**INFO6205: Program Structure and Algorithms**

**Assignment 2 – Benchmark Test**

**By**

**Ajay Mohandas**

**NEU ID: 001426741**

Contents

[Benchmark.java 1](#_Toc536826651)

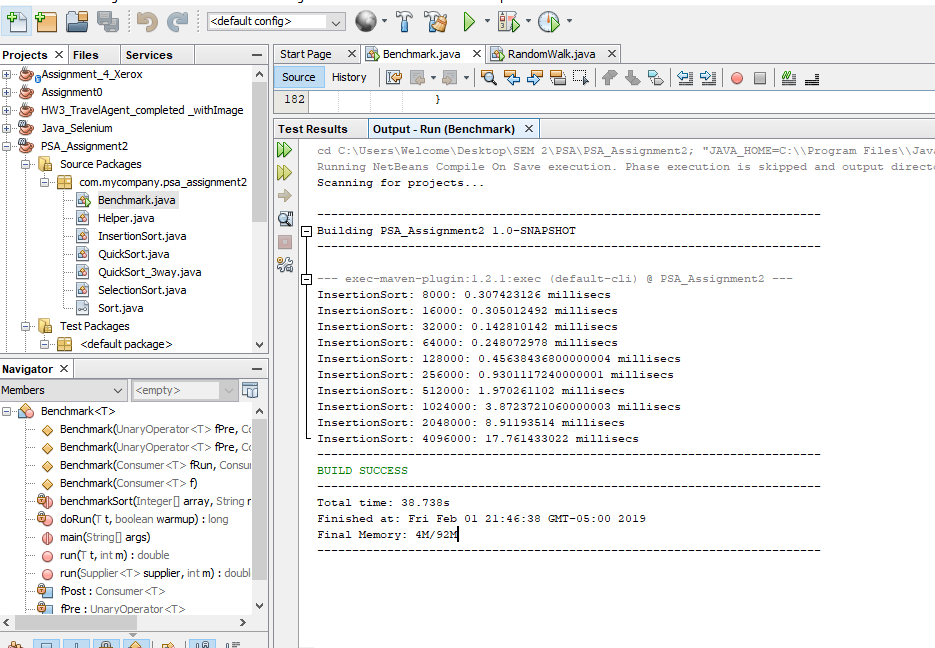
[BenchmarkTest.java 2](#_Toc536826652)

[Observation 2](#_Toc536826653)

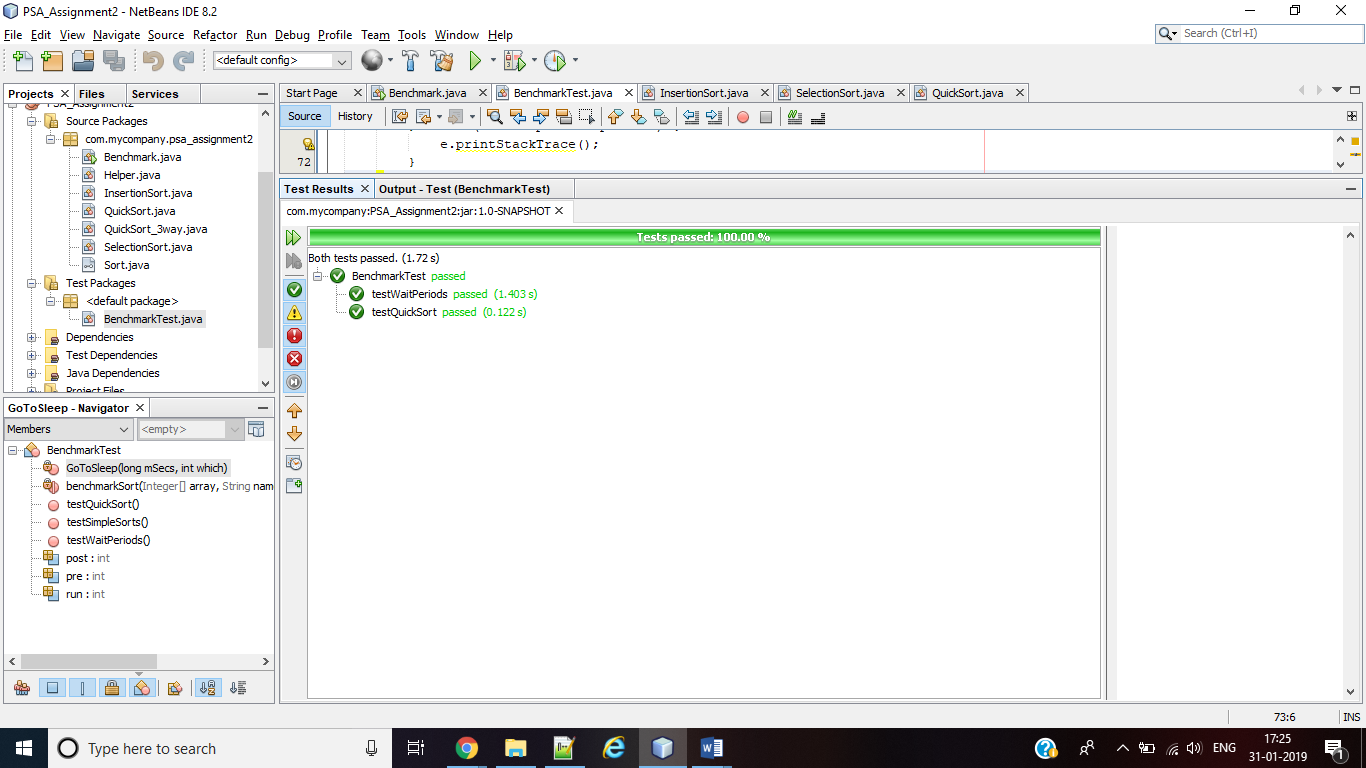
[Analysis 2](#_Toc536826654)

[Conclusion 2](#_Toc536826655)

# Benchmark.java



# BenchmarkTest.java



# Observation

1. X axis is the number of steps taken and Y axis is the running time for the insertion sort for each scenario
2. When array has **random inputs** the time required for sorting the array number of steps given initially falls and then increases extensively
3. When array has inputs which already **ordered** the time required for sorting the array number of steps given initially falls and then increases gradually as number of steps increases
4. When array has inputs in **reverse ordered** the time required for sorting the array number of steps given initially falls and then increases extensively
5. When we have **partially ordered** array then running time for sorting falls steeply and then increases gradually as number steps increases

# Analysis

1. Collaborating the results of all four scenarios the running time of insertion sort for any input of arrays has a fall and then increases as number of steps increases.
2. The growth of ordered, reverse ordered and random array is almost similar but the growth of partially ordered array has a different growth than others.
3. The growth of the curve is polynomial raise to power of 2 i.e function of time with respect ot number of steps taken is y = 1E-09x2 + 0.0001x + 0.9812, with an error value of R² = 0.9952

# Conclusion

1. The algorithm works more efficiently for ordered array, random array and reverse ordered array in increasing order respectively.
2. The running time for partially ordered array will depend on the number of elements which are ordered and then running time increases as random inputs are present in array.
3. Since we are using Arrays.sort which uses 3\_way quicksort algorithm for sorting inputs the order of growth is **O(n\*log(n))** for the average case scenarios