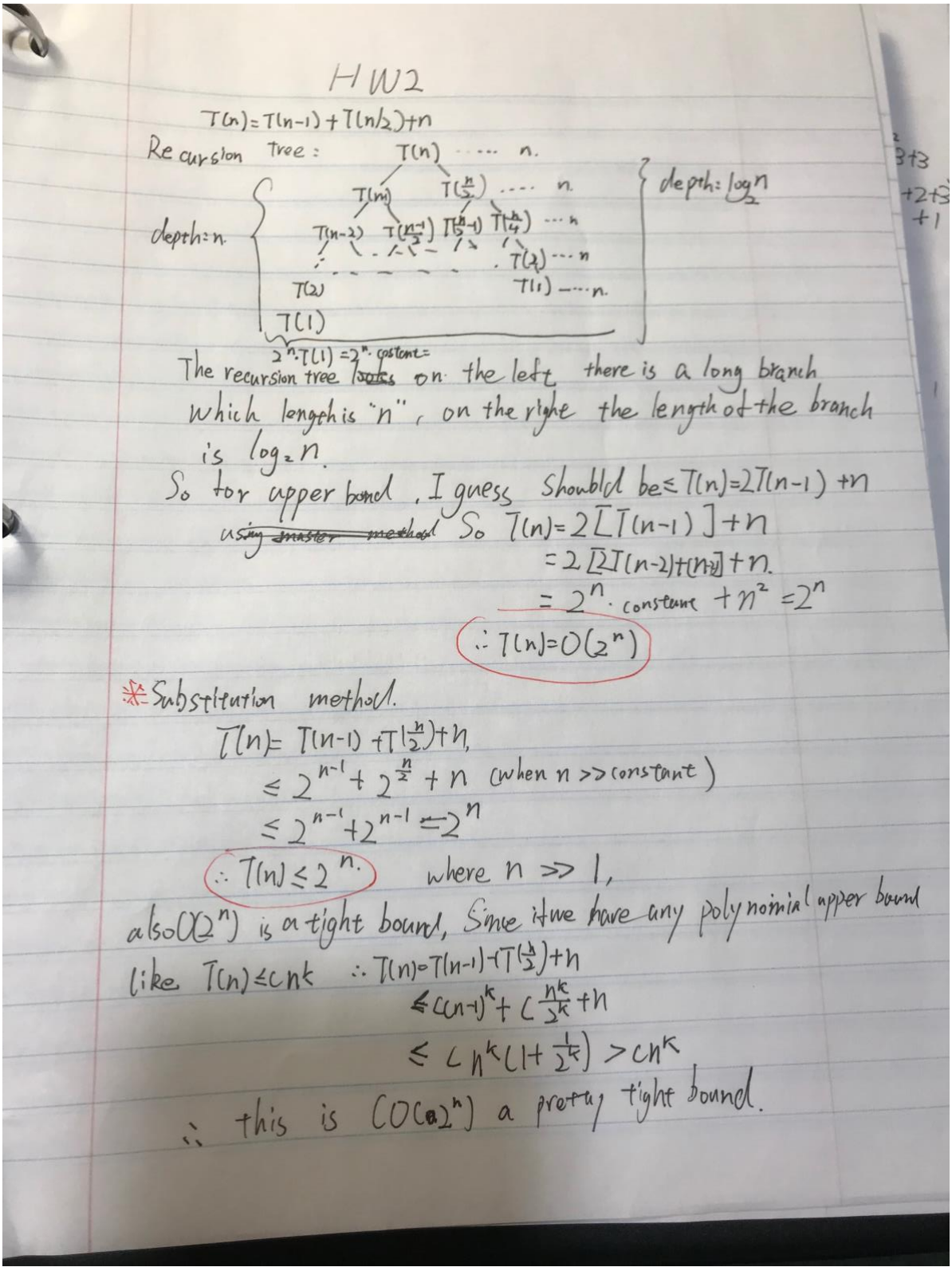
Name: Shuo Wu Student ID:452666 wustl key: wushuo

CSE541 HW2

Q1,



Name: Shuo Wu Student ID:452666 wustl key: wushuo

Q2,

A.

1. Divide the n elements of the input array into [n/9] groups of 9 elements each and at most one group made up of the remaining n mod 9 elements.
2. Find the median of each of the [n/9] groups by first insertion-sorting the elements of each group (of which there are at most 9) and then picking the median from the sorted list of group elements.
3. Use SELECT recursively to find the median x of the [n/9] medians found in step 2. (If there are an even number of medians, then by our convention, x is the lower median.)
4. Partition the input array around the median-of-medians x using the modified version of PARTITION. Let k be one more than the number of elements on the low side of the partition, so that x is the kth smallest element and there are n-k elements on the high side (or we can say, right side)of the partition.
5. If i = k, then return x. Otherwise, use SELECT recursively to find the ith smallest element on the low side if i < k. or the (i-k)th smallest element on the high side if i > k.

B.

For groups size of 9, we know at least half of medians found in step 2 are greater than or equal to the median-of-medians x. Thus, at least half of the [n/9] groups contribute at least 5 elements that are greater than x, except for the one group that has fewer than 9 elements if 9 does not divide n exactly, and the one group containing x itself. Discounting these two groups, it follows that the number of elements greater than x is at least:

$$5*([1/2*[n/9]]-2) \geq 5/18*n -10= ((5n/18)-10)$$

Similarly, at least 5/18*n-10 elements are less than x. thus, in the worst case, step 5 calls select recursively on at most 13/18*n+10 elements.

C.

$$T(n) \leq \begin{cases} O(1) & \text{if } n < \text{constant (like: 140)} \\ T(n/9) + T(13n/18+10) +O(n) & \text{if } n \geq \text{constant (like: 140)} \end{cases}$$

D.

We show that the running time is linear by substitution. More specifically, we will show that $T(n) \leq c*n$ for some suitably large constant c and all n>0. We begin by assuming that $T(n) \leq c*n$ for some suitably large constant c and all n < 140; this assumption holds if c is large enough. We also pick a constant a such that the function described by the O(n) term above (which describes the non-recursive component of the running time of the algorithm) is bounded above by $a*n$ for all n>0. Substituting this inductive hypothesis into the right-hand side of the recurrence yields:

$$\begin{aligned} T(n) &\leq c[n/9]+ c[13n/18+10]+ an \\ &\leq c*n/5 + 13c*n/18+10c +an \\ &= 83c*n/90 +10c+a*n \\ &= c*n + (-7c*n/90 +10c+a*n) \end{aligned}$$

which is at most c*n if

$(-7c*n/90 +10c+a*n) \leq 0$ - 1.1

Inequality (1.1) is equivalent to the inequality $c \geq 90a (n /(7n-900))$ when $n > 900/7$. Because we assume that $n \geq 140$, we have $n/(7n-900) <2$, and so choosing $c \geq 180a$ will satisfy inequality (1.1). (Note that there is nothing special about the constant 140; we could replace it by any integer strictly greater than 70 and then choose c accordingly.) The worst-case running time of SELECT is therefore linear.

So $T(n) = O(n)$.