Week 2

Insertion sort and Merge sort

Preliminary

Lecturing on insertion sort and mergesort algorithms

Workshop

PROBLEM:
Write a program that sorts the list of input numbers.

INPUT: a sequence of n numbers. Consecutive numbers are separated by a space

OUTPUT: the list of the n numbers, sorted into *monotonically increasing order*.

Materials:

- SortingTest.zip : the test cases for measuring running time
- Inssort.py : an incomplete insertion sort program
- mergesort.py : an incomplete merge sort program

1) Complete the provided inssort.py program so that it utilizes *insertion sort* to sort the input data.
    - Verify the correctness with a few simple inputs.

2) Add running time recording code (given in the worksheet 1, last week). Then, measure the running time of your program on the provided test cases.
    - Project the running time increase with the input size, n. What do you conclude as the upperbound of the insertion sort running time ?

$$T(n) = O(\_\_\_\_\_)$$

3) Complete the mergesort.py program so that it utilizes merge sort to sort the input data. The merge function is already given in the program.
    - Verify the correctness with a few simple inputs.

4) Add running time recording code. Then, measure the running time of your program on the provided test cases.
    - The running time of merge sort, in any case, is $O(nlgn)$. Does the running time of your program on the provided test cases agree with the theoretical upperbound?

5) Is there an advantage of insertion sort over merge sort?