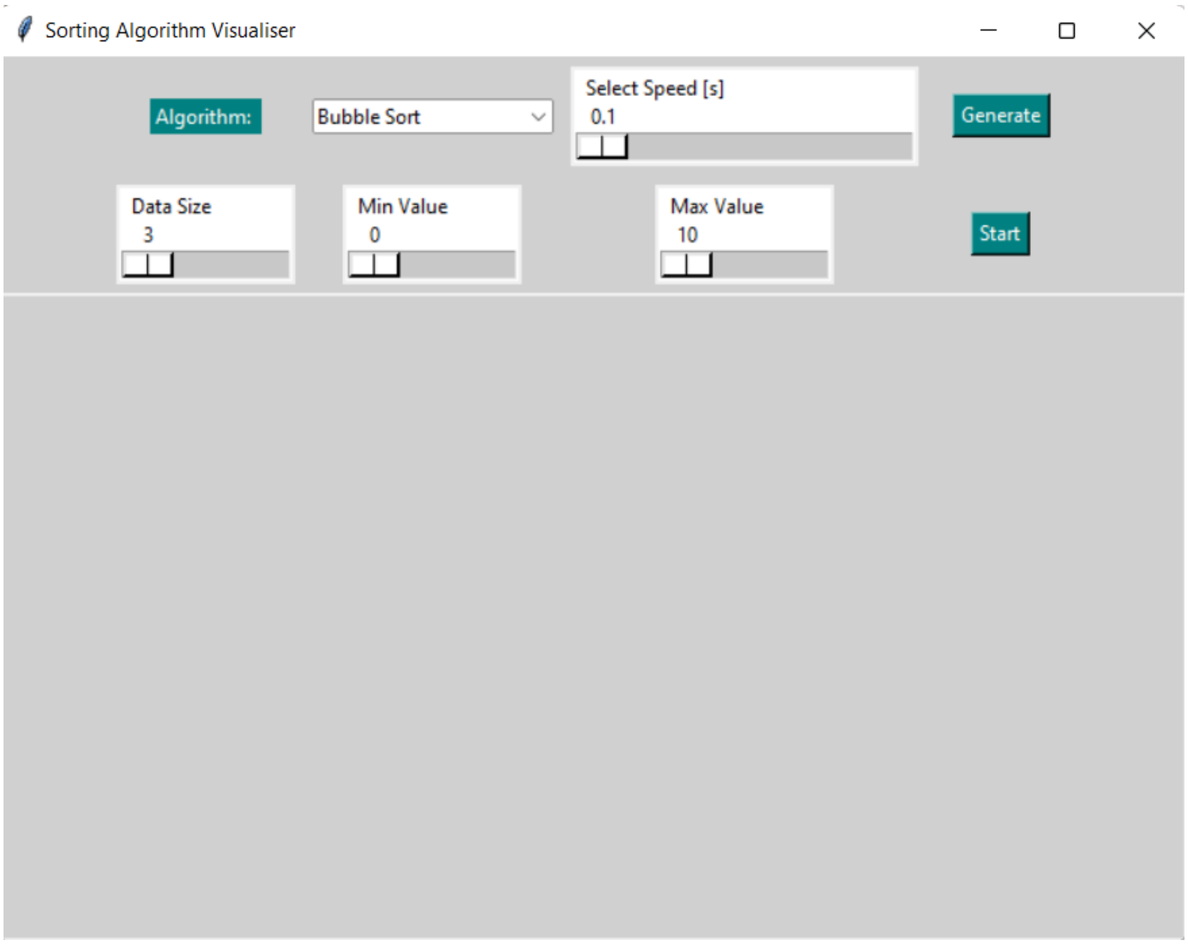
Hardware Requirements

- ➔ Laptop or Desktop

Software Requirements

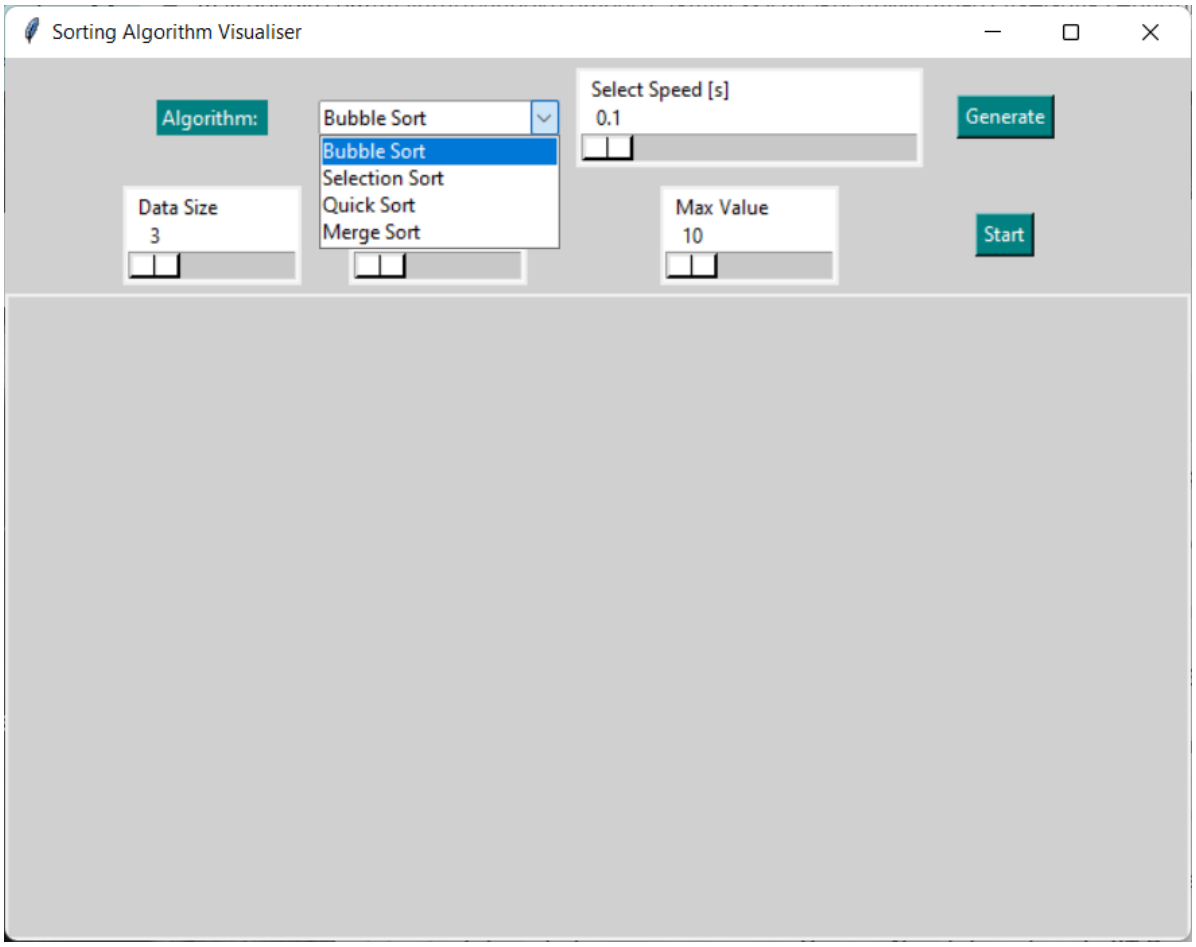
- ➔ Windows 7 or above or latest IOS version.
- ➔ Python version 3.7 or above
- ➔ Tkinter module of python with latest version

3.5 Experiment and Results



Algorithm: Bubble Sort
Select Speed [s] 0.1
Data Size 3
Min Value 0
Max Value 10

Fig 3.5.1. Dashboard



Sorting Algorithm Visualiser

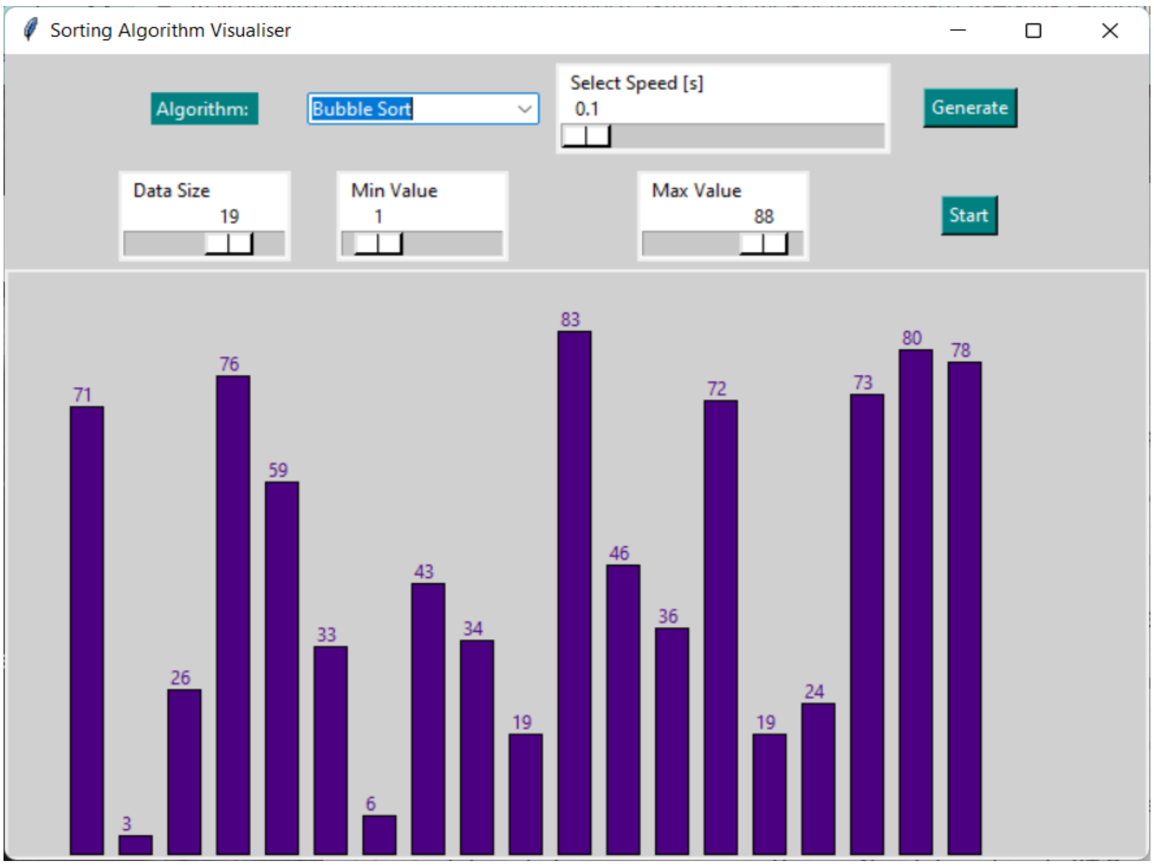
Algorithm: Bubble Sort

Select Speed [s] 0.1

Data Size 3

Max Value 10

Fig.3.5.2(a): Bubble Sort



Sorting Algorithm Visualiser

Algorithm: Bubble Sort

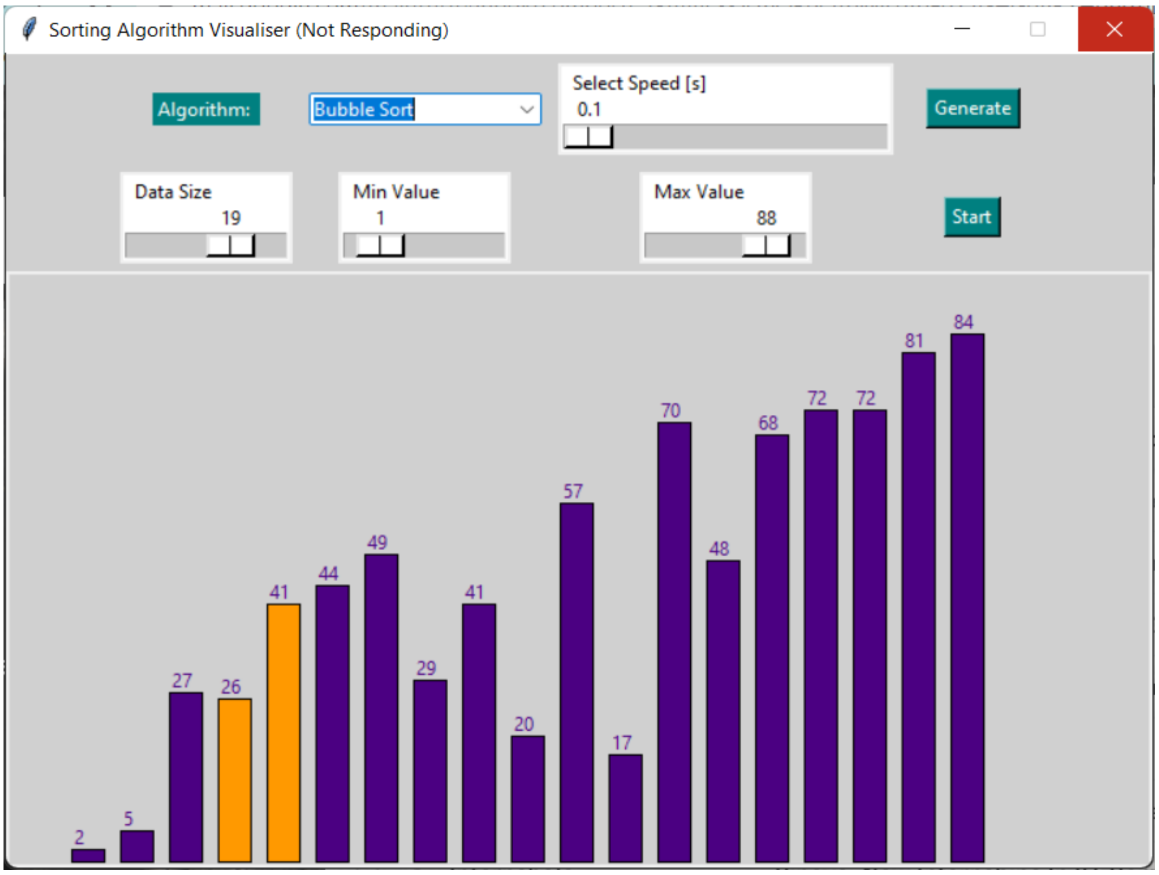
Select Speed [s] 0.1

Data Size 19

Min Value 1

Max Value 88

Fig.3.5.2(b): Bubble Sort


Sorting Algorithm Visualiser (Not Responding)

Algorithm: Bubble Sort

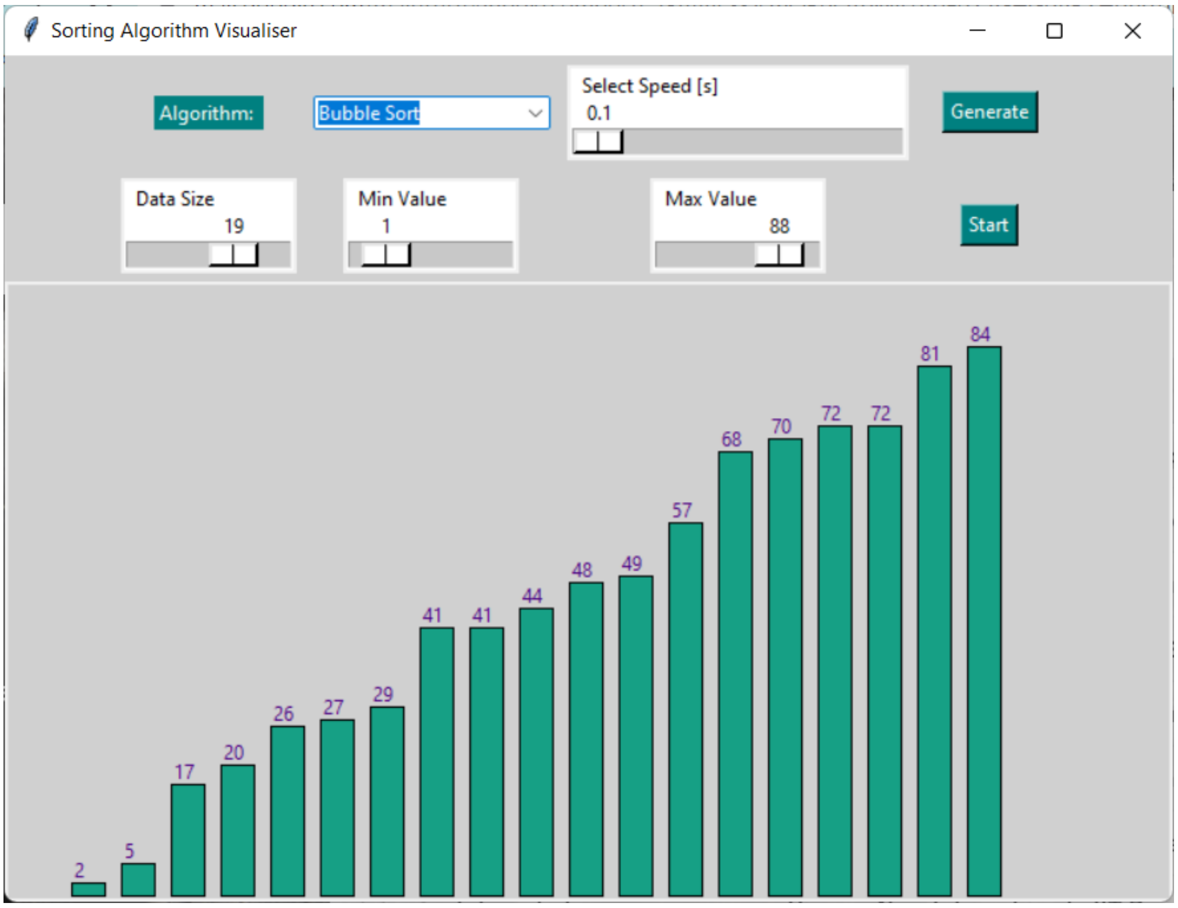
Select Speed [s] 0.1

Data Size 19

Min Value 1

Max Value 88

Fig.3.5.2(c): Bubble Sort


Sorting Algorithm Visualiser

Algorithm: Bubble Sort

Select Speed [s] 0.1

Data Size 19

Min Value 1

Max Value 88

Fig.3.5.2(d): Bubble Sort

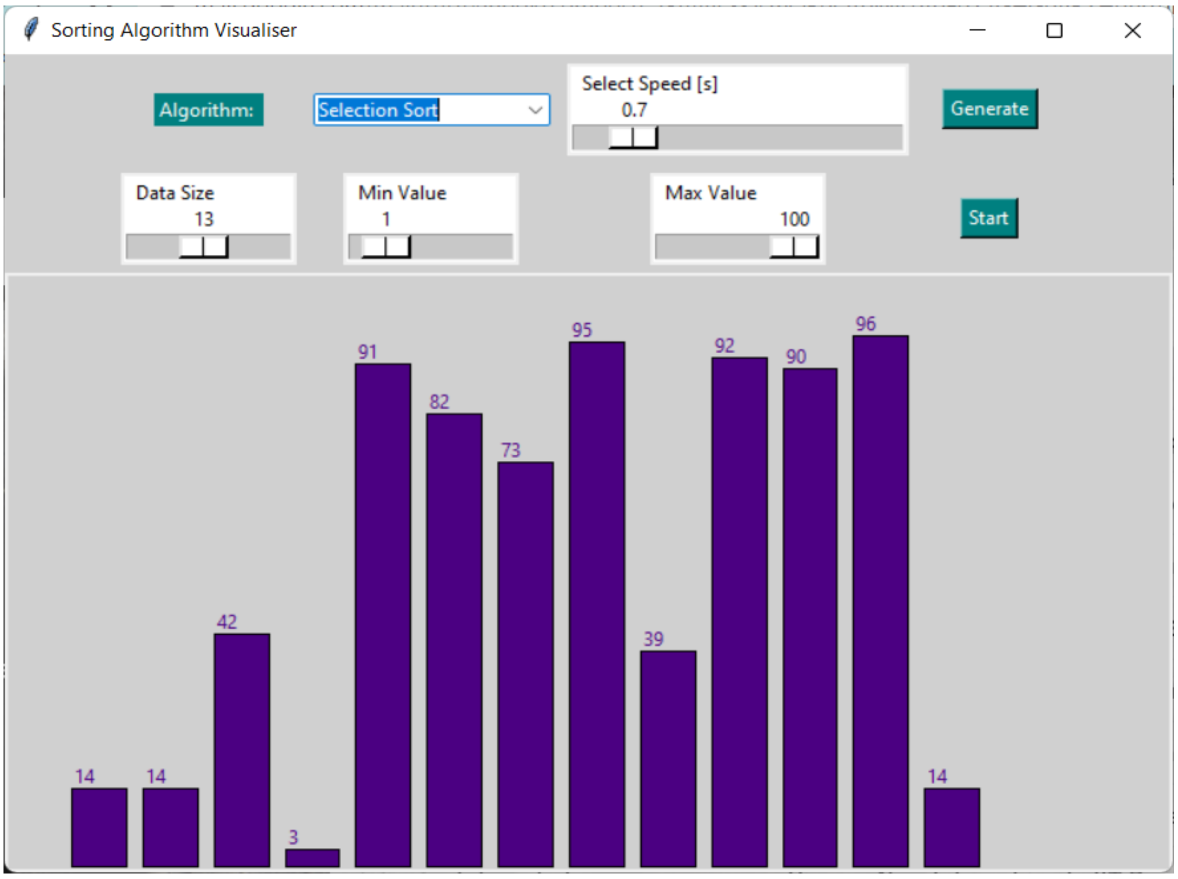


Fig.3.5.3(a): Selection Sort

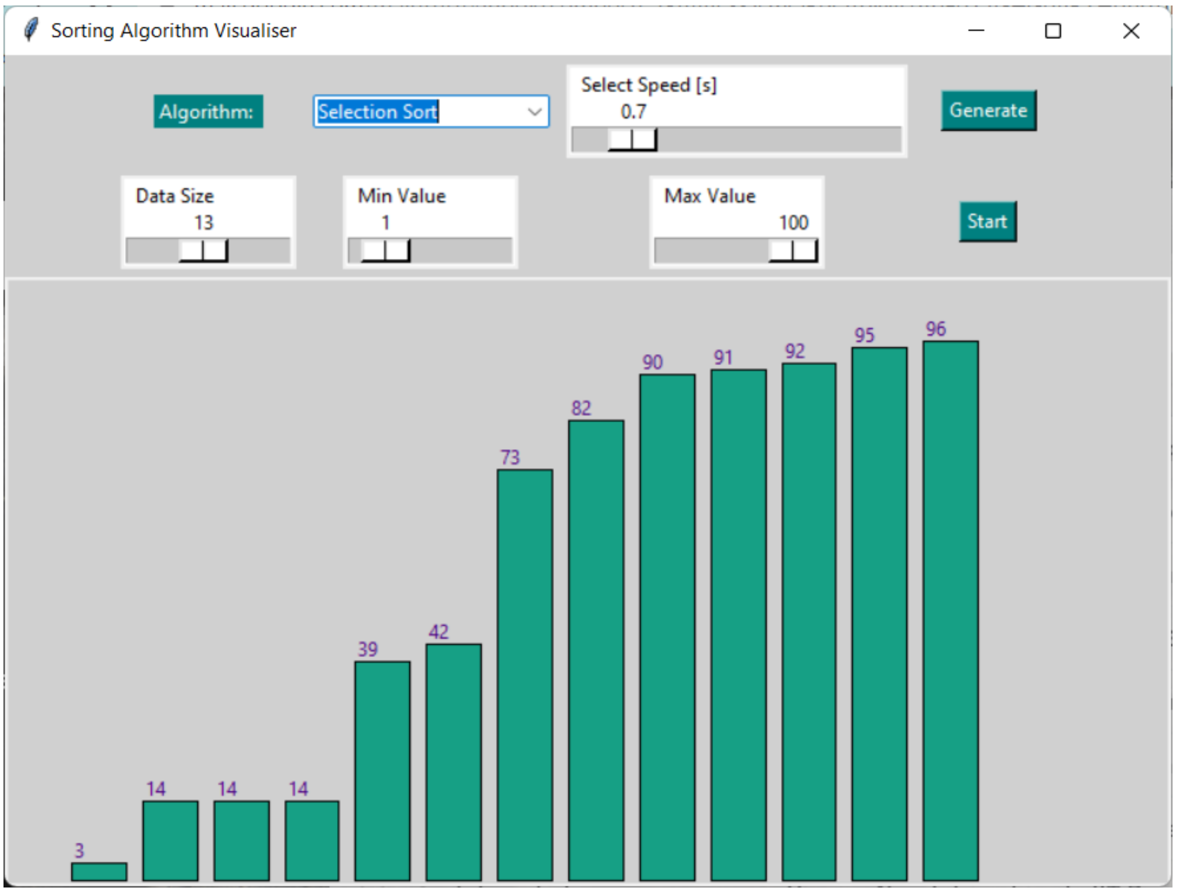
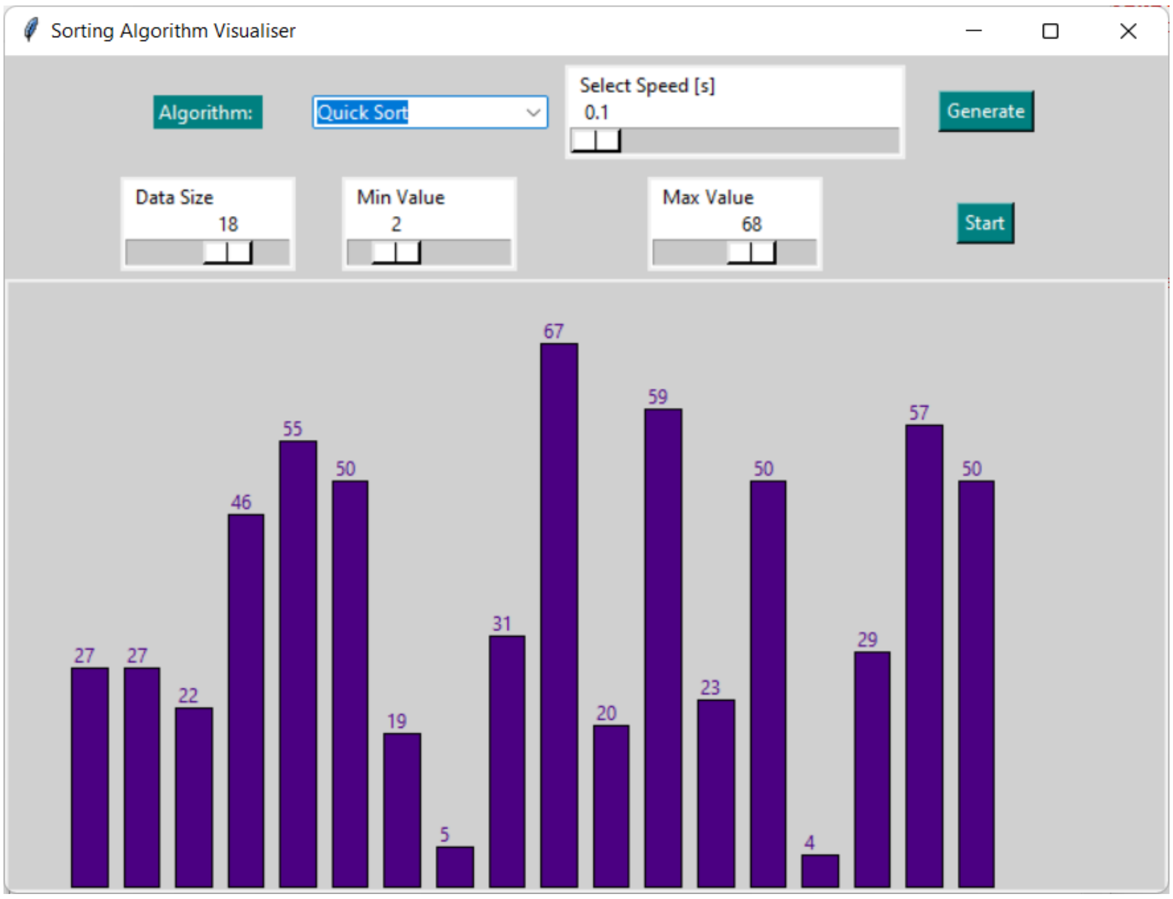


Fig.3.5.3(b): Selection Sort



Algorithm: Quick Sort

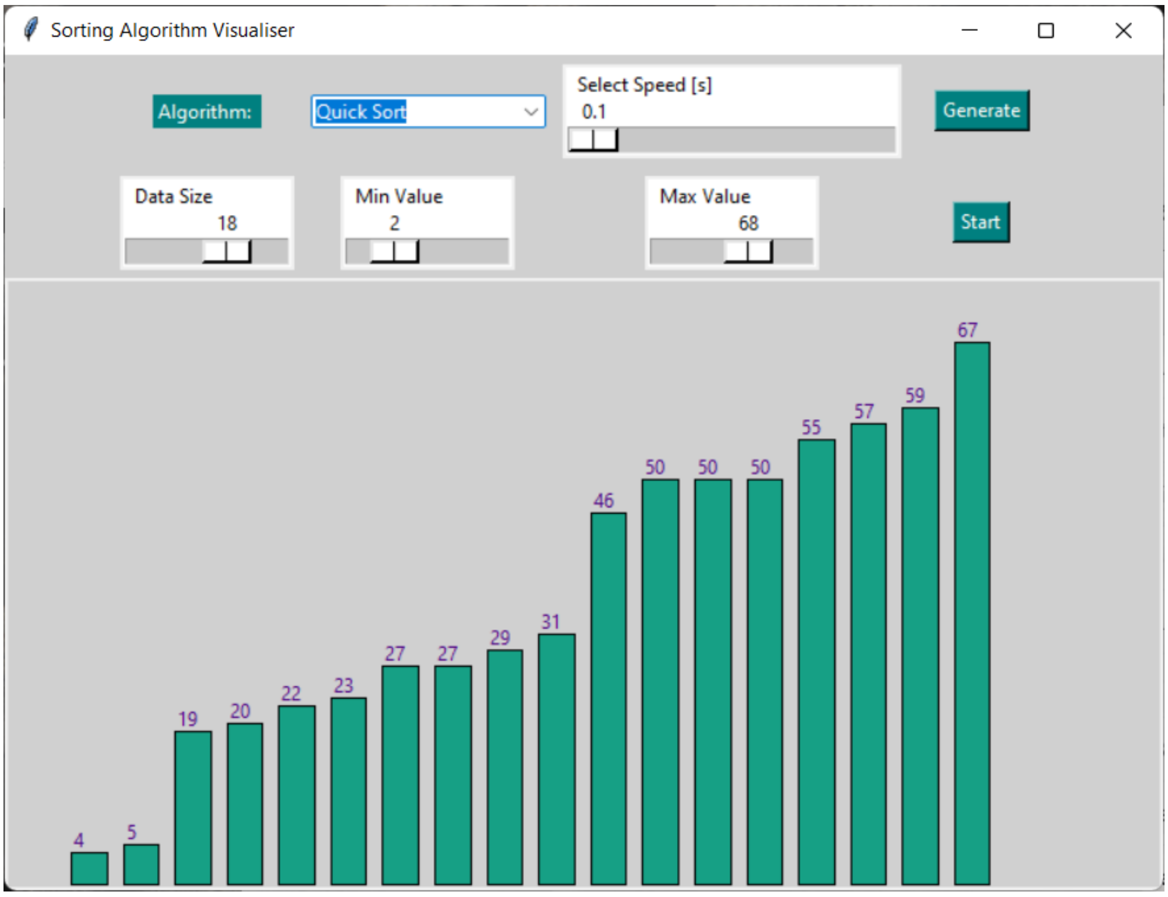
Select Speed [s] 0.1

Data Size 18

Min Value 2

Max Value 68

Fig.3.5.4(a): Quick Sort



Algorithm: Quick Sort

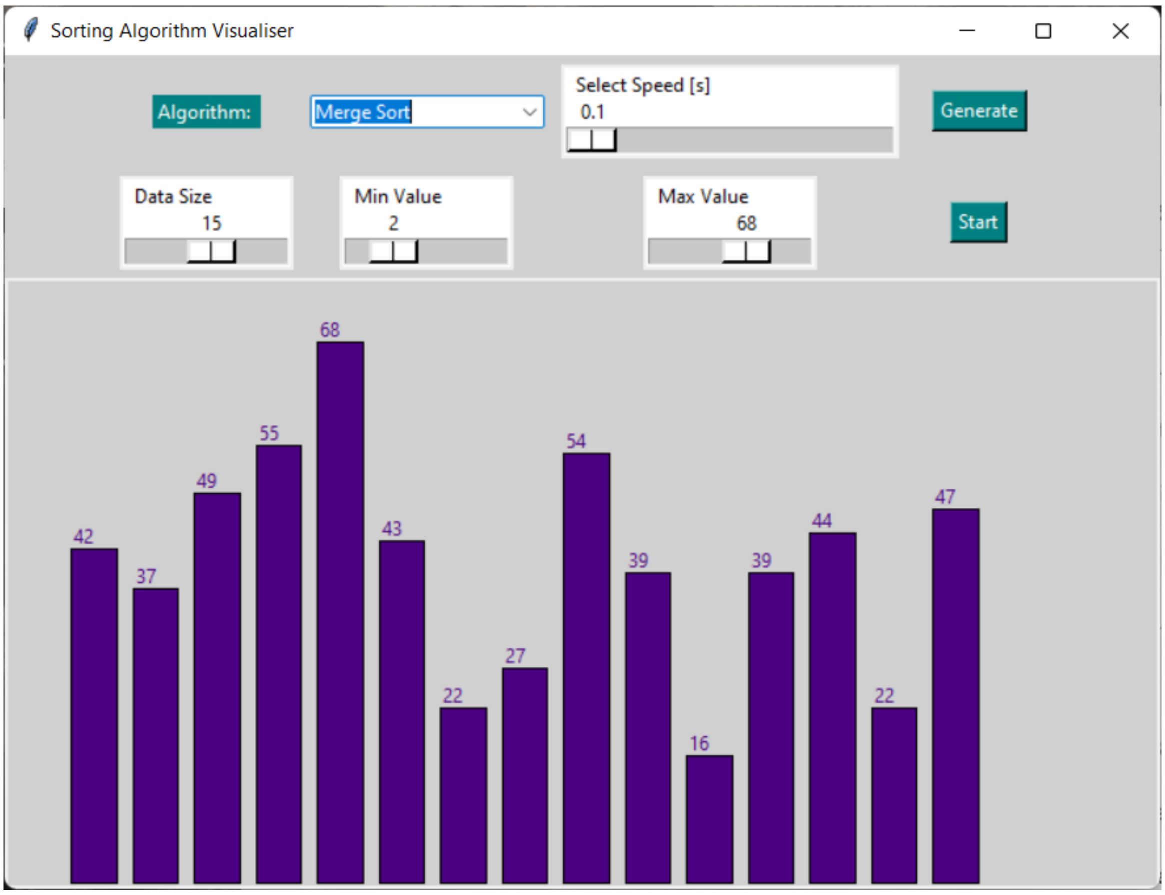
Select Speed [s] 0.1

Data Size 18

Min Value 2

Max Value 68

Fig.3.5.4(b): Quick Sort


Sorting Algorithm Visualiser

Algorithm: Merge Sort

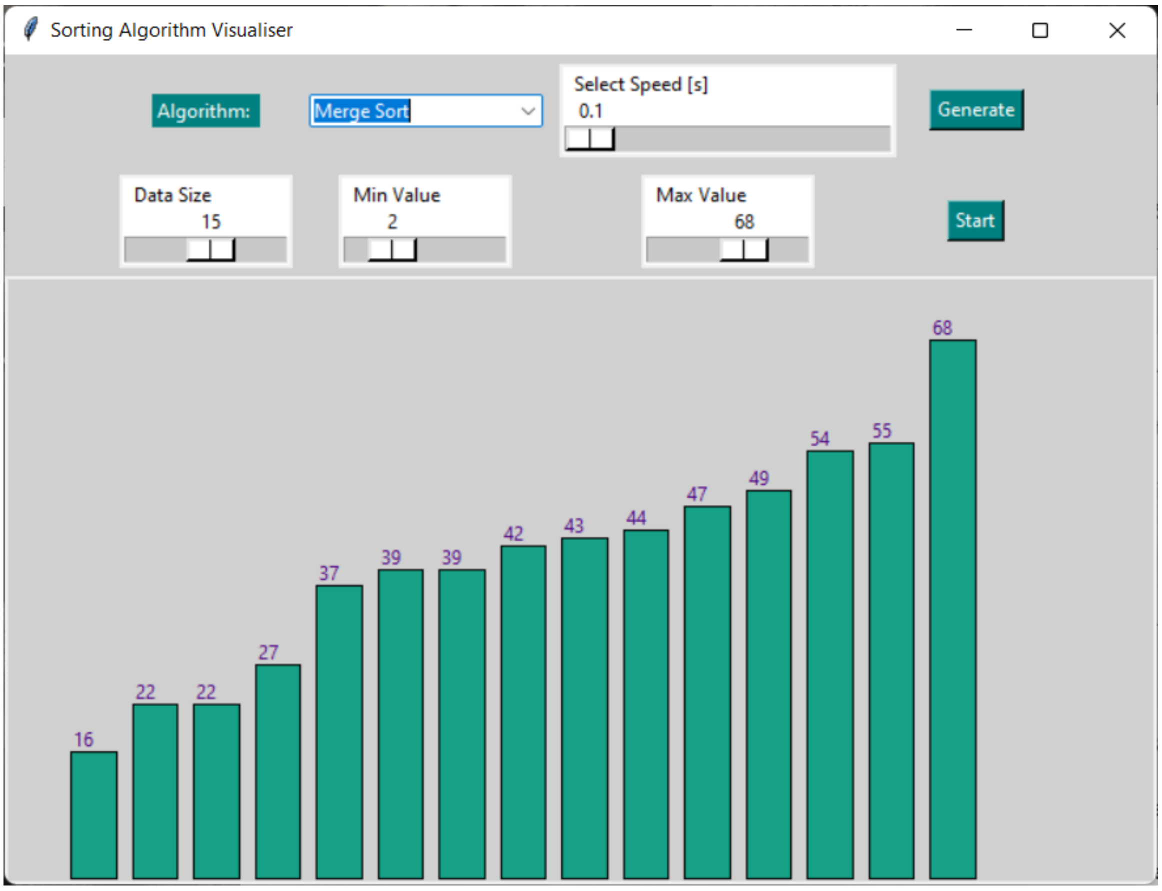
Select Speed [s] 0.1

Data Size 15

Min Value 2

Max Value 68

Fig.3.5.5(a): Merge Sort


Sorting Algorithm Visualiser

Algorithm: Merge Sort

Select Speed [s] 0.1

Data Size 15

Min Value 2

Max Value 68

Fig.3.5.5(b): Merge Sort

3.6 Conclusion and Future work

Conclusion:

With the inclusion of the proposed features, the algorithm visualizer would not only help in better visualization and retention of concepts by students, but also, enable the student's to have a gradual and smooth transition from school level to college level of education, hence it will enhance their knowledge and productivity.

Future Work:

In future a feature of calculating and displaying time complexity of each algorithm can be implemented. Also the numbers that are displayed on the bars randomly can be taken from the user itself.

References

[1] T. Bingmann. “The Sound of Sorting - ‘Audibilization’ and Visualization of Sorting Algorithms.” Panthemanet Weblog. Impressum, 22 May 2013. Web. 29 Mar. 2017. .

[2] Bubble-sort with Hungarian (”Cs´ang´o”) Folk Dance. Dir. K´atai Zolt´an and T´oth L´aszl´o. YouTube. Sapientia University, 29 Mar. 2011. Web. 29 Mar. 2017. .

[3] A. Kerren and J. T. Stasko. (2002) Chapter 1 Algorithm Animation. In: Diehl S.(eds) Software Visualization. Lecture Notes in Computer Science, vol 2269. Springer, Berlin, Heidelberg. .

[4] A. Moreno, E. Sutinen, R. Bednarik, and N. Myller. Conflictive animations as engaging learning tools. Proceedings of the Koli Calling ‘07 Proceedings of the Seventh Baltic Sea Conference on Computing Education Research - Volume 88, Koli ‘07 (Koli National Park, Finland), pages 203-206. .

[5] J. Stasko. Using Student-built Algorithm Animations As Learning Aids. Proceedings of the Twenty-eighth SIGCSE Technical Symposium on Computer Science Education. SIGCSE ‘97 (San Jose, California), pages 25-29. . 41

[6] J. Stasko, A. Badre, and C. Lewis. Do Algorithm Animations Assist Learning?: An Empirical Study and Analysis. Proceedings of the INTERACT ‘93 and CHI ‘93 Conference on Human Factors in Computing Systems, CHI-93 (Amsterdam, the Netherlands), pages 61-66. .