Midterm Solutions

(1) $a(a+g+t)^*(g+t)$

(2) On input 0, we go to State $\{A, B, C\}$ and on input 1, we go to State $\{B\}$.

(3) The run-time is dominated by the first set of loops, in which constant work is done. The time is $\Theta(n^3)$.

(4) The algorithm calls itself six times on inputs of length $n/2$ but otherwise does only constant, administrative work. Thus $T(n) = 6T(n/2) + C$. We calculate $\log_2(6) \approx 2.585$. Since $C \in O(n^{\log_2(6)-\epsilon})$, $T(n) = \Theta(n^{\log_2(6)})$, which is indeed pretty horrid.

(5) If $n = 3m$, we have m groups. For each group, we use three comparisons. This gives $3m$ comparisons so far. Then we calculate the maximum of the m group maxima using $m-1$ comparisons and the minimum of the m group minima using $m-1$ comparisons. This gives a total of $5m-2$ comparisons, which is indeed less than $2(n-1) = 2(3m-1) = 6m-2$ comparisons. If $n = 3m+1$, we still use three comparisons for the first m groups, but none for the last group that only has one element. This gives $3m$ comparisons. But now we have $m+1$ potential maxima and minima because of the lone element in the last group. This means, we need $2m$ comparisons to determine the maximum and the minimum. In total, we have $3m + 2m = 5m$ comparisons, which is still better than the $2(n-1) = 2(3m+1-1) = 6m$ comparisons of the naïve method.

(6) We need to do a left rotate. "cod" goes to the root, "eel" comes down to the yellow node, which also acquires node "dog doe" in addition to node "kea kid".

(7) Since $100 = 64 + 36$, the level is 6 and the split pointer is 36. Since $100 \,\%\, 64 = 36$, which is **not** smaller than the split pointer, the record goes to Bucket 36.

(8) Before the first iteration, i.e. for $j = 0$, answer is $A[0]$, which is the maximum of $A[0:1]$. For the induction step, we assume that answer is the maximum of $A[0:j+1]$ and have to show that for the next iteration value, namely $j+1$, the loop invariant is true. We distinguish two cases. First case: The maximum is the last element in $A[0:j+2]$, namely $A[j+1]$. In this case, the if triggers and answer is set to $A[j+1]$, which is indeed the maximum. Second case: The maximum in not the last element in $A[0:j+2]$, namely $A[j+1]$. Then $A[j+1]$ is smaller than some number in $A[0:j+1]$ and the if-statement does not trigger. In this case, `answer` stays the same and by induction hypothesis, is equal to the maximum of $A[0:j+1]$, which is also the maximum of $A[0:j+2]$ because of our case assumption. This terminates the proof. Applying the loop invariant to the situation after the last iteration shows the correctness.