CSE222 / BİL505
Data Structures and Algorithms
Homework #6 – Report

EMRE KİBAR

1) Selection Sort

Time Analysis	Selection sort is a quadratic/O(n^2) sort algorithm. The aim is finding the minimum element and swap it with the current index which starts from beginning. For best case(sorted array),the sort method makes unnessesary compresions because for each iteration the smallest element is current element.For average(randomized) and worst(reversely sorted) case,sort method search method again makes comperisions which is necessary and swaps minimum element with current index.
Space Analysis	The space complexity is O(1) for selection sort because in any case it doesn't use new memory except of a few variables.

2) Bubble Sort

Time Analysis	Bubble sort is a quadratic/O(n^2) sort algorithm. The aim is taking consecutive pairs from arrays and swap the bubbled elements in order. For best case(sorted array), the iteration will stop after swap doesn't happen at first iteration which indicates array is sorted and this makes reducing the unnessesary compersions but for average case(randomized) and worst(reversely sorted) case,the iteration and comperision continues until swaps ends that means array is sorted.
Space Analysis	The space complexity is O(1) for bubble sort because in any case it doesn't use new memory except of a few variables.

3) Quick Sort

Time Analysis	Quick sort is a O(nlogn) sort algorithm in average case where all elements is randomized.The aim is that achieving two subarrays by using a random pivot point and sorting them seperately in recursive manner. Quick sort works in poor efficency for sorted and reversely sorted arrays because choosing a pivot and dividing the main array into two parts doesn't work as efficent as expected.
Space Analysis	The space complexity is O(logn) due to dividing the array two parts based on pivot point and sorting them seperately in a recursive manner.

4) Merge Sort

Time Analysis	Merge sort is an O(nlogn) sort algorithm. The aim is dividing the array into two halves recursively and after two halves are sorted then mergeing them into single array. Merge sort works in the same order with best, average and worst case, in terms of efficency it works in the same manner and there isn't any swaping for merge sort because it uses temporary arrays to hold sorted arrays.
Space Analysis	The space complexity is O(n) due to using temporary arrays for recursive steps to sorting and mergeing.

General Comparison of the Algorithms

Selection sort is the least preferable sorting algorithm in terms of time performance because in any case it has O(n^2) complexity.Then,bubble sort comes in second, it also has O(n^2) time complexity in worst and average case but it becomes O(n) for the best case which the array is already sorted.In terms of space complexity, both selection and bubble sort has same O(1) complexity because they doesn't use any extra memory space except a few variables but both quick and merge sort works in a recursive manner and both of them uses temporary memory in each recursive step where quick sort has O(logn) and merge sort has O(n) space complexity. If we compare the time complexity of both of quick and merge sort, quick sort has O(nlogn) complexity where the array elements are randomized to divide the array into two part based on pivot but the time complexity gets worse for sorted and reversely sorted arrays because dividing the array doesnt work efficently. Merge sort has O(nlogn) for every case because it divides the array into two parts regardless of the order of the elements,so we can say that merge sort both efficent and stable. At the end, the most preferable sorting algorithm between four of them is quick sort because it is both efficent in average case and has a better space complexity than merge sort.