Bubble Sort and Merge Sort Algorithms

**By:**

**Najla Hussein Al-Qassemi**

**Bubble Sort**

Bubble Sort is the **simplest** [sorting algorithm](https://www.geeksforgeeks.org/sorting-algorithms/) that works by **repeatedly swapping the adjacent elements** if they are in the wrong order. This algorithm is **not suitable for large data sets** as its average and worst-case time complexity is quite high.

**Time Complexity**: O (N2)  
**Auxiliary Space**: O (1)

**How it works?**

* traverse from left and compare adjacent elements and the higher one is placed at right side.
* In this way, the largest element is moved to the rightmost end at first.
* This process is then continued to find the second largest and place it and so on until the data is sorted.

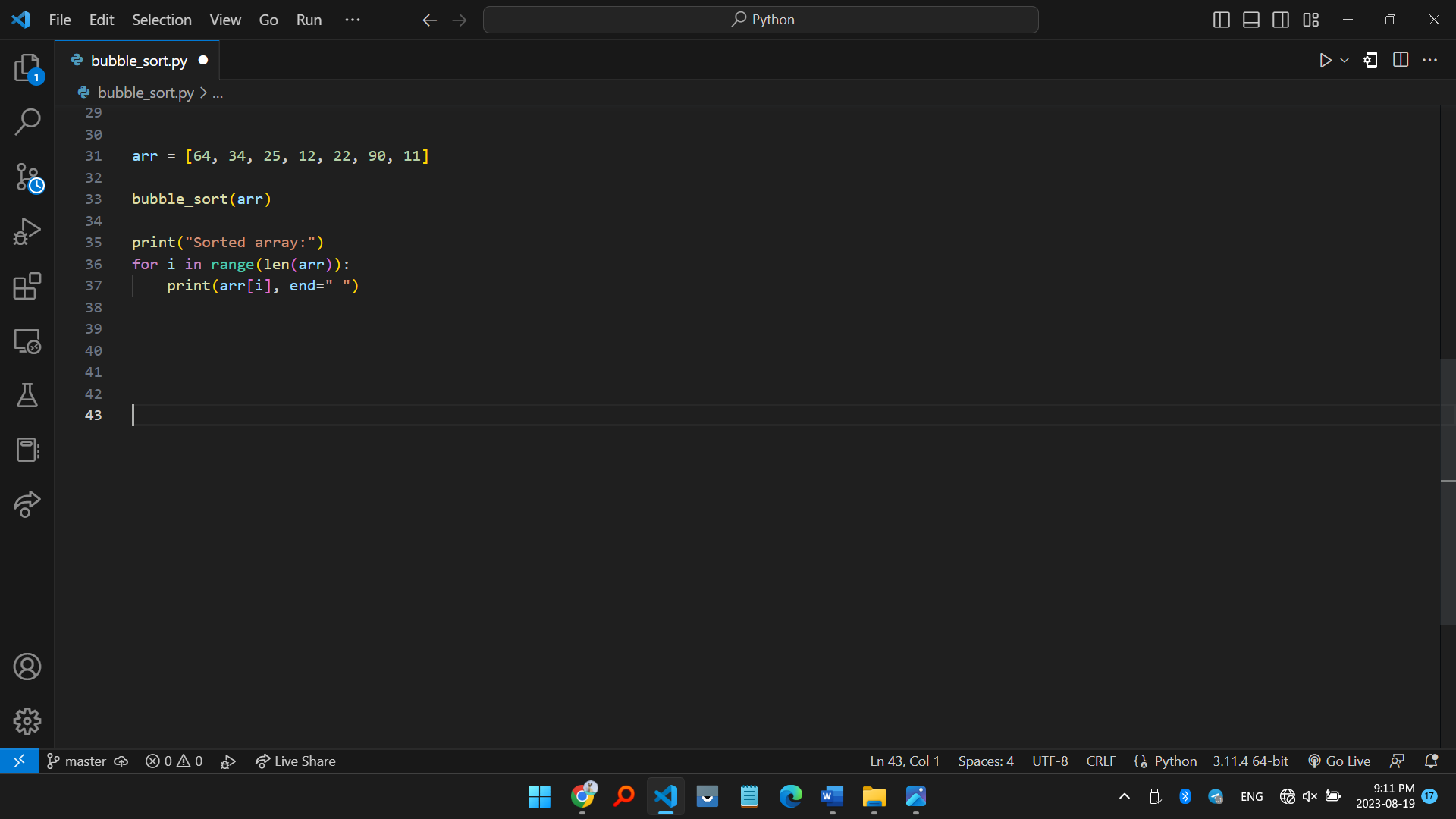
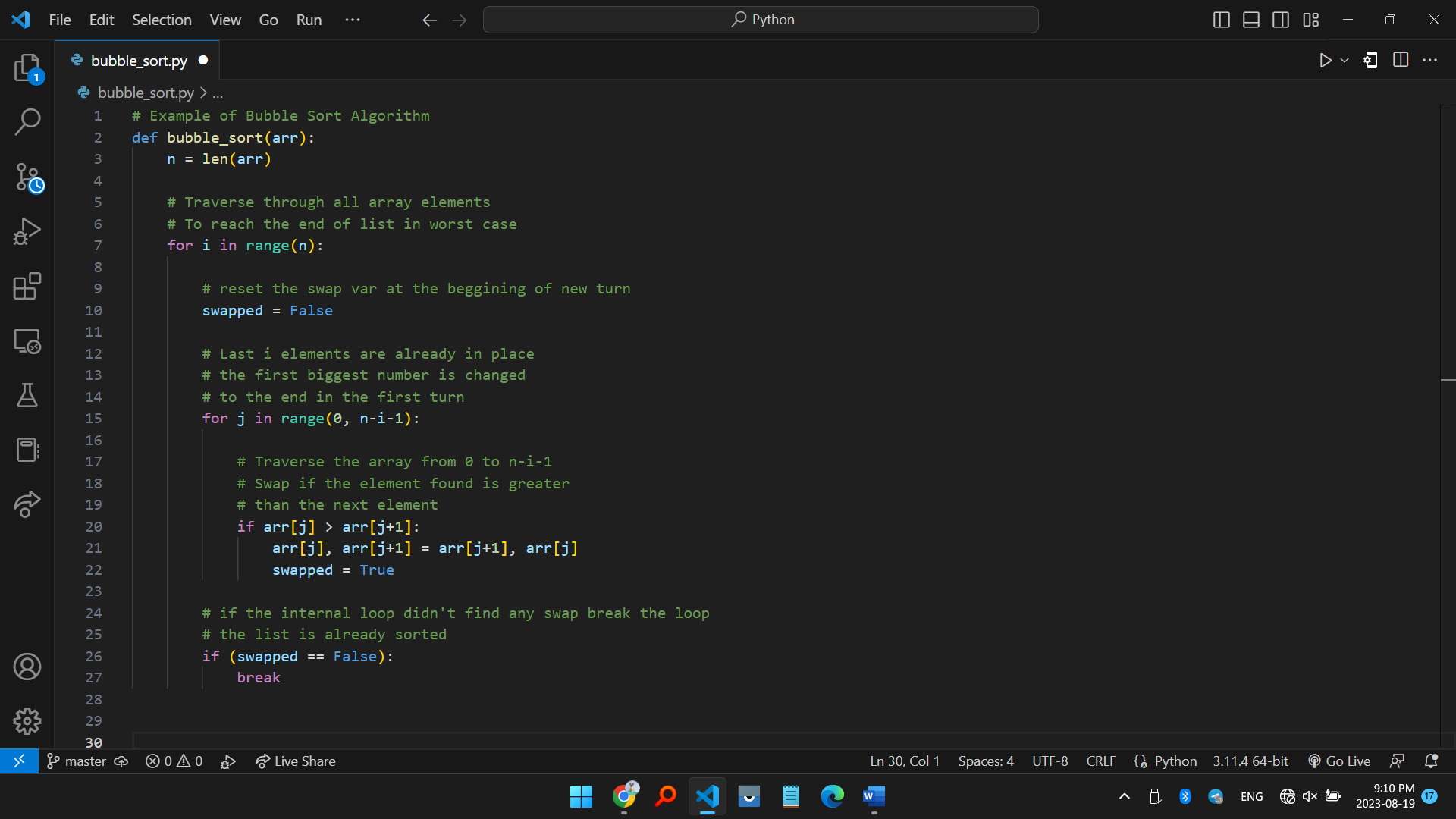
**Advantages of Bubble Sort:**

* Bubble sort is **easy** to understand and implement.
* It does **not** require any **additional memory space**.
* It is a **stable** sorting algorithm, meaning that elements with the same key value maintain their relative order in the sorted output.

**Disadvantages of Bubble Sort:**

* Bubble sort has a **time complexity** of O(N2) which makes it very **slow** for large data sets.
* Bubble sort is a **comparison-based** sorting algorithm, which means that it requires a comparison operator to determine the relative order of elements in the input data set. It can **limit the efficiency of the algorithm in certain cases.**

**Python code for Bubble sort algorithm:**



**Merge Sort**

Merge sort is defined as a [sorting algorithm](https://www.geeksforgeeks.org/sorting-algorithms/) that works by dividing an array into smaller subarrays, sorting each subarray, and then merging the sorted subarrays back together to form the final sorted array.

**Time Complexity:**O (N log(N))

**Auxiliary Space:** O (N)

**How it works?**

Merge sort is a **recursive algorithm** that continuously **splits the array in half** until it cannot be further divided i.e., the array has only one element left (an array with one element is always sorted). Then the sorted subarrays are merged into one sorted array.

**Advantages of Merge Sort:**

* Merge sort is a **stable** sorting algorithm, which means it maintains the relative order of equal elements in the input array.
* Merge sort has a worst-case time complexity of O (N logN), which means it **performs well even on large datasets**.
* Merge sort is a **naturally parallelizable** algorithm, which means it can be easily parallelized to take advantage of multiple processors or threads.

**Disadvantages of Merge Sort:**

* Merge sort requires **additional memory** to store the merged sub-arrays during the sorting process.
* Merge sort is **not** an **in-place sorting** algorithm, which means it requires additional memory to store the sorted data. This can be a disadvantage in applications where memory usage is a concern.
* For **small datasets**, Merge sort has **a higher time complexity** than some other sorting algorithms, such as insertion sort. This can result in slower performance for very small datasets.

**Python code for Merge sort algorithm:**

