Logo, company name

Description automatically generated

**The University of Texas at Arlington**

**DESIGN & ANALYSIS OF ALGORITHMS**

**(CSE 5311)**

**PROJECT 1**

**Team Member:-**

1. Pathik Patel

UTA ID :- 1001846320

1. Kuldip Rameshbhai Savaliya

UTA ID :- 1001832000

**Description of work done by each team member.**

Pathik have worked on insetion sort, merge sort and generates array of various sizes and I kuldip worked on quick sort and time taken to sort the data. We both equally participate in project 1.

1. **List of sites/sources referred in the Project.**
   * + Geeks for geeks for the Algorithm and Knowledge.
     + Tutorials Point for Sorting Method and Concept.

1. **Time Complexity of the algorithm**
2. **Insertion Sort Time Complexity**

Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands.

* Best case: **O(n)**
* Average case: **O(n²)**
* Worst case: **O(n²)** (reverse sorted array)

1. **Merge Sort Time Complexity**

Merge Sort is a Divide and Conquer algorithm. It divides the input array into two halves, calls itself for the two halves, and then merges the two sorted halves.

* Best case: **O(n\*log\*n)**
* Average case: **O(n\*log\*n)**
* Worst case: **O(n\*log\*n)**

1. **Quick Sort Time Complexity**

Quick sort is a highly efficient sorting algorithm and is based on partitioning of array of data into smaller arrays.

* Best case: **O(n\*log\*n)**
* Average case: **O(n\*log\*n)**
* Worst case: **O(n²)**

1. **Experimental Results**

**Insertion sort:**

Duration for insertion sorting array of 20 numbers: 7.120500000024066e-05 seconds

Duration for insertion sorting array of 100 numbers: 0.0006489979999999562 seconds

Duration for insertion sorting array of 1000 numbers: 0.07550644999999978 seconds

Duration for insertion sorting array of 4000 numbers: 1.5674032070000004 seconds

**Merge Sort:**

Duration for merge sorting array of 20 numbers: 9.83450000000552e-05 seconds

Duration for merge sorting array of 100 numbers: 0.0003869069999999919 seconds

Duration for merge sorting array of 1000 numbers: 0.008671297999999883 seconds

Duration for merge sorting array of 4000 numbers: 0.030819139999999967 seconds

**Quick sort:**

Duration for quick sorting array of 20 numbers: 8.132900000012988e-05 seconds

Duration for quick sorting array of 100 numbers: 0.00032774799999990556 seconds

Duration for quick sorting array of 1000 numbers: 0.002993164000000048 seconds

Duration for quick sorting array of 4000 numbers: 0.02029512500000008 seconds

1. **Differences between the Experimental and Theoretical results.**

In above task, we can see the experimental results of all three algorithms and also can see the duration for all sorting algorithms. Theoretical results could convey the range of time algorithm would take to excute using the notations given above.

1. **Compare and contrast the results between the three sorting algorithms and time taken to sort the 4 arrays.**

From above results, we can see that quick sort perform better compared to insertion sort and merge sort. Insertion sort takes more time to sort the data as compared to merge sort and quick sort.

1. **List of 4 arrays used in the experiments.**

We used these four files for our project 1.

arr20.txt

arr100.txt

arr1000.txt

arr4000.txt

1. **Honor code**

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and uphold the spirit of the Honor Code

I will not participate in any form of cheating/sharing the questions/solutions.

**Pathik Patel**

**Kuldip Rameshbhai Savaliya**