CIS 575. Introduction to Algorithm Analysis
Remarks on Assignment #4, Spring 2014

Question 1 For (1), we appeal to the Master Theorem stated on the slides (the question doesn't specify whether we take $\lfloor n/3 \rfloor$ or $\lceil n/3 \rceil$ but that doesn't matter). We have $a = 9$, $b = 3$ and thus $r = \log_3(9) = 2$. It is thus case 1 of the Theorem that applies, telling us that $\mathbf{T(n) \in \Theta(n^2 \lg(n))}$.

For (2), the Master Theorem is *not* immediately applicable. But it is easy to see that $T(n) = n + (n-1) + (n-2) + \ldots$ and thus $\mathbf{T(n) \in \Theta(n^2)}$.

For (3), we may again apply the Master Theorem; we have $a = 3$, $b = 2$ and thus $r = \log_2(3)$. Since $n = n^1$ with $1 < \log_2(3)$, it is case 3 of the Theorem that applies, and we get $\mathbf{T(n) \in \Theta(n^{\lg(3)})}$ (and thus $T(n) \in O(n^2)$ and $T(n) \in \Omega(n)$.)

Question 2 Let us start with the inductive step, and calculate (using the induction hypothesis on $\lfloor n/2 \rfloor$)

$$\begin{aligned}
T(n) &= 7T(\lfloor n/2 \rfloor) + n^3 \\
&\leq 7c\lfloor n/2 \rfloor^3 + n^3 \\
&\leq 7c(n/2)^3 + n^3 \\
&= (7c/8 + 1)n^3
\end{aligned}$$

and infer that $T(n) \leq cn^3$ will hold if we demand $7c/8 + 1 \leq c$ which amounts to $\mathbf{c \geq 8}$.

And all $c \geq 8$ will indeed also work for the base case, $n = 1$, as there $T(n) = 5 = 5 \cdot n^3 \leq cn^3$.

Question 3 We again start with the inductive step, and get (using the induction hypothesis on $\lfloor n/2 \rfloor$)

$$\begin{aligned}
T(n) &= 4T(\lfloor n/2 \rfloor) + 1 \\
&\leq 4(c\lfloor n/2 \rfloor^2 - d) + 1 \\
&\leq 4c(n/2)^2 - 4d + 1 \\
&= cn^2 - 4d + 1
\end{aligned}$$

and infer that $T(n) \leq cn^2 - d$ will hold if we demand $-4d + 1 \leq -d$ which amounts to $d \geq 1/3$.

Let us pick say $\mathbf{d = 1}$, and now consider the base case $n = 1$. Then $T(n) = 3 \leq cn - d$ will hold iff $\mathbf{c \geq 4}$.

Question 4 For given $n > 1$, we observe that a call FindVal(A[1..n]) causes at most one recursive call, with an argument of size approximately $n/2$, and also the execution of a constant number of commands. Hence the time complexity is described by a recurrence

$$\mathbf{T(n) = 1 \cdot T(n/2) + f(n)}$$

where[1] $\mathbf{f(n) \in \Theta(1)}$. We can thus apply the Master Theorem, with $a = 1$ and $b = 2$ and thus $r = \log_2(1) = 0$, and with $q = 0$ as $1 = n^0$. Since $r = q$, we get

$$T(n) \in \Theta(n^r \lg(n)) = \mathbf{\Theta(\lg(n))}$$

[1]Though one could argue, cf. the slide on "What is an Elementary Instruction?", that the command $m \leftarrow \lceil n/2 \rceil$ runs in time $\Theta(\lg(n))$ since it takes $\lg(n)$ bits to represent n.