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

**Feridun Taha Açıkyürek**

* **Selection Sort**

|  |  |
| --- | --- |
| **Time Analysis** | Time complexty is best case:Ω(N2) ,average case:Θ(N2), worst case:O(N2).  Since swap counter and comparison counter always remain the same in cases where the size of the array has not changed, the time confusion is the same in all cases. |
| **Space Analysis** | Since the array is not copied to another array and does not perform a recursive function, the space complex is constant and O(1). |

* **Bubble Sort**

|  |  |
| --- | --- |
| **Time Analysis** | If the array is sorted, it gives the best running time. As the complexity of the array increases, the time complexity increases. And as the irregularity of the array increases, the comparssion counter and swap counter also increase. If the array is sequential, it performs the comparssion counter as much as the size of the array and there is no need to swap. |
| **Space Analysis** | Since the array is not copied to another array and does not perform a recursive function, the space complex is constant and O(1). |

* **Quick Sort**

|  |  |
| --- | --- |
| **Time Analysis** | Time complexity varies depending on how the pivot is selected. If the pivot value is large, the number of swaps increases. If the series is sequential and a bad pivot is chosen, the time complexity takes its worst value. |
| **Space Analysis** | It uses its own array, but since it works recursively, the space complexity is logarithmic.O(log N) worst case. |

* **Merge Sort**

|  |  |
| --- | --- |
| **Time Analysis** | Merge sort does not perform a swap operation; instead, during the merge operation, the elements of two subarrays are compared and copied into a new array in the appropriate order. The number of comparisons made in the merge sort operation is determined during the merge phase of the array. Since the division operation is performed in all cases, regardless of whether the array is sequential or not. The time complexity does not change and is O(N log N). |
| **Space Analysis** | It uses a non-in-place sorting method and uses temporary arrays during merge operations. Since it is recursive, it does not use heap memory, so each recursive call creates a record in memory. This affects the space complexity. Space complexity is O(N). |

**General Comparison of the Algorithms**

If the array is sorted, bubble sort is more efficient in terms of time and space. QuickSort and MergeSort are time efficient for large data sets, but these two are not space efficient as they are recursive. Seleciton shorts are space efficient but time complexity is constant for all cases. Quick sort is when speed is important, seleciton sort and buble sort are in terms of memory usage. Merge sort may be preferred for stability.