Program Structures and Algorithms Spring 2023

(SEC –03)

NAME: Tiruchunapalli Venkata Abhishek

NUID: 002770934

Assignment – 3

1. To implement three methods namely, repeat, getClock and toMilliSec according to business logic and requirement provided in assignment description.
2. Implement Insertion Sort by applying the algorithm by utilizing helper methods provided in the code.
3. Test the sorting algorithm implemented in part 2 under various scenarios such as using Random array, Ordered array , Partial-Ordered Array and Reverse-Ordered array as inputs.

Use Doubling method to compute values and draw conclusions based on observations

**Relationship Conclusion:**

When analysing the performance of the insertion sort algorithm, it was observed that the input array's order had a significant impact on the algorithm's time complexity. Here are the conclusions that were drawn:

1. Arrays that are already ordered resulted in the shortest processing time, with a performance that closely resembles a constant-time function.
2. Partially ordered arrays took longer to process than ordered arrays, and the processing time increased at an exponential rate as the size of the array increased. This performance is similar to that of a 𝑛(log 𝑛)^2 function, with a shallower slope compared to the random array case.
3. Random arrays took even longer to process, with processing time increasing at a rate similar to that of 𝑛(log 𝑛)^2, but with a steeper slope compared to the partially ordered arrays.
4. Reverse ordered arrays took the longest to process, with processing time increasing at an exponential rate similar to 2𝑛(log 𝑛)^2, and with the steepest slope among all the cases observed.

These findings highlight the importance of the order of the input array in determining the performance of the insertion sort algorithm.

Top of Form

Bottom of Form

1. T v N Graph with N values ranging from 10 to 160 calculated for pow(10,4) runs.

Chart, line chart

Description automatically generated

2. Unit tests for TimerTest.java

Text

Description automatically generated

2. Unit tests for BenchmarkTest.java

A screenshot of a computer

Description automatically generated with medium confidence

3. Unit tests for InsertionSortTest.java

Text

Description automatically generated

Code Snippets:

Part 1:

Timer.java: Repeat function

Text

Description automatically generated

Timer.java: Clock and Millisec

Text

Description automatically generated

Part 2:

InsertionSort.java:

A screenshot of a computer

Description automatically generated with medium confidence

Part-3: InsertionSortTest.java: code for all four types of input arrays.

Text

Description automatically generated

Part-4: Helper functions

Text

Description automatically generated