# ISE315 HOMEWORK 1

150130208

Omer Dogan

### About my code

My program asks to user the input size and after that it writes running times in a .csv file. (It calculates mergesort and bubblesort in one step. In mergesort, program creates an another sorted array. So bubblesort will sort unsorted array.)

Tested in

|  |
| --- |
| g++ version: 7.2.0 (ubuntu 16.04)  g++ version: 4.9.2 (windows 10)  IDE: Eclipse c++ |

Running Commands

|  |
| --- |
| g++ -c -std=c++11 main.cpp -o main.o  g++ main.o -o main.exe  ./main.exe |

# Asymptotic Upper Bound And Running Time

Mergesort is a divide and conquer algorithm which means the input size will be halved in each iteration. The entire input will be iterated thought, and each iteration must occur O(log(n)) times. n items iterated log(n) times and it gives us O(nlog(n)) as an upper bound.

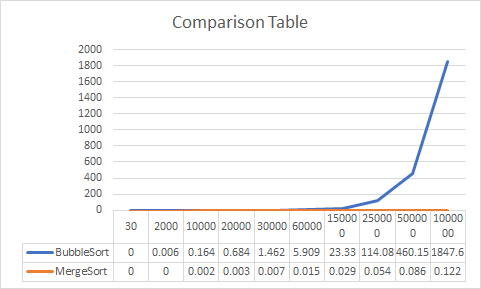
Also, The worst case for Bubble Sort is in the case of the array is already sorted in descending order. In this case, in first iteration it have to make comparison n element, then n-1 and so on. Until the last iteration.

For BubbleSort;

n + (n-1) + (n-2) + (n-3) …. + 1 = [n \* (n+1)] / 2 = n^2/2 + n/2 which means O(n^2)

My outputs for these two algorithms.

|  |  |
| --- | --- |
|  |  |



# Which One is Better?

Bubblesort is a slow alogrithm compared with the merge sort. However, bubblesort should be more effective than merge sort in the case which you have limited memory usage. Because mergesort requires a second array with the same size as the array to be sorted, that’s why it doubles the memory requirements.

Best case scenario for the bubblesort and mergesort if they sort sequential data which is nearly sorted. For a nearly sorted data, bubblesort requires at least 2 pass though the data, however it takes O(n) times. For this reason bubblesort is an adaptive algorithm.

### Extra

Also you can check my map implementation for this homework

<https://github.com/Omerdogan3/Distances-to-Istanbul>

Result

<https://distances-omerdogan3.c9users.io/index.html>