**Algorithms**

**Algorithm Sorting**

There are many different types of data sorting.

Two important ones:

Bubble Sort

A bubble rises to the top.

1. Step through the list

2. Swaps them if they are the wrong order

* Each step through the list is a ‘Pass’.

There are **n** items to sort.

**n – 1** pairs.

After the first sort, the largest value is at the bottom.

There are **n-1** items left to sort.

**n – 2** pairs.

At end, 0 pairs left to sort.

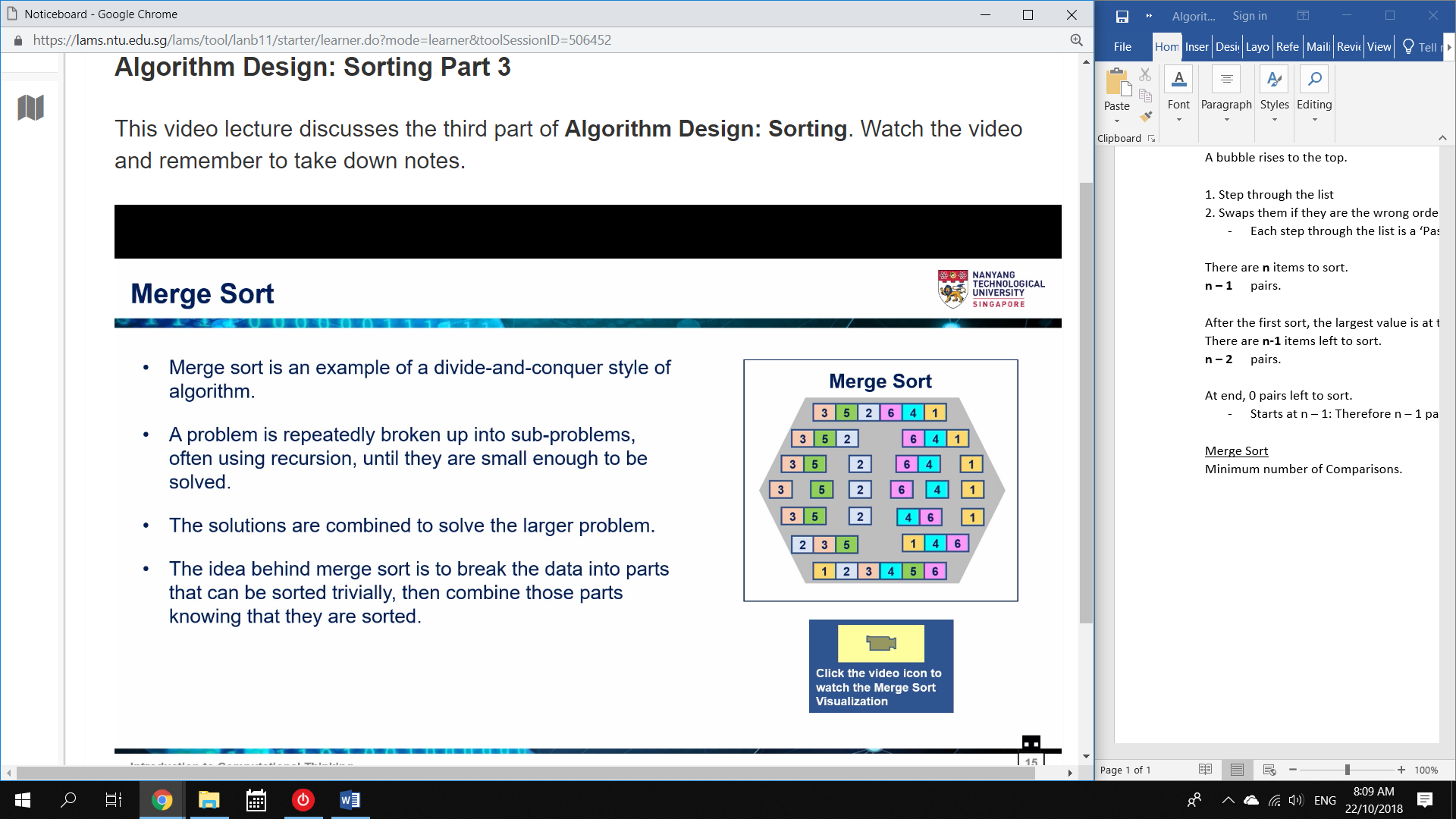
* Starts at n – 1: Therefore n – 1 passes total.

Merge Sort

Minimum number of Comparisons.

* Breaks data using Recursion
* Solves Trivial Problems
* Combines to solve larger proble.

This is a Divide-and-Conquer method.



SPLIT

1. Split the list.

* For odd-numbered lists, floor the first list.

2. Repeat till there is only One Element in any List.

You now have two ‘sorted’ lists.

SORT

1. Compare the first Element of both lists.

2. Place the Smaller Element in a new List, ‘Sorted Items’.

3. Repeat with a new Element from the List

* which the smaller element was from

4. When one list is Empty, Dump the other list into ‘Sorted Items’.

* The remaining list is already sorted

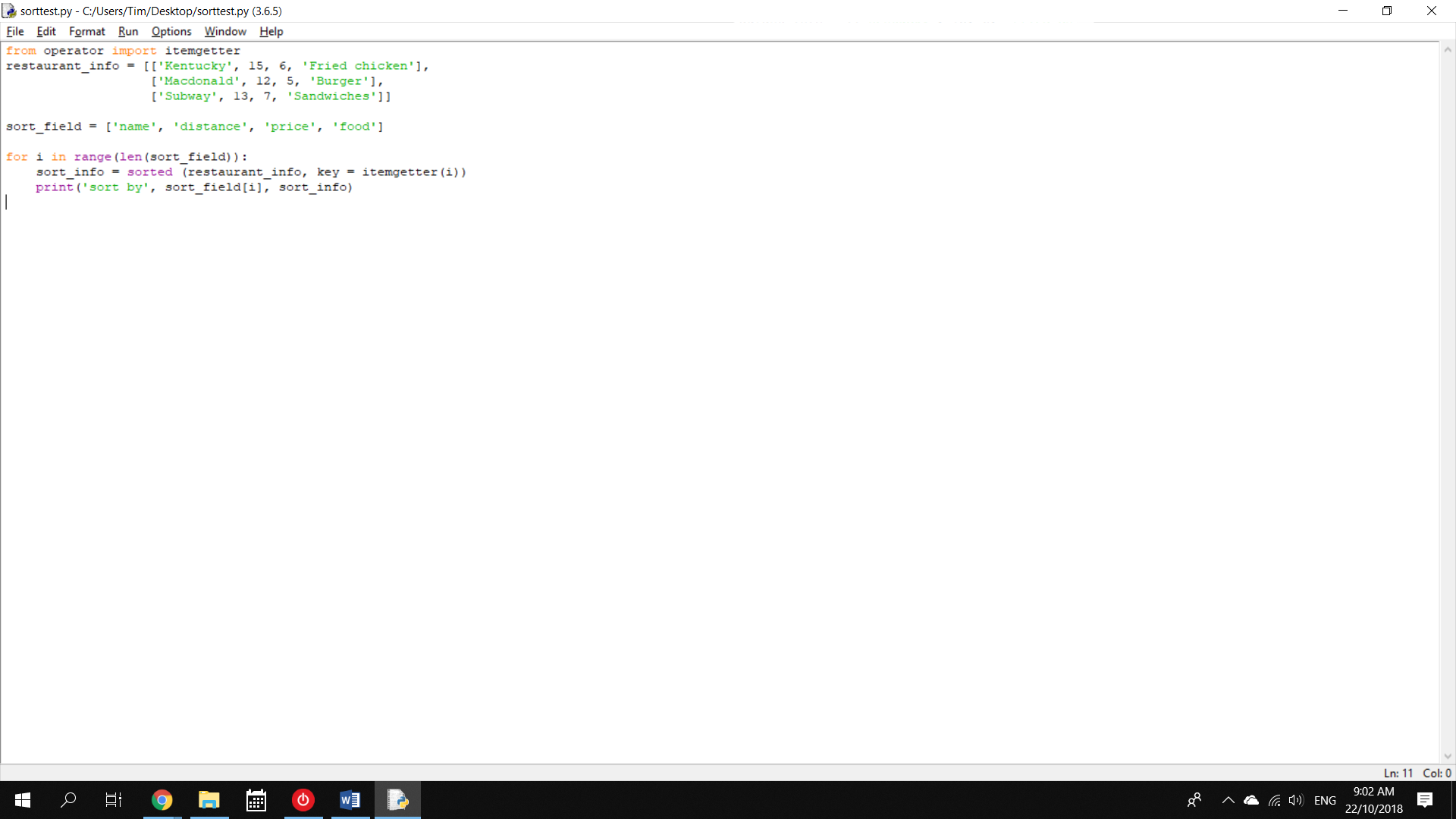
Repeat for each list until sorted.

Remember that no single sort is best.

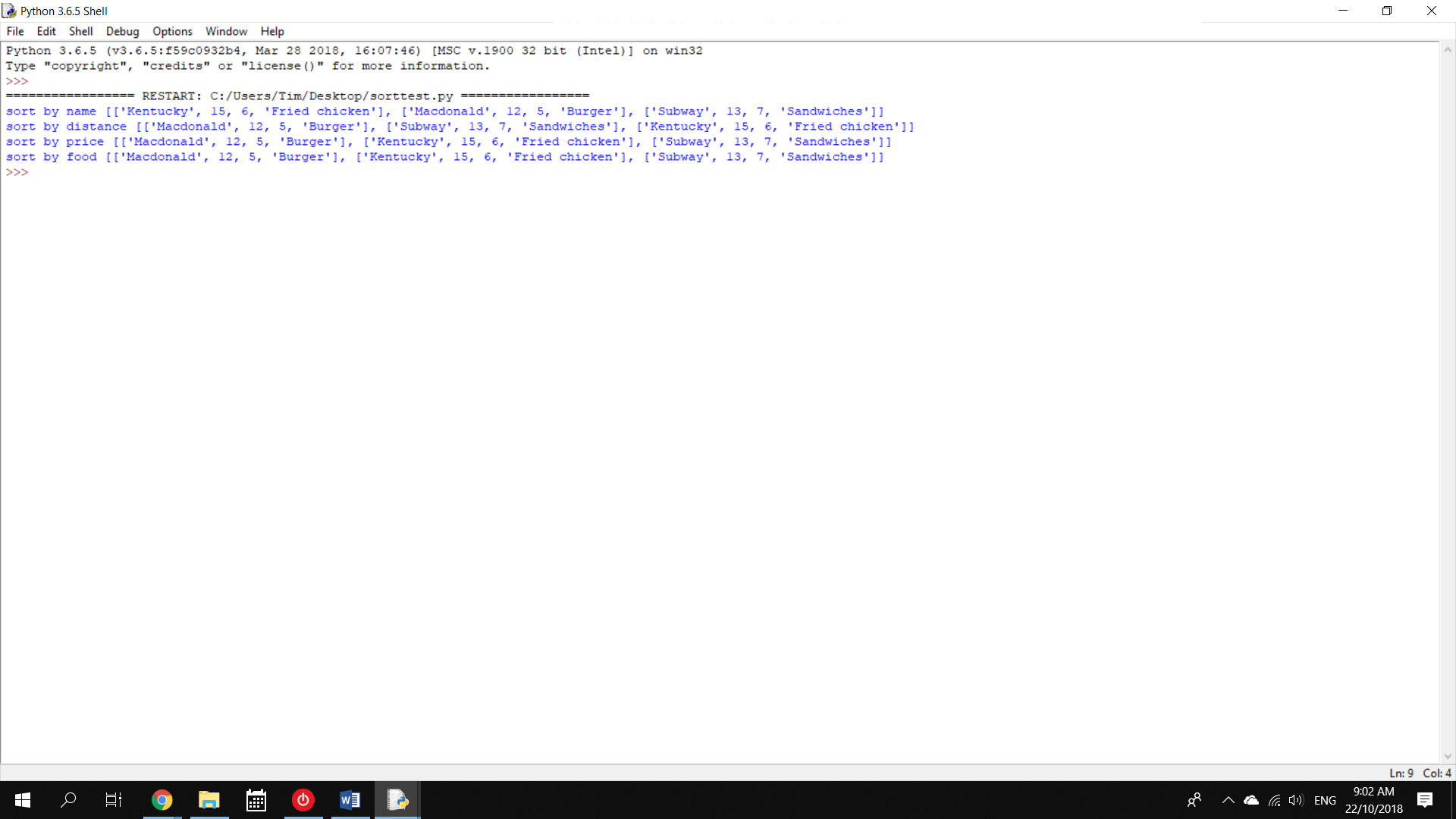
Different algorithms have different asymptotic and constant-factor tradeoffs.

**Sorting in Python**

Sort Restaurants by a bunch of Critera:



Output:



Python uses TIMSORT.

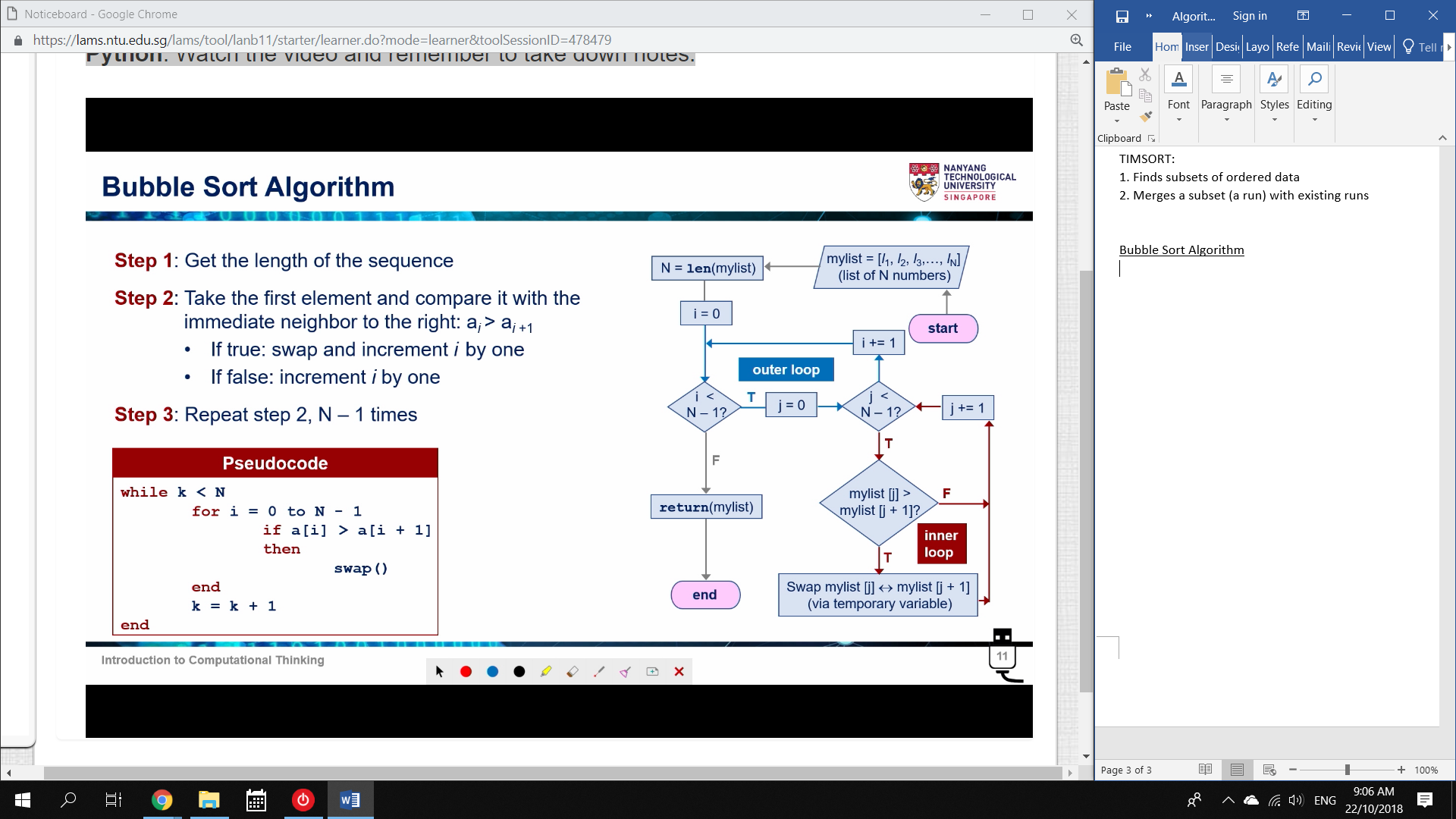
* Hybrid sort: Merge + Insertion sort.

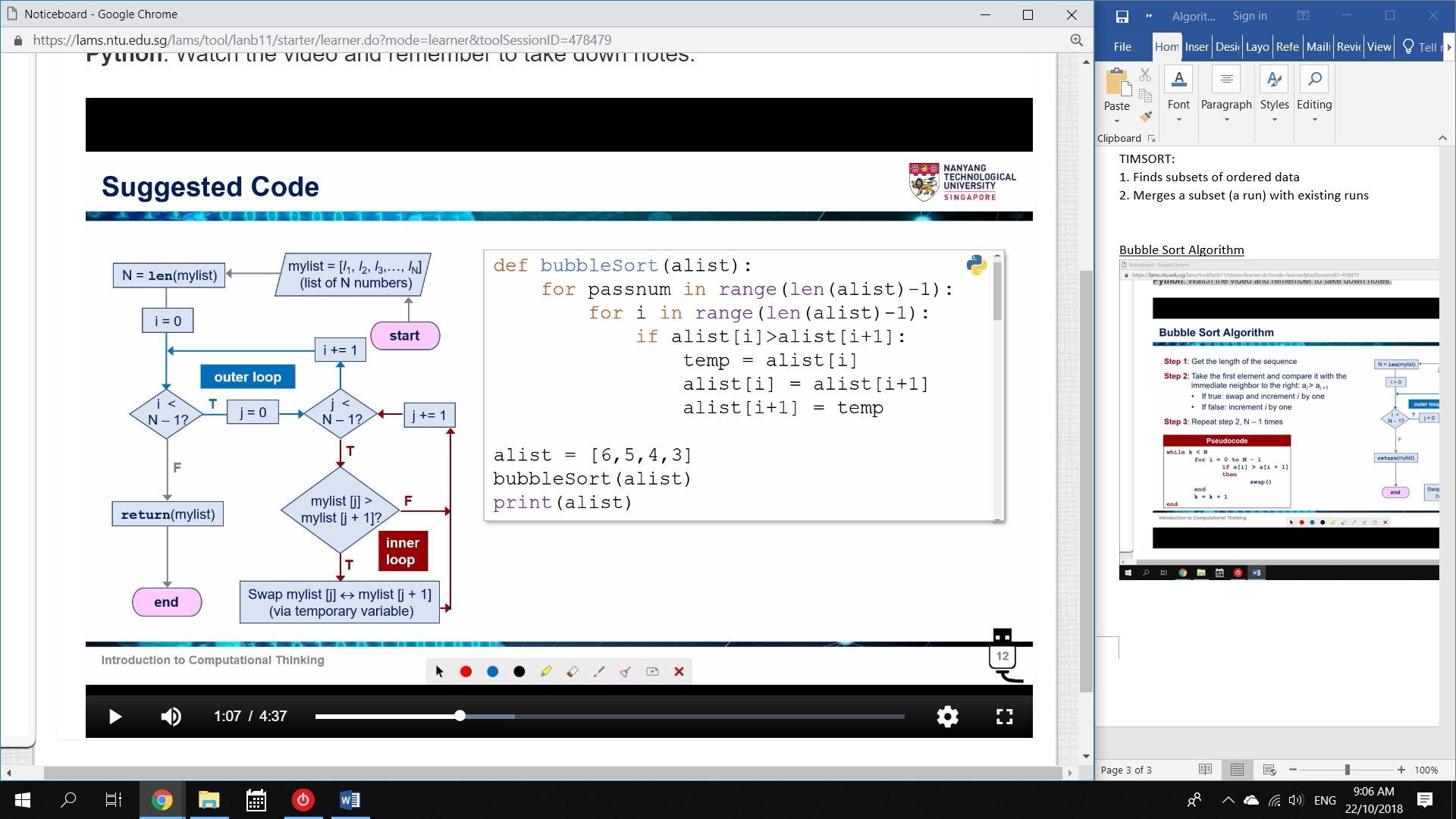
TIMSORT:

1. Finds subsets of ordered data

2. Merges a subset (a run) with existing runs

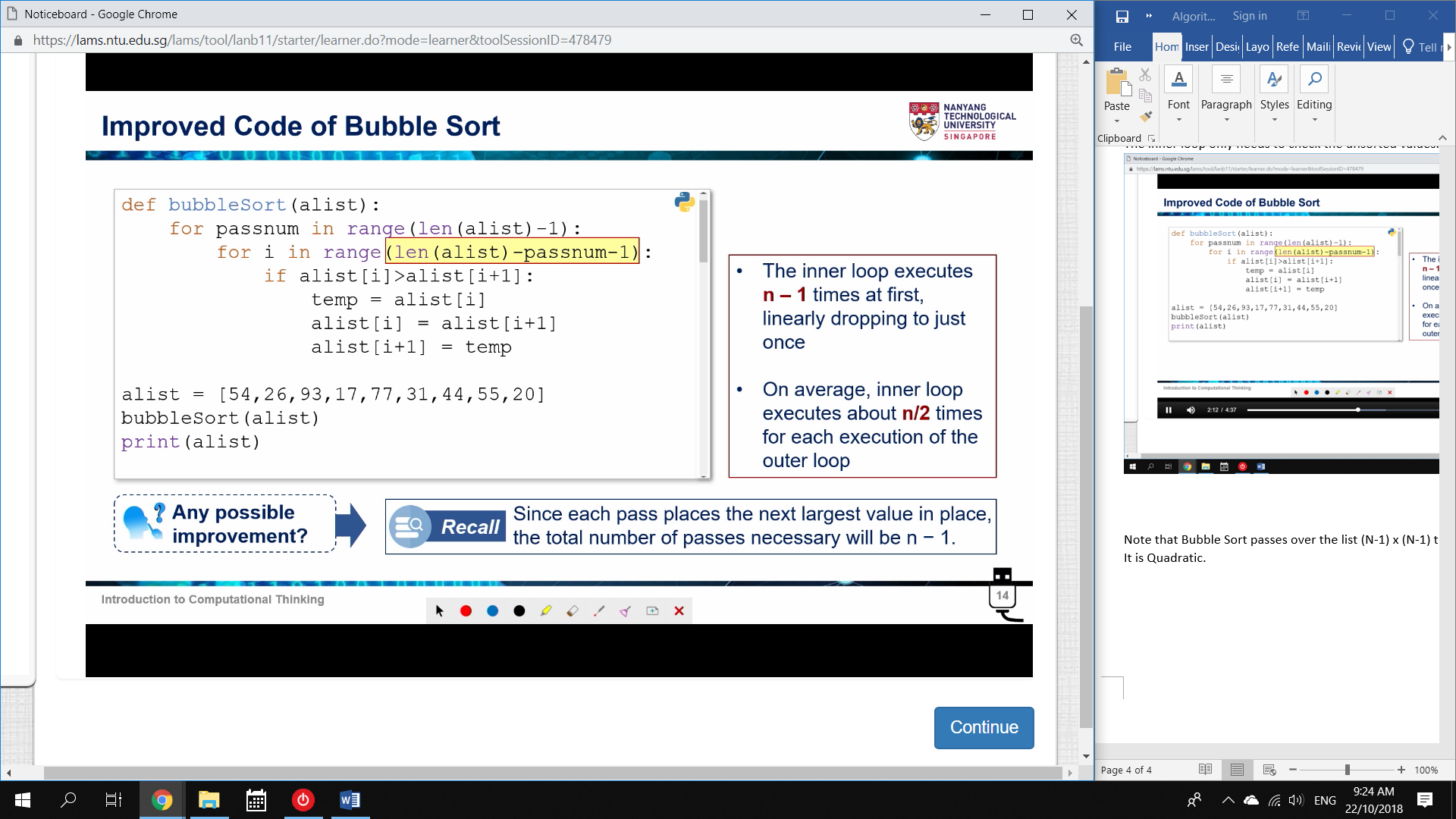
Bubble Sort Algorithm



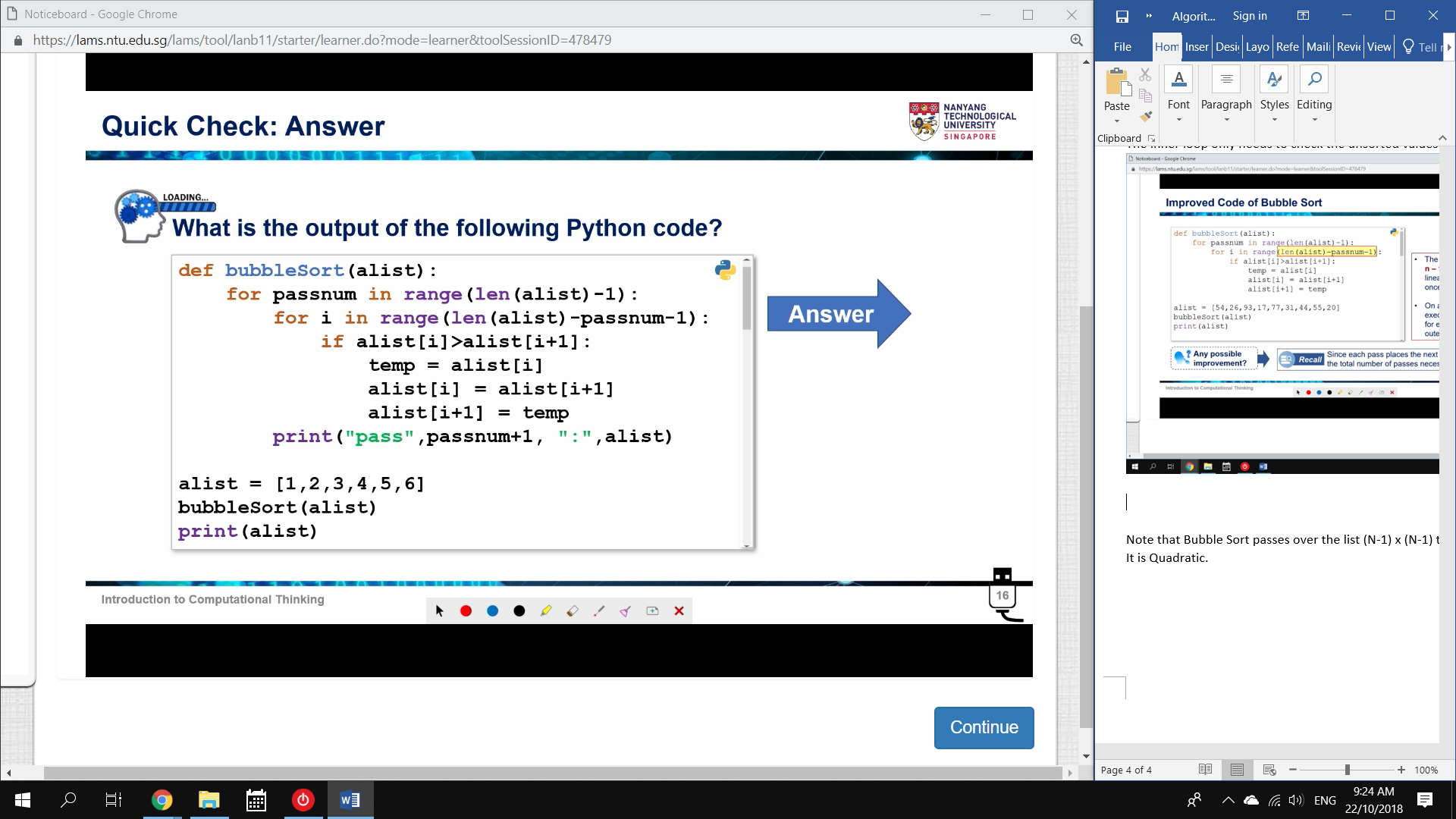


This is inefficient.

The inner loop only needs to check the unsorted values.

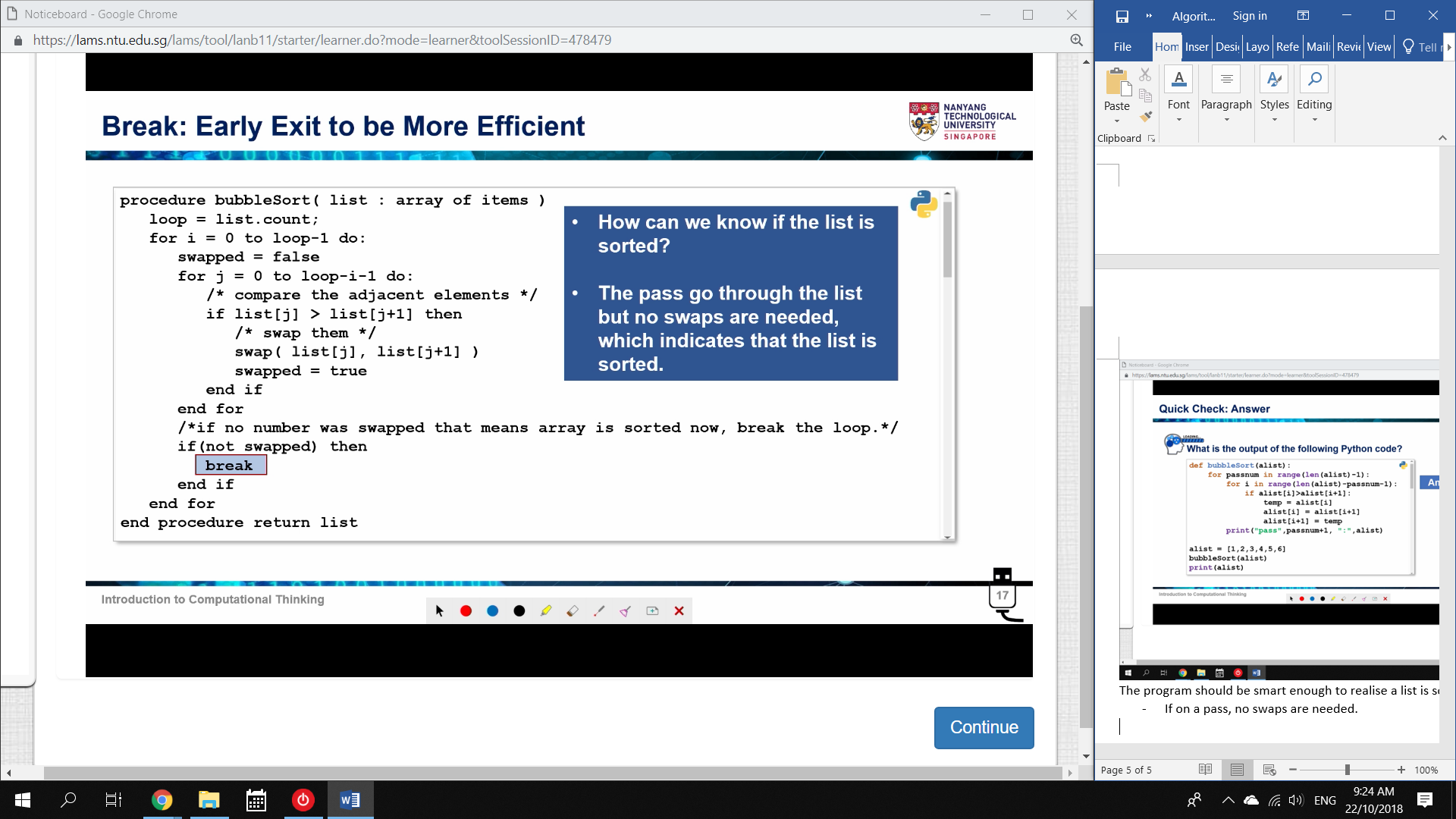


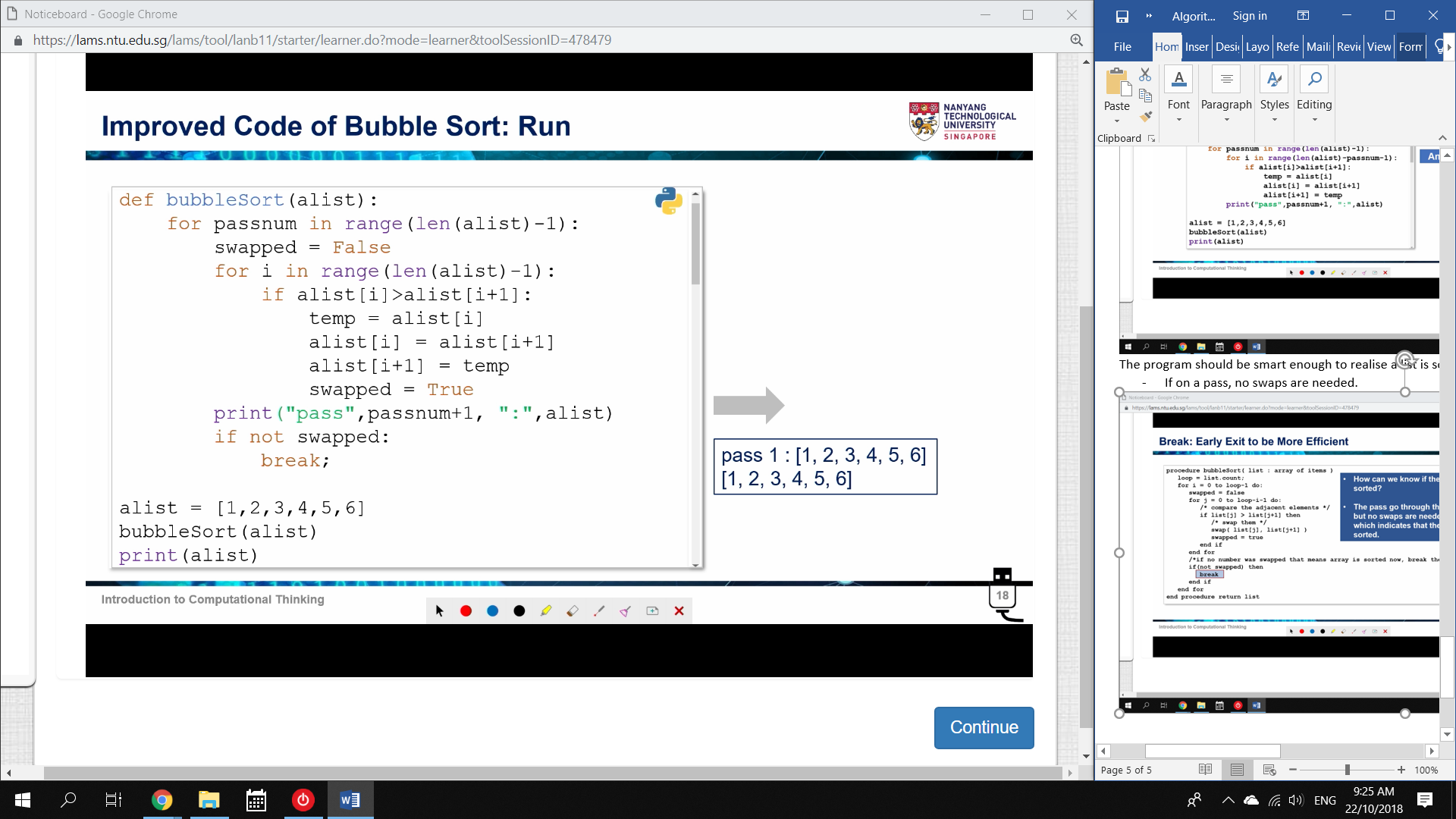
What if the list is already in order?



The program should be smart enough to realise a list is sorted.

* If on a pass, no swaps are needed.





Basically, if every pair compared is in order, no swaps are needed.

* Remember that the machine starts by holding nothing.

Therefore, if no swaps are done on a pass, we want a break.

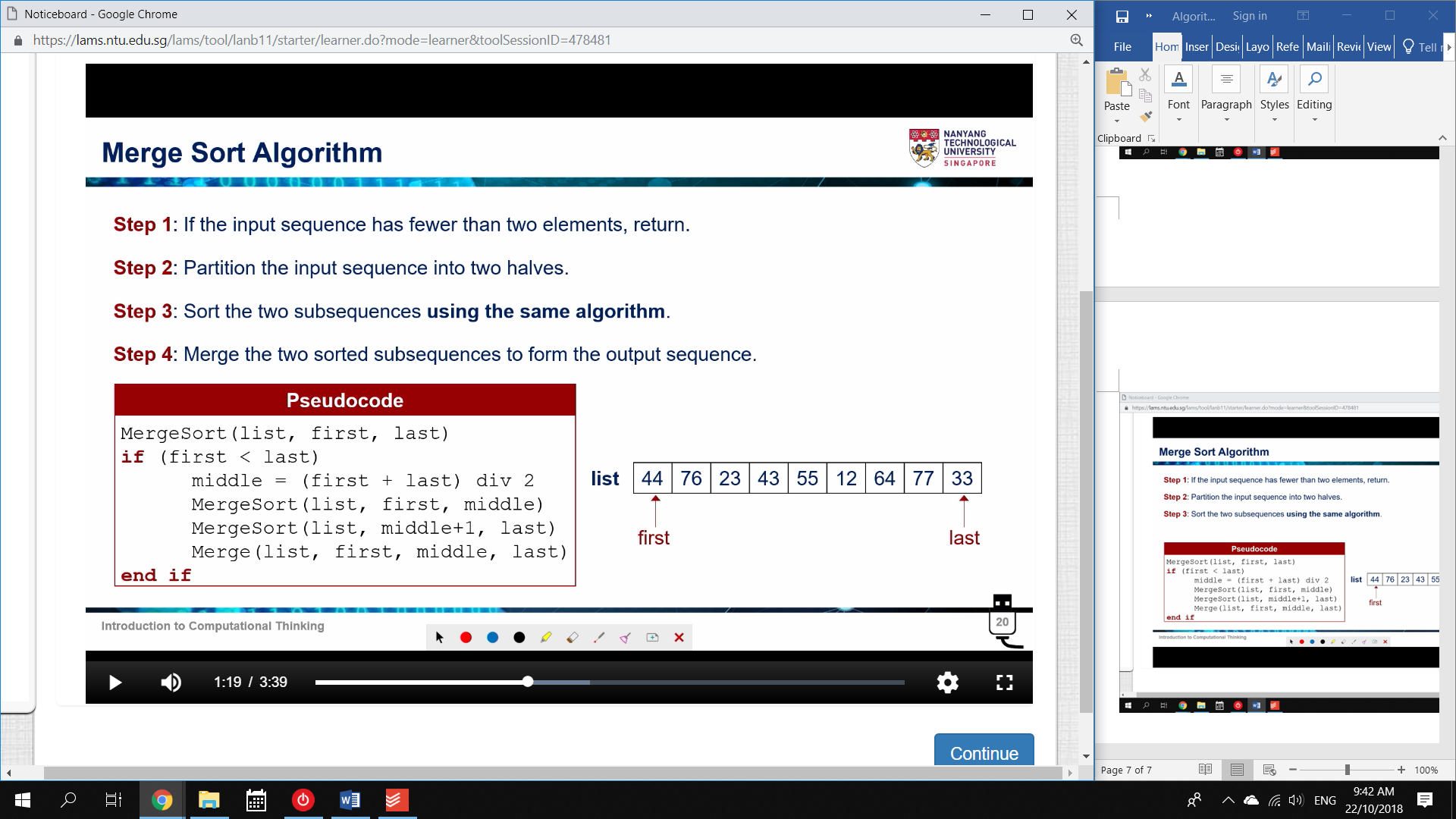
* Swapped = False unless a swap occurs.

Note that Bubble Sort passes over the list (N-1) x (N-1) times.

It is Quadratic.

Merge Sort





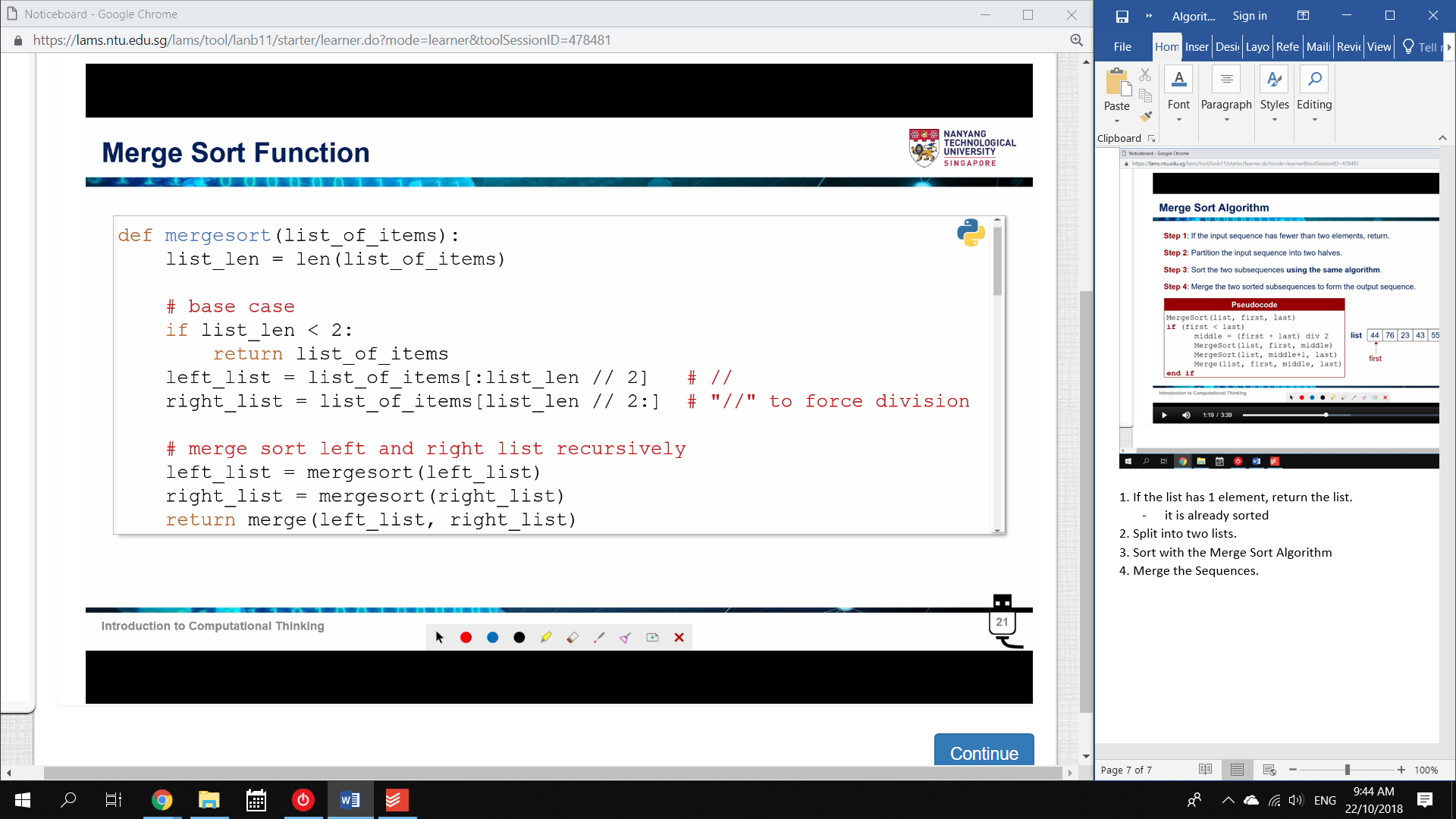
1. If the list has 1 element, return the list.

* it is already sorted

2. Split into two lists.

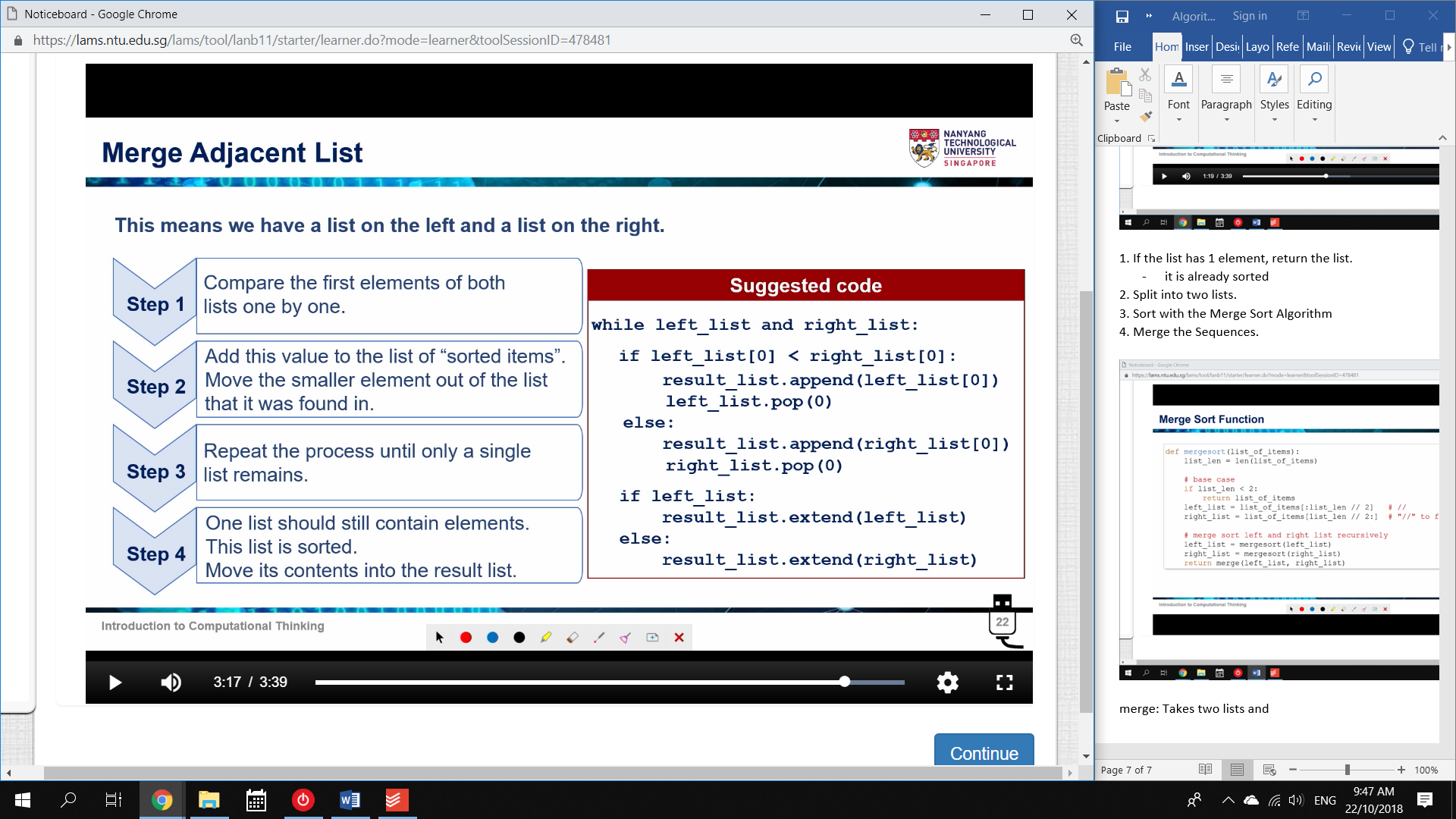
3. Sort with the Merge Sort Algorithm

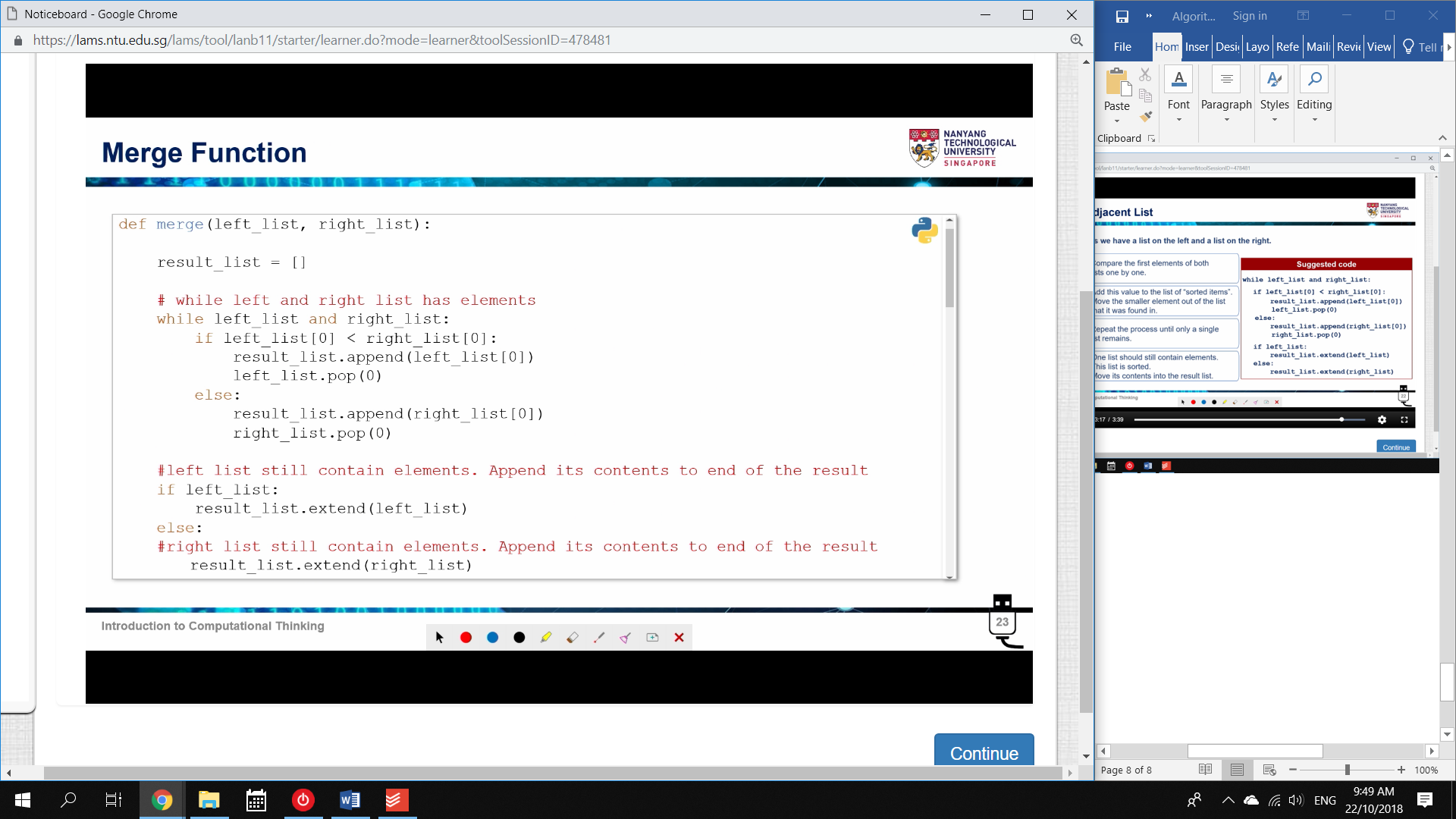
4. Merge the Sequences.



merge: Takes two lists and sorts them into one list.

* Recall the SORT part of Mergesort





The completed merge code

