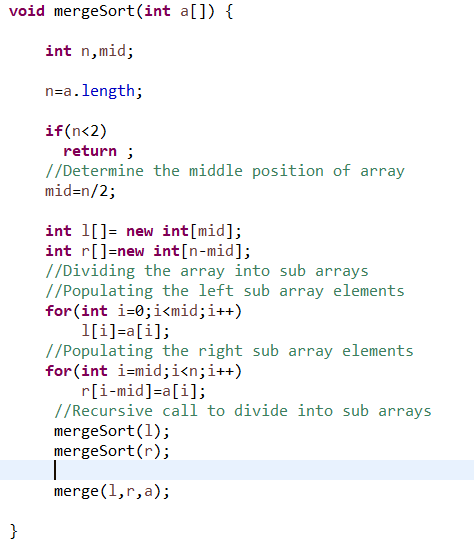
Programming Assignment 1 Report

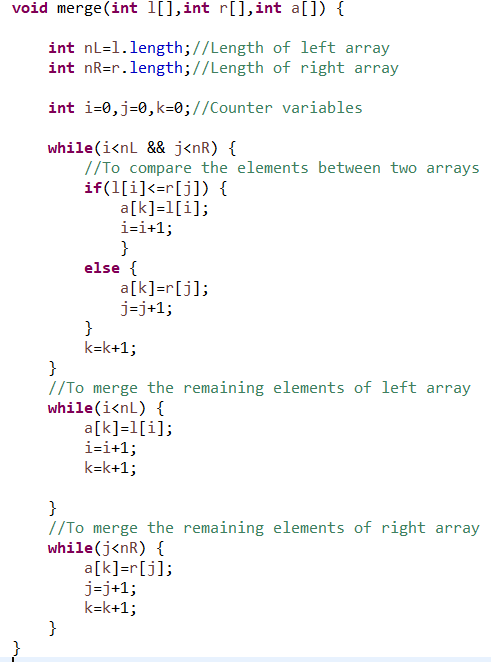
1. Merge Sort Explanation

Methods:

* mergeSort(int a[])- Array is passed as an argument. The array is divided into two sub arrays based on the middle index. This is a recursive method. After all the elements are divided until there are two elements left in sub array, merge() method is called.



* merge(int l[],int r[],int a[])- Two sub arrays and the array is passed as arguments to this method. The elements of left and right most sub array are compared and sorted. The sorted elements are stored in the merged array a[].

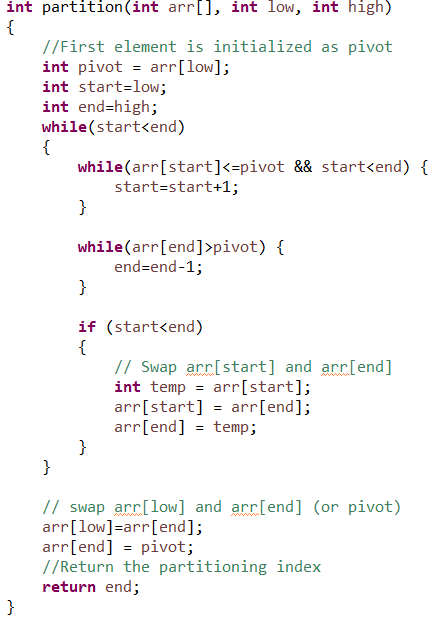


* main() – It takes the length and array elements input from the user. It passes the array to the mergeSort() method and displays the sorted array with the execution time taken.

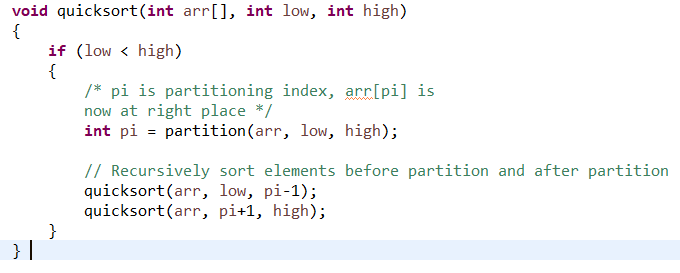
Quick Sort Explanation

Methods:

* partition(int arr[],int low, int high)- It takes array, starting and ending position of array as arguments. This method takes the first element as pivot, the pivot at its correct position ,places all smaller elements to left of the pivot and all greater elements to right of the pivot.



* quicksort()- It recursively sorts the elements before the partition index and elements after the partition index.



* main()– It takes the length and array elements input from the user. It passes the array to the partition() method and displays the sorted array with the execution time taken.

1. Time Complexity of Merge Sort

Merge Sort is a recursive algorithm and time complexity can be expressed as T(n)= 2T(n/2)+n. Using second case of the Master method, the expression is solved to **Θ(n log n).** Hence,time complexity for merge sort is Θ(n log n) for all the three cases.

Time Complexity of Quick Sort

 The worst case occurs when the partition process always picks greatest or smallest element as pivot. If we consider above partition strategy where last element is always picked as pivot, the worst case would occur when the array is already sorted in increasing or decreasing order. Worst Case is Θ(n2).

The best case occurs when the partition process always picks the middle element as pivot. It is same as Merge Sort for best case, which is Θ(n log n).

The average case is also Θ(n log n).

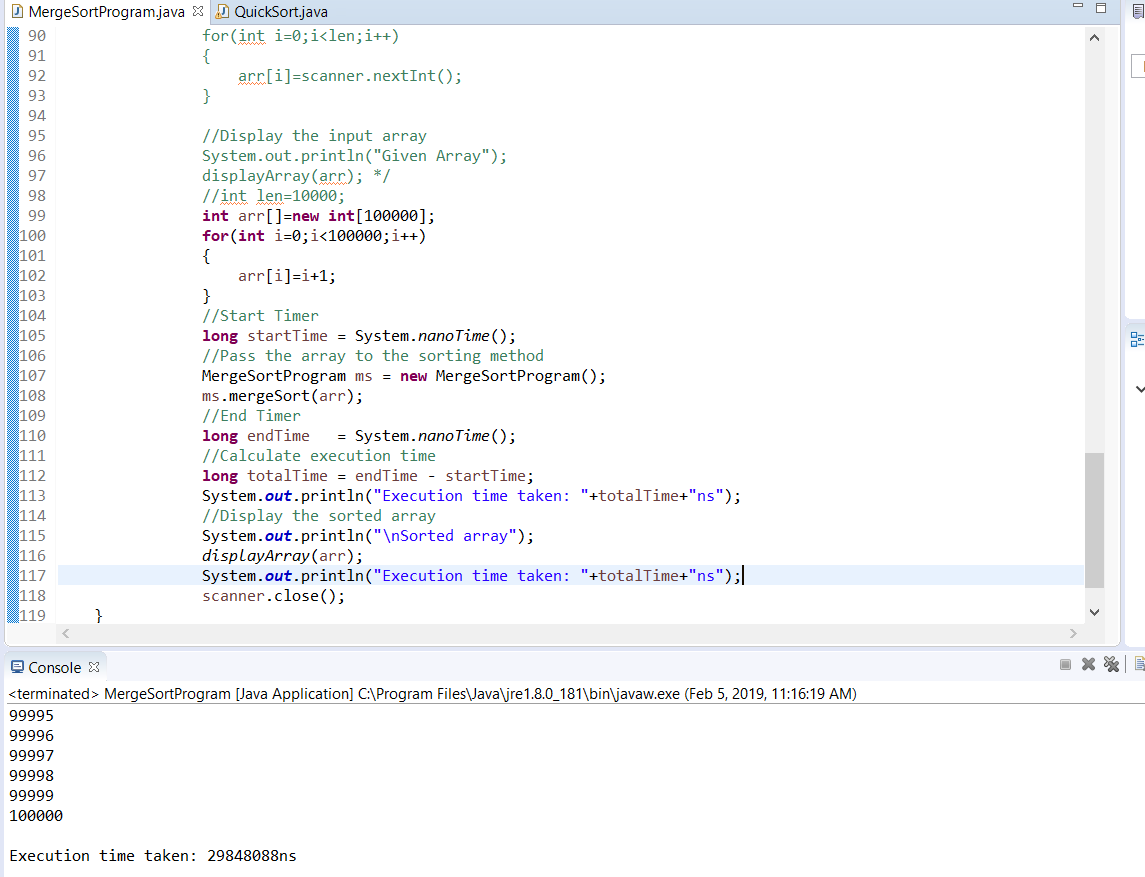
1. Applications of Merge Sort and Quick Sort

Quick Sort in general is an in-place sort algorithm. It does not require any extra storage. Hence quick sort is preferred over merge sort for sorting arrays. For arrays, merge sort loses due to the use of extra O(N) storage space.

Merge Sort is preferred over Quick Sort for sorting linked lists.  Unlike arrays, linked list nodes may not be adjacent in memory. Unlike array, in linked list, we can insert items in the middle in O(1) extra space and O(1) time. Therefore merge operation of merge sort can be implemented without extra space for linked lists.

Real time application of merge sort can be seen in E-commerce websites such as travel websites, which obtains the flight cost and hotel cost from other website and displays which hotel/ flight is the cheapest. Real world example of quick sort is while playing cards game, you select one random card as pivot and then place the remaining cards before or after it in a sorted fashion.

1. **Sorting of 100000 integer sorted array using Merge Sort**

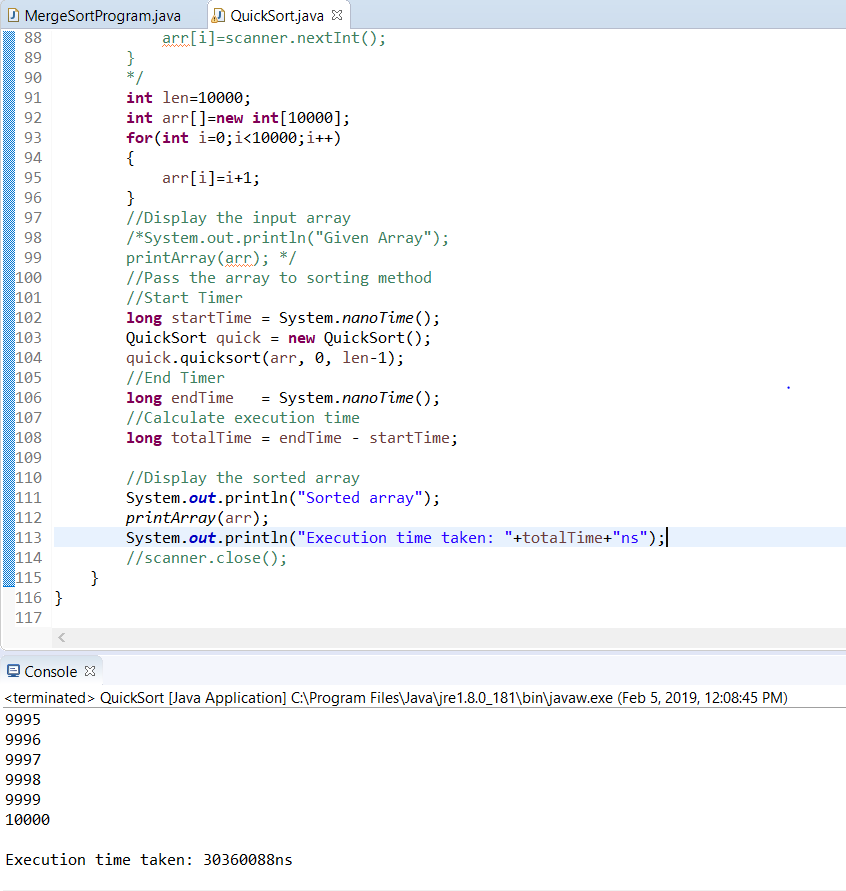


**Time taken is 29848088 nanoseconds. This conforms with the theoretical time complexity.**

**Sorting of 100000 integer sorted array using Quick Sort**

When we pass an array of 100000 integer to quick sort, it would lead to stack overflow error. **This is because we use first element as pivot, which is the worst case for quick sort as discussed in the time complexity.**

**Sorting of 10000 integer sorted array using Quick Sort**



We could see that for 10000 elements, time taken is 30360088 ns. This is more because taking first element as pivot is worst case for Quick Sort.

1. **References**

The following links were used as references.

* Class Slides:http://crystal.uta.edu/~gonzalez/alg\_spring\_2019-05.html
* <https://www.youtube.com/channel/UCZCFT11CWBi3MHNlGf019nw>
* <https://www.youtube.com/channel/UCVLbzhxVTiTLiVKeGV7WEBg>
* https://www.geeksforgeeks.org/