Assignment – 2

Tasks Required:

To Benchmark Insertion sort and draw conclusions for reverse, ordered, partially ordered and random ordered array.

Part 1: To Implement 3 methods in Timer Class.

public <T, U> double repeat(int n, Supplier<T> supplier, Function<T, U> function, UnaryOperator<T> preFunction, Consumer<U> postFunction) {  
// TO BE IMPLEMENTED  
}

private static long getClock() {  
 // TO BE IMPLEMENTED  
}

private static double toMillisecs(long ticks) {  
 // TO BE IMPLEMENTED  
}

Part 2: Implementation of Insertion sort

Part 3: Implementing a main program as an entry point for the benchmarking process

Output Obtained:

The output obtained is after running the benchmark over 10 times per doubling input starting with 20 for Reversed, Ordered, Partially ordered and Random ordered arrays

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Array Length** | **Reversed order** | **Ordered** | **Partially ordered** | **Random order** |
| 20 | 0.01512 | 0.00119 | 0.00721 | 0.00128 |
| 40 | 0.0112 | 0.00155 | 0.00838 | 0.00187 |
| 80 | 0.03575 | 0.00175 | 0.01911 | 0.00213 |
| 160 | 0.12322 | 0.00298 | 0.02135 | 0.00913 |
| 320 | 0.23651 | 0.00572 | 0.06884 | 0.02863 |
| 640 | 1.41593 | 0.01008 | 0.15305 | 0.05009 |
| 1280 | 0.34849 | 0.01957 | 0.27203 | 0.17958 |
| 2560 | 1.41989 | 0.03903 | 0.50182 | 0.70052 |
| 5120 | 6.15856 | 0.07075 | 1.09717 | 2.63182 |
| 10240 | 23.99875 | 0.15919 | 4.54772 | 12.69387 |

A screenshot of a computer

Description automatically generated with medium confidence

Test case Results:

Benchmark Test Result:

A screenshot of a computer

Description automatically generated with medium confidence

InsertionSort Test Result:

A screenshot of a computer

Description automatically generated with medium confidence

Timer Test Results:

A screenshot of a computer

Description automatically generated with medium confidence

Observations and Conclusions:

Plotting a line graph from the obtained outputs for the differently sorted inputs with mean sorting times on the y-axis and array length on the x-axis.

* We observe that with increase in array length the mean time taken to sort also increases, irrespective of the sorting of the input.

**Meantime to sort is proportional to array length.**

* From the same graph we can also observe that the sorting of the input array affects the meantime to sort in the below way.

Meantime for sorting of

**Reverse ordered array > Random ordered > Partially ordered > Ordered array**

* We can also observe that the time complexity for sorting varies from

**O (n2 )** for the worst case i.e. reverse ordered input to **O(n)** which is the best case scenario i.e. ordered input with partially ordered and random lying in between these two limits.