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

EMİRHAN ALTUNEL

1) Selection Sort

Time Analysis	Selection short try to find the nth smallest element in the array and put it in the nth position. It does this by iterating through the array and finding the smallest element and swapping it with the nth element. This is done n times to sort the array. So the time complexity of selection sort is O(n^2).
Space Analysis	Selection sort is an in-place sorting algorithm. That means it does not require any copy of the array to sort it. So the space complexity of selection sort is O(1).

2) Bubble Sort

Time Analysis	Bubble sort iterates through the array and compares adjacent elements and swaps them if they are in the wrong order. This process guarentees that the nth largest element is in the nth position. This process is repeated n times to sort the array. So the time complexity of bubble sort is O(n^2). In this homework i have implemented the optimized version of bubble sort. I put lower and upper bounds to the array to keep track of the sorted and unsorted parts of the array. This way i can skip the sorted part of the array and only iterate through the unsorted part.
Space Analysis	Bubble sort is also an in-place sorting algorithm. So the space complexity of bubble sort is O(1).

3) Quick Sort

Time Analysis	Quick sort is a divide and conquer algorithm. It picks a pivot element and partitions the array around the pivot. It then recursively sorts the subarrays. The time complexity of quick sort is O(nlogn) due to the partitioning of the array. Unlike merge sort, quick sort does not quarentee a balanced partitioning of the array. So the worst and best case could be different than O(nlogn). I made small modifications to the quick sort algorithm to make least swaps possible but it costs me to make more comparisons.
Space Analysis	In this homework i have implemented the in-place version of quick sort. So the space complexity of quick sort is O(1).

4) Merge Sort

Time Analysis	Merge sort is also a divide and conquer algorithm. It divides the array into two halves and sorts them. Then it merges the two sorted halves. The time complexity of merge sort is O(nlogn) because it divides the array into two equal halves so best, worst and average case time complexity is O(nlogn).
Space Analysis	Unfortunately merge sort is not an in-place sorting algorithm. It requires extra space to store the two halves of the array. So the space complexity of merge sort is O(n). There are in-place versions of merge sort but they make much more comparisons and swaps.

General Comparison of the Algorithms

Algorithm	Time Complexity	Space Complexity	For small n	For large n	Simplicity
Selection Sort	O(n^2)	O(1)	Good	Bad	Simple
Bubble Sort	O(n^2)	O(1)	Good	Bad	Simple
Quick Sort	O(nlogn)	O(n)	Bad	Good	Complex
Merge Sort	O(nlogn)	O(n)	Bad	Good	Simple