Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre



Algorithm and Complexity (COMP 314)

Lab 2 Report

Submitted to:

Dr. Rajani Chulyadyo

Department of Computer Science and Engineering

Submitted By:

Mani Dumaru

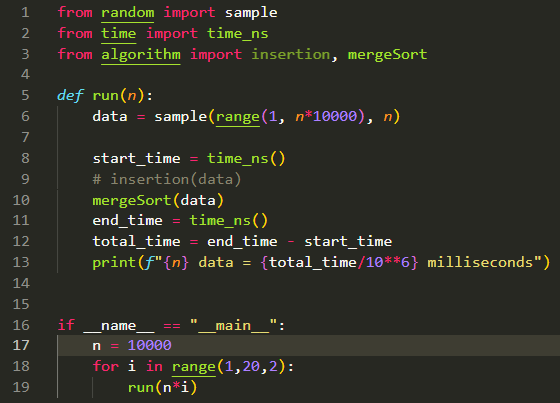
Roll no.: 15

CE-2019 3rd year/2nd semester

Submission Date: 25th April, 2023

Implementation, Testing and Performance of Insertion and Merge sort.

Code for performance testing:

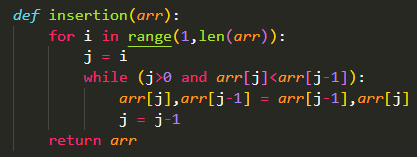


‘n’ number of random data is generated in line 6 of above code. A timer is started in line 8 and then the randomly generated array is then passed for sorting to respective sorting algorithms (insertion and merge). When completion of sorting, the timer is then stopped. Total amount of time taken to sort ‘n’ number of data is then calculated as the end\_time – start\_time and is printed in milliseconds.

1. **Insertion Sorting:**

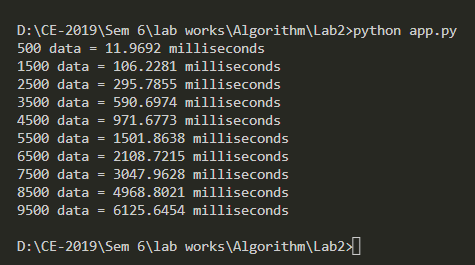
In this algorithm, the array is divided into two parts, sorted and unsorted. The key element of unsorted array is then placed into the correct position of the sorted part of the array.

*Algorithm:*



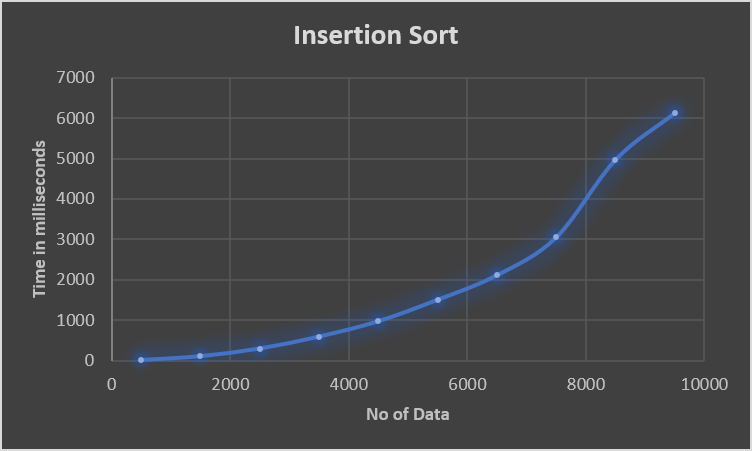
Algorithm for insertion Sorting

*Performance:*



*Graph for Insertion sort:*

With the data given by performance testing, following graph is plotted.

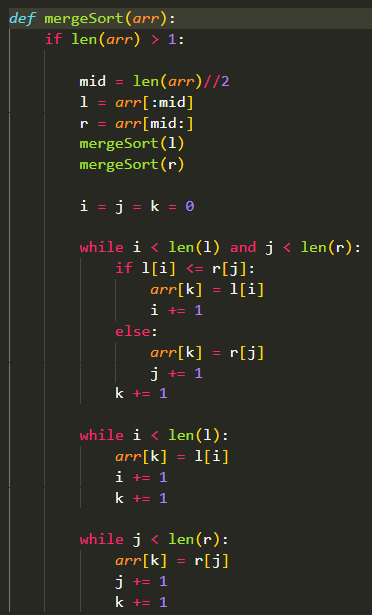


From the Graph, it is clear that the time complexity for insertion sorting algorithm is O(n2)

1. **Merge Sorting**

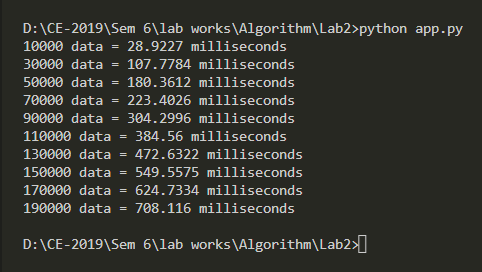
This algorithm of sorting data is based on divide and conquer approach. An array is divided into n sub arrays where n is the length of array so that each sub array contains only one element. Then the divided sub arrays are merged back together in sorted order. Such algorithm is called merge sorting.

*Algorithm:*



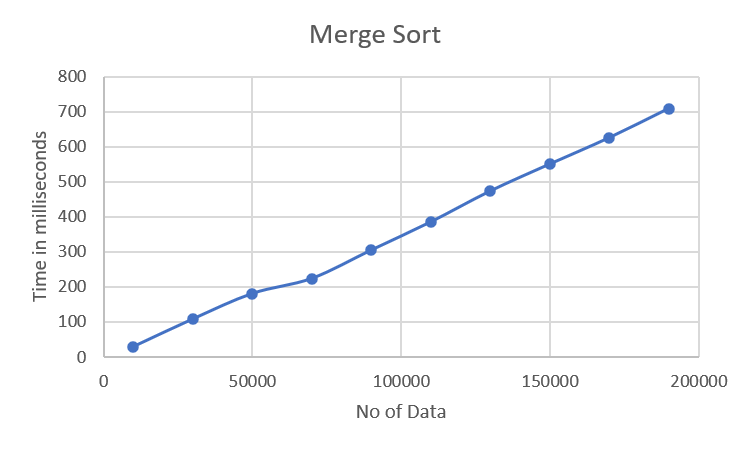
Algorithm for Merge Sort

*Performance:*



*Graph for Merge Sort:*

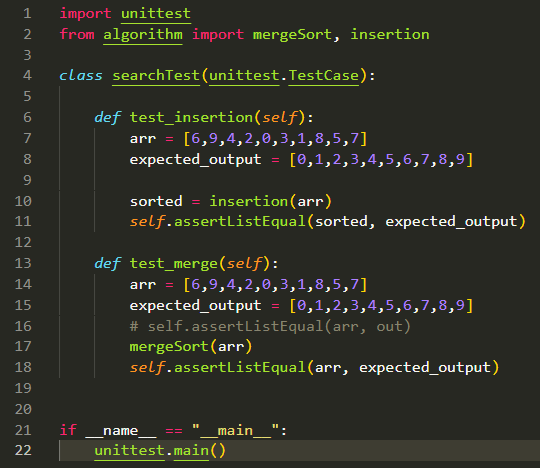
With the data given by performance test on merge sort, following graph is plotted.



From this graph, we can see that the time of complexity of Merge sort seems to be O(nlogn)

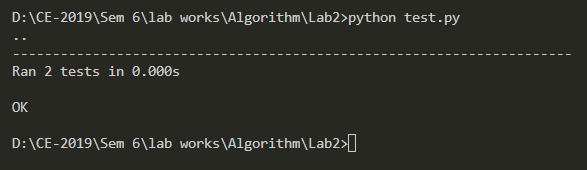
Testing the algorithm:

Code for testing insertion and merge sorting algorithm.



A random array along with another array with its sorted result is created. The random array is then passed to the sorting algorithm. After it has been sorted by the algorithm, it is checked if the result provided by the algorithm matches the expected output. If they match, the test is successful otherwise, the test is failed.

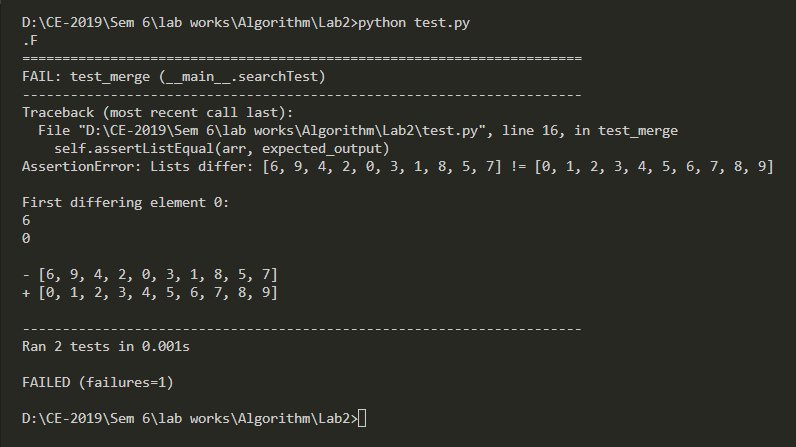
**Successful Test Output:**





In this test, in line number 16, a test is done before the array is passed to the sorting algorithm. This test should fail since, the array and expected result does not match.

**Failed Test Output:**

****

Conclusion:

Hence, Insertion and Merge sort algorithms were implemented. We checked if the algorithms were correct. Also, their performance was tested based on how much time they took to sort ‘n’ amount of data. With the results we got, merge sort algorithm is clearly a better algorithm for sorting as it took way lesser time to sort huge number of data than insertion sorting did. Time complexity for insertion sort was found to be O(n2) while that of merge of sort was found to be O(nlogn).