**University of Texas Arlington**

**Department of Computer Science Engineering**

**CSE 5311 Design & Analysis of Algorithms**

**Spring 2022**

**Project-1**

**Sorting Algorithms**

**Instructor: Dr.Negin Fraidouni**

**Abstract:**

A Sorting Algorithm is used to rearrange a given array or list elements according to a comparison operator on the elements. The comparison operator is used to decide the new order of element in the respective data structure. In Practice, Sorting plays a very important role as many applications uses the sorted data. This project is about different sorting algorithms and to show how these algorithms performs if the size of the inputs vary . I have tested all the algorithms with data of different sizes. I have also compared their performance and tabulated them. The user first enters an unsorted data and, on a button, click, the data would be sorted based on the algorithm that the user selects and the execution time of the data using the selected algorithm is displayed.

For this, I used HTML, CSS and typescript to fulfill the requirement.

**Project 1:**

This project contains 7 sorting algorithms. The sorting algorithms used in this project are:

Sorting Algorithms used:

1. Insertion Sort
2. Selection Sort
3. Bubble Sort
4. Merge Sort
5. Heap Sort
6. Quick Sort
7. 3 Median Quick Sort

**Implementation**:

I decided to do the project in Angular as it works on the client side and single page application it is fast. The UI developed in angular is interactive and flexible. It also has the ability of JavaScript, but the language used is called typescript so normal computation is easy to execute. The representation of the graph is better using chartJS.

**Simple Readme:**

I have coded the project in HTML, CSS, and typescript. The user must follow the steps to get the output.

1. Go to the project code folder open the command prompt and type ‘ng serve’.
2. The user must go to the browser and type ‘localhost:4200’ or else use ‘ng serve –open’ which will open into your default browser.
3. If the user wants to sort the data, then the user needs to enter the array data.
4. Another option for user is to click on the randomize button.
5. It will then give option to enter the size of the array and select the algorithm.
6. Then user must select the algorithm and click on calculate.
7. The output will show the sorted array if the array size is 30 or less than 30. It will also show the time taken for sorting.

**Programming Language:**

1. For UI : HTML, CSS, CanvasJS
2. For the Scripting: Typescript

Data Structures mostly used were JSON and Array 95% of my code has array has data structure but for CanvasJS I used JSON as my data structure to store the values.

**Sample Inputs:**

Sample Input Size:n=15

12 15 10 4 7 8 0 3 5 14 1 2 6 11 13

or

12,15,10,4,7,8,0,3,5,14,1,2,6,11,13

Above shown array was used to test the normal functionality program.

Large input size array of 10,000 was used for comparing the performance. The array was generated randomly by using random option.

Average input size array of 1000 was used for comparing the performance. The array was generated randomly by using random option.

Small input size array of 100 was used for comparing the performance. The array was generated randomly by using random option.

**Data Comparison Table(Runtime vs Size of Input):**

**Table

Description automatically generated**

The above table describes the performance of each sorting algorithm for different sizes of inputs.

**Data Visualization:**

**Chart, line chart

Description automatically generated**

The graph shows the output when the array length is small N=500. According to my observation the quick is the fastest and the sorting time for merge and heap are approximate same. The algorithm that takes maximum time is Bubble sort.

Chart, line chart

Description automatically generated

The diagram shows the performance when the input size is increased to 1000 that is average input size. This shows that the merge sort performance improves when dataset is big, or the performance of heap sort is getting affected by the size of the input. The quick sort is the best among all algorithms.

Chart, line chart

Description automatically generated

This is the representation of all the data input, the input is increased to the 10000 that is the largest the most efficient algorithm is quick sort it gives best performance for all the input size the next algorithm is merge and heap sort. The worst is bubble sort.