ITCS_6114 Algorithms and Data Structures

Project I

Name: Aditi Helekar

Student ID: 800966667

Report for Project I:

Approach:

For improving the performance of Quicksort we have introduced Insertion sort for number of elements less than 10.

Description of Program:

1. I have read a comma separated sequence of numbers from input file and added it to an array of size 100000
2. The CUTOFF for Insertion sort is 10 This cutoff gives the optimum number of key comparisons for variety of input sequences.
3. The output of sorted sequence is written into a text file with comma separated values.

Supporting Data:

1.
```
Enter file name: ni.txt
Total number of elements in input sequence: 8
Number of key Comparisons using Median: 8
Number of key Comparisons using Randomized Pivot Element: 7
Number of key Comparisons using Median with Insertion Sort: 13
Number of key Comparisons using Randomized Pivot Element with Insertion Sort: 13
```

2.
```
Enter file name: Array.txt
Total number of elements in input sequence: 100
Number of key Comparisons using Median: 312
Number of key Comparisons using Randomized Pivot Element: 656
Number of key Comparisons using Median with Insertion Sort: 76
Number of key Comparisons using Randomized Pivot Element with Insertion Sort: 35
```

3.
```
Enter file name: 100.txt
Total number of elements in input sequence: 100
Number of key Comparisons using Median: 312
Number of key Comparisons using Randomized Pivot Element: 656
Number of key Comparisons using Median with Insertion Sort: 76
Number of key Comparisons using Randomized Pivot Element with Insertion Sort: 35
```

4.

```
Enter file name: 10000.txt
Total number of elements in input sequence: 10000
Number of key Comparisons using Median: 48166
Number of key Comparisons using Randomized Pivot Element: 150191
Number of key Comparisons using Median with Insertion Sort: 8845
Number of key Comparisons using Randomized Pivot Element with Insertion Sort: 2857
```

5.

```
Enter file name: 100000.txt
Total number of elements in input sequence: 100000
Number of key Comparisons using Median: 566545
Number of key Comparisons using Randomized Pivot Element: 1987810
Number of key Comparisons using Median with Insertion Sort: 82809
Number of key Comparisons using Randomized Pivot Element with Insertion Sort: 99062
```

Conclusion:

1. According to the observation, I found that the number of key comparisons depends is greater in Median of 3 implementations than Randomized Pivot element implementation with Insertion Sort.
2. For small input sequence, algorithm Randomized Pivot gives better results than any algorithm with insertion sort.
3. As we increase the size of input such as 100 the algorithms using insertion Sort work good. There is a considerable difference between the number of key comparisons.
4. For inputs in range 10000, Randomized Pivot element implementation with Insertion Sort gives better results.
5. But for higher input sequence size, Median of 3 with Insertion Sort performs well.

The selection of appropriate Sorting Algorithm using QuickSort solely depends on size of input. For smaller input algorithm without Insertion sort are suitable and for larger input algorithm with Insertion sort are suitable. But Overall Randomized Pivot with Insertion Sort gave consistent performance.