Question 1

‘If condition’ = O(1)

For each recursive call mid updated to half of previous, so complexity become 0(log n) .

By recurrence relations,

*T(1) = 1*

*T(n) = 1 + T(n/2), when n > 1*

Thus,

*T(n)= 1 + (1 + T(n/4)) =*

*2 + (1 + T(n/8))  ===>>> k  + T(n / 2^k) =*

*( FOR K = LOGN );*

*log n + T(n / 2^log n) = log n + T(1)*

*=log n + 1 =  O(log n)*

Question 2

T(n)=T(n-1) + n - 1 -----> O(n log n)

“Also analyze, what will change by following such an approach. What will be the complexity? Write down your analysis (in a doc file).”

The important thing in shuffling before running the quick sort;

Makes classic quick sort optimized and improved in terms of complexity.Possibility of sorted array is too low after shuffle thus complexity become lower(less than O(n^2)) and pivot become different because of shuffled array.

Complexity is even in worst condition O(n log n)

(Assuming after shuffle array are not sorted).

Serkan Koçoğlu 17070006004 - Best Regards..