|  |
| --- |
|  |
|  |
| |  |  |  | | --- | --- | --- | | Anuvind M P | [Date] | [Course title] | |

Sort-search algorithm visualizer

**Guide : Dr. Divya Udayan**

**Group Members :**

**Anuvind M P (AM.EN.U4AIE22010)**

**Harishankar Binu Nair (AM.EN.U4AIE22023)**

**R S Harish Kumar (AM.EN.U4AIE22042)**

Amrita school of Computing, Amritapuri

**Abstract**

In the realm of computer science education, algorithm visualization serves as a powerful tool for enhancing understanding and intuition about fundamental computational processes. This project introduces a comprehensive algorithm visualization tool implemented using Pygame, Tkinter, and MySQL, designed to demonstrate sorting and search algorithms using user-provided input data.The tool utilizes Pygame for creating interactive visualizations of sorting algorithms and Tkinter for search algorithms, providing users with a dynamic platform to explore algorithm behavior. Additionally, MySQL is employed for data storage, enabling users to input their own arrays or graphs and observe how algorithms such as Quick Sort, Merge Sort, Breadth First Search, and Depth First Search operate on custom data sets. The visualizations generated by the tool provide real-time representations of algorithm execution, with user-provided data seamlessly integrated into the algorithmic process. This dynamic interaction enhances understanding of algorithmic concepts and fosters a deeper appreciation for algorithm efficiency and effectiveness.Our tool aims to be an educational resource, empowering users to explore algorithm behavior and enhance their computational thinking skills. Whether used for educational purposes or practical algorithm analysis, our visualization tool offers a hands-on approach to learning and understanding algorithms, inspiring curiosity and facilitating deeper insights into the algorithms that drive modern computing systems.Furthermore, the integration of Pygame and Tkinter libraries allows for a rich and interactive user experience. Users can interact with the visualizations, exploring different scenarios by adjusting array sizes, animation speeds, and algorithm parameters. This customization enables users to gain insights into algorithm behavior under varying conditions, enhancing their analytical skills and computational intuition.Additionally, the use of MySQL for data storage ensures scalability and efficiency in handling large input data sets. This feature enables users to work with complex data structures and visualize the performance of algorithms on real-world data, further enhancing the educational value and practical utility of our visualization tool.

**Introduction**

In the vast landscape of computer science, understanding the intricacies of algorithms is fundamental. This project introduces a comprehensive algorithm visualization tool that offers a hands-on approach to learning sorting and search algorithms.Bubble Sort, a simple yet inefficient sorting algorithm, demonstrates how it iterates through a list, compares adjacent elements, and swaps them if they are in the wrong order.Insertion Sort, known for its simplicity and efficiency on small lists, showcases how it builds the final sorted array one element at a time by inserting each unsorted element into its correct position.Selection Sort, another straightforward but inefficient algorithm, demonstrates how it repeatedly finds the minimum element from the unsorted part of the array and swaps it with the first unsorted element.Heap Sort, a comparison-based sorting algorithm, showcases how it uses a binary heap data structure to sort elements in ascending or descending order.Quick Sort, a highly efficient divide-and-conquer algorithm, demonstrates how it divides the array into smaller subarrays and then recursively sorts them.Merge Sort, another efficient divide-and-conquer algorithm, demonstrates how it divides the array into two halves, sorts them separately, and then merges them.Radix Sort, a non-comparative integer sorting algorithm, showcases how it sorts integers by grouping digits that share the same significant position and value.On the searching side, Breadth First Search (BFS), a fundamental graph traversal algorithm, showcases how it systematically explores a graph's vertices, starting from a selected node and moving through all of its neighbors at the current depth before moving on to the nodes at the next depth level.Depth First Search (DFS), another fundamental graph traversal algorithm, demonstrates its ability to traverse deep into a graph before backtracking.

Binary Search, a fast and efficient algorithm for finding a target value within a sorted array.

Linear Search, on the other hand, is a simple search algorithm that checks each element in a list sequentially until the target value is found or the list is exhausted.

Overall, this tool aims to provide a dynamic and interactive platform for exploring the behavior and efficiency of various sorting and search algorithms. Users can input their own arrays or graphs, observe how these algorithms operate on custom data sets, and gain a deeper understanding of their mechanics and efficiency.

**Requirements & Specifications**

1 page

**Project plan**

**Design strategy**

Both total 2 or 3 page

**Code snippets**

3 or 4 pages

**Summary**

1 page