**Counting Inversions in an Array**

In this assignment, we are building a program which counts the number of inversions in an array. First of all, what is an inversion? Inversion basically is a state where an element of an array is in the unsorted place or not in the correct order. This inversion count indicates how far the array is from being sorted. In the programming notation, inversion is a condition whereas we have an array, called A []. The element of A[i] is greater than element of A[j]; however, i is less than j. It means that A[i] and A[j] are not in the correct order of being sorted. It is the basic explanation of an inversion.

There are several ways to find an inversion and count them. In this assignment, we will use the more efficient method or algorithm which has a time complexity of O(nlogn). If we see an algorithm which has O(nlogn) of time complexity, we almost can predict that it uses a recursion to solve the problem. One of the most efficient and suitable method to solve this problem is using Merge Sort algorithm, which we had learned it in Chapter 2: Getting Started and Designing an Algorithm during the class period.

The basic approach of Merge Sort is using the Divide and Conquer approach. First we divide the array into smaller arrays, then we conquer the problem in each arrays recursively, and at last combine the arrays that have been solved.

**Divide and Conquer method of Inversion Count:**

1. Divide: separate the array into two halves left and right.
2. Conquer: count the inversions in each separated array recursively.
3. Combine: count inversions (a,b) where a are the elements of left array and b are the elements of right array.
4. Return the total of three inversion counts.

**The Algorithm**

SORT(Arr)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

IF (array Arr only has one element)

RETURN Arr

//Divide the array into two halves LEFT and RIGHT

SORT (Left\_Arr) ←T(n/2)

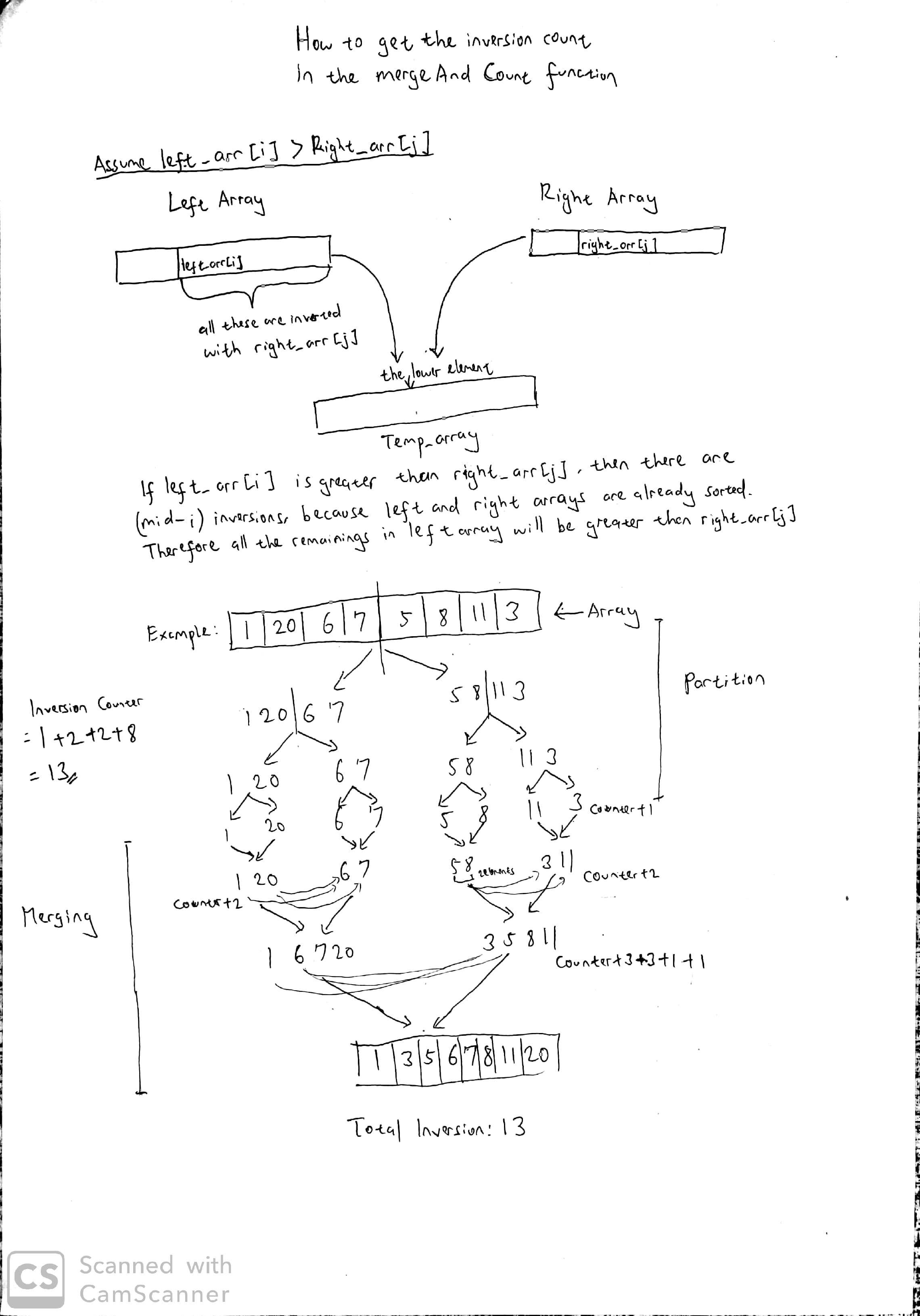
SORT (Right\_Arr) ←T(n/2)

MERGE-AND-COUNT (Left\_Arr, Right\_Arr) ←θ(n)

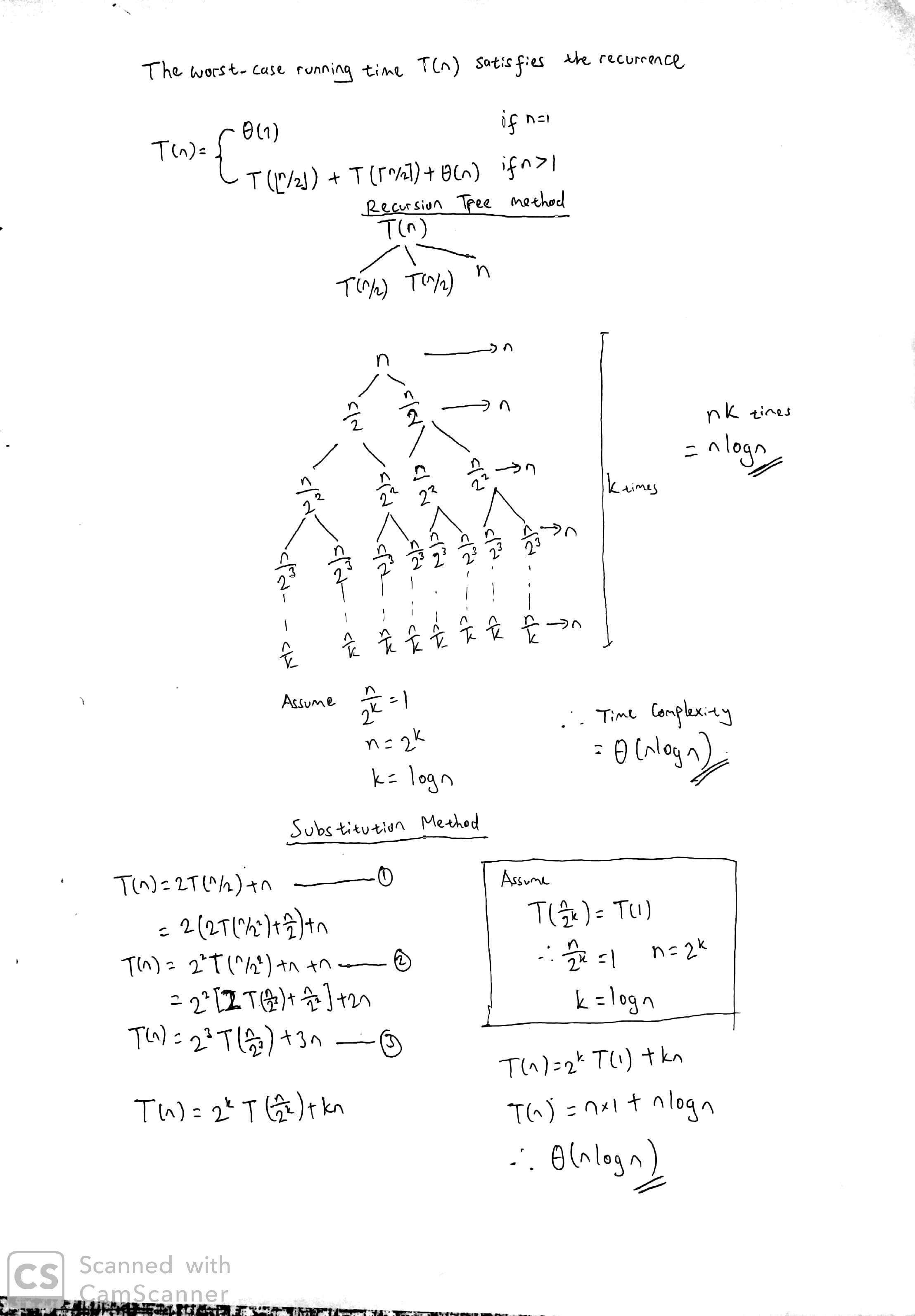
RETURN (Arr, Total\_Count)

**Merge-And-Count Analysis**

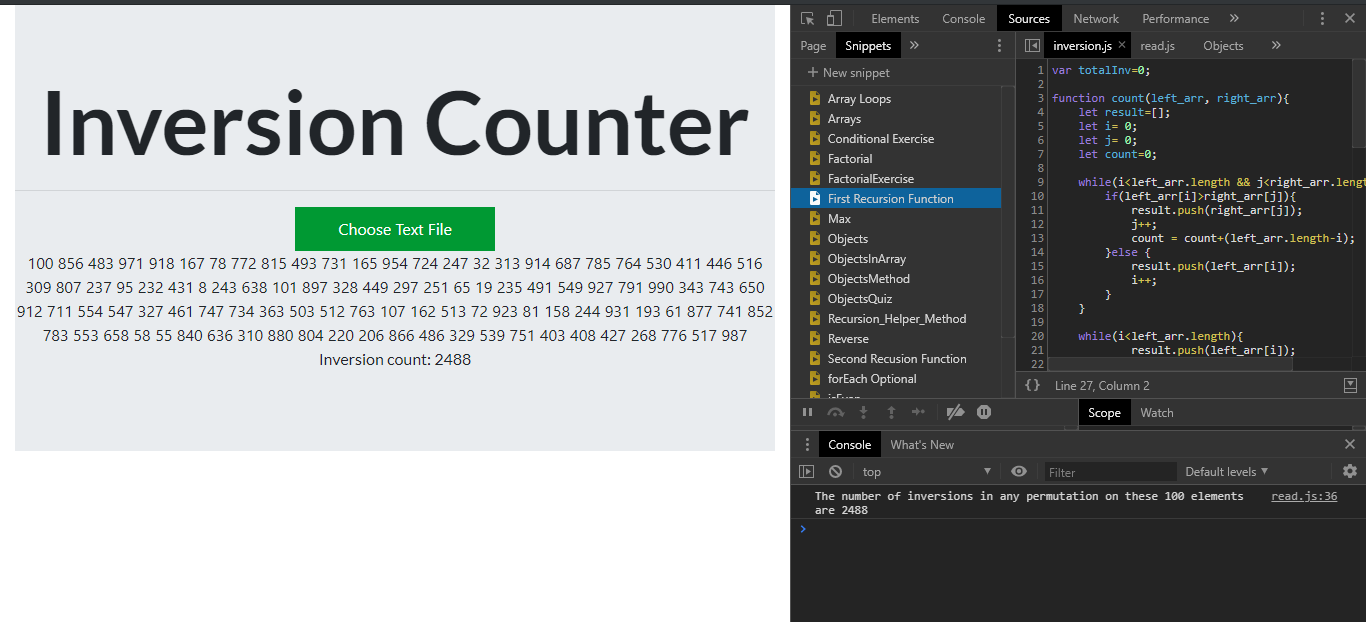
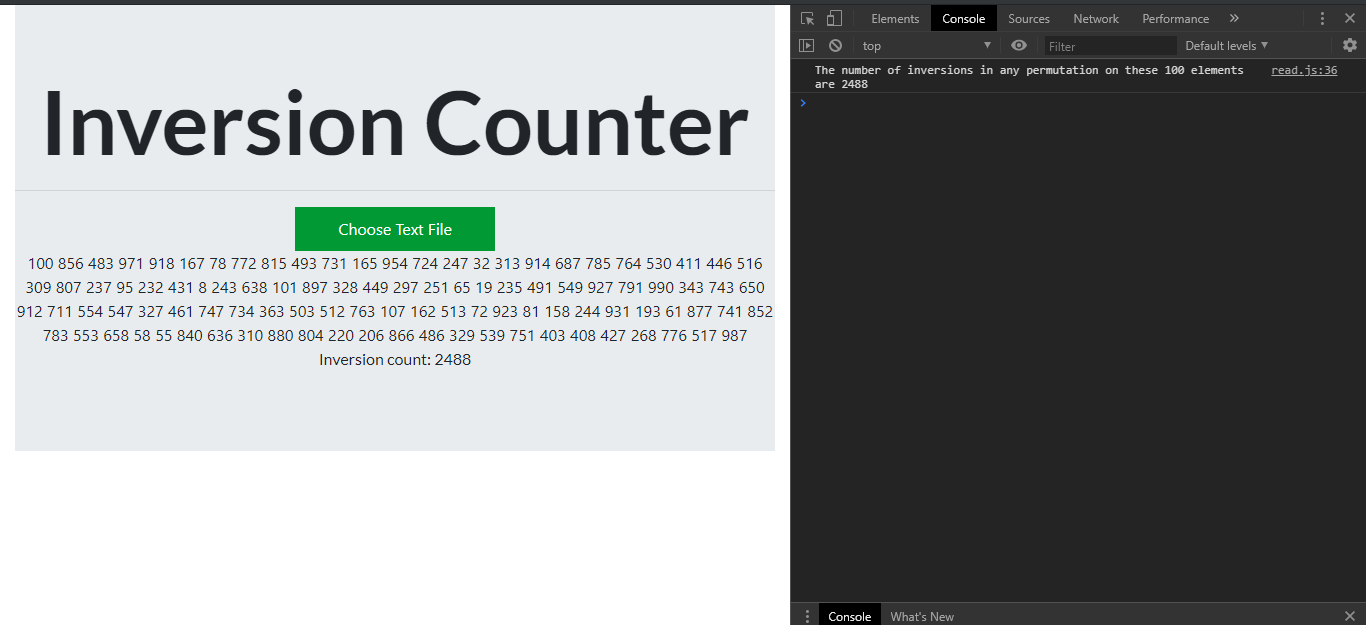
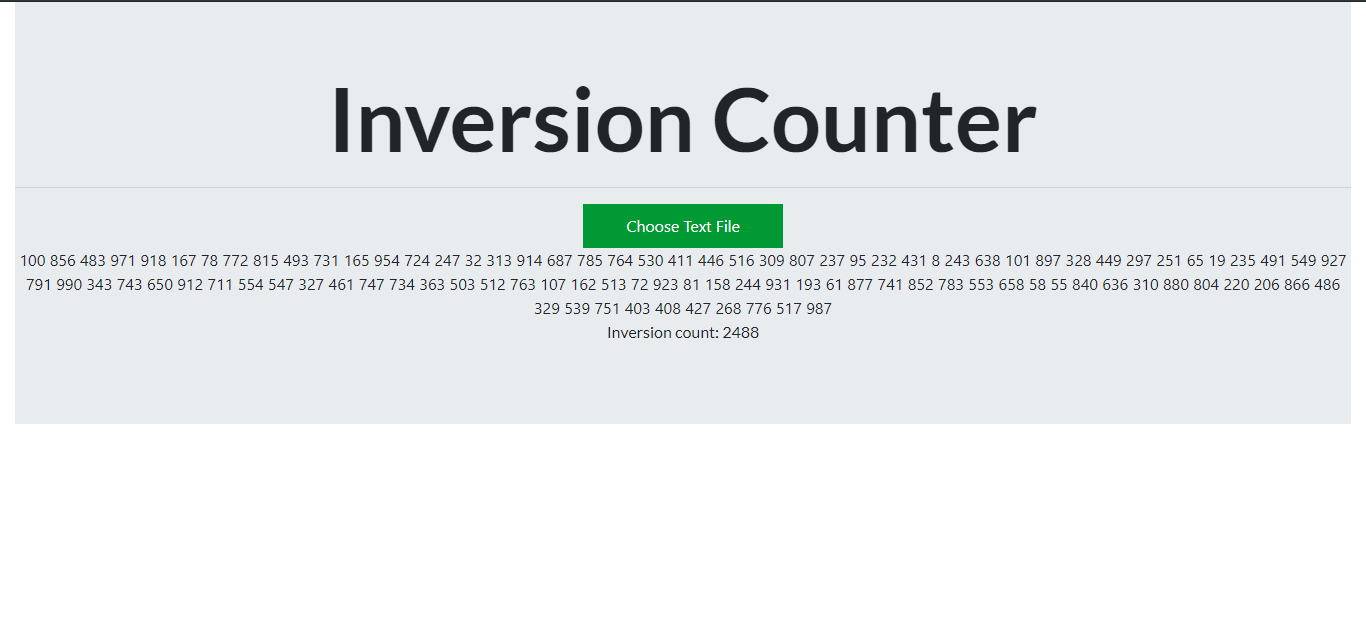
1. Scan the left array and the right array from left to right
2. Compare the element of the left array and the element of the right array which are Left\_Arr[i] and Right\_Arr[j]
3. If Left\_Arr[i]<Right\_Arr[j], then Left\_Arr[i] doesn’t have any inversion with any element left in the right array.
4. If Left\_Arr[i]>Right\_Arr[j], then Right\_Arr[j] has inversion with all the elements in the left array. Add every number of inversions to the counter.
5. Insert the smaller element into the temporary array to be sorted.



**Time Complexity**

The time complexity of this algorithm is θ(nlogn). The analysis is below.

**Execution Result**



**The Implementation**

In this assignment, I’m implementing the algorithm through JavaScript code. Why do I use JavaScript? Because lately I’ve been taking JavaScript and web development online course and I really want to learn more about the web development. The code is deployed via html file which can be viewed as a webpage. While working on this assignment, I thought of an idea on making this Inversion Counter program as a webpage which can help users to count the number inversions via online.

Throughout the process of working on this assignment, I had an opportunity to implement the things that I have learned on the web development courses which are the DOM manipulation of JavaScript, the enhancement of merge-sort algorithm, and some new JavaScript syntaxes.

**Conclusion**

From the analysis above we can conclude that there are several ways to obtain the number of inversions in an array; however, by implementing merge-sort algorithm we can achieve a more efficient way to solve the problem. I have learned that by applying sorting algorithm we can solve many programming problems and we can reverse-engineer it in order to solve many problems. After finishing this assignment, I have learned that I can implement the algorithms that we have learned during class to make a simple web application.