TIME COMPLEXITY ANALYSİS

Q1)

If the recursive calls in the function are ignored, the time complexity of the function is constant time. Therefore, the time complexity of the function depends on how many recursive calls are there. For the best case that the first character of the key String is not in main String, the time complexity is only depends on length of the main String and it can be shown as Theta(n). For the worst case that the main String has the same characters with the key String sequentially, the time complexity depends on length of the main String, length of the key String and number of occurrence and it can be shown as Theta(n*m*o). For this reason, the time complexity of the function is O(n*m*o).

Q2)

If the recursive calls in the function are ignored, the time complexity of the function is constant time. Therefore, the time complexity of the function depends on how many recursive calls are there. For the best case that the middle element of the array is equal to given value, the time complexity is constant, and it can be shown as Theta(1). For the worst case that the array does not contain the given value, the time complexity is logarithmic, and it can be shown as Theta(logn). For this reason, the time complexity of the function is O(logn).

Q3)

If the recursive calls in the function are ignored, the time complexity of the function is constant time. Therefore, the time complexity of the function depends on how many recursive calls are there. The function does not contain early exit so there is not best and worst case for this function. The time complexity of the function depends on the number of subarrays of the array. The number of subarrays of the array depends on square of the length of the array. For this reason, the time complexity is quadratic, and it can be shown as Theta(n^2).