ADS practice questions: Introduction to complexity

Question 1.

Given are three different integers a, b and c.

1. Give an algorithm that yields the middle number.
2. How many worst case comparisons will your algorithm need to perform? Can this be improved? How many average case comparisons?

Question 2.

Give an algorithm that yields the largest and smallest element of an array of n numbers; try to give an algorithm that needs about $1.5n$ comparisons in the worst case.

Question 3.

Let $p(n) = a_k n^k + a_{k-1} n^{k-1} + \ldots + a_1 n + a_0$ be a polynomial in n with degree k (so $a_k \neq 0$). Proof that $p(n) \in \Theta(n^k)$.

Question 4.

Arrange the following functions from the lowest asymptotic order to the highest asymptotic order (and also indicate any functions are of the same asymptotic order):
n, 2^n, $n \log n$, n^3, n^2, $\log n$, $n - n^3 + 7n^5$, $n^2 + \log n$, e^n

(Hint: $\lim_{n\to\infty} \frac{\log n}{n^p} = 0$ (for $p > 0$) and $\lim_{n\to\infty} \frac{n^p}{a^n} = 0$ (for $a > 1$).

Question 5.

Give the asymptotic order of the solutions to the following recursive formulae, with both recursion trees and the Master theorem. $T(1) = 1$, $n > 1$ and $c > 0$ holds for all comparisons.

1. $T(n) = 2 \cdot T(n/2) + cn$
2. $T(n) = 2 \cdot T(n/2) + cn^2$
3. $T(n) = T(n/2) + cn$

Question 6.

The Tower of Hanoi: a number of discs of different sizes are stacked on a peg (the *start* peg), in order of size (with the largest disc at the bottom). There are two empty pegs next to this first peg, the *spare* and *destination* pegs. The objective is to move the entire stack from the *start* to the *destination*, disc by disc, without placing any disc on top of a smaller disc. The *spare* peg may be used when moving the discs. A recursive solution:

```
void hanoi(numberOfDisks, start, destination, spare)
{  if (numberOfDisks > 0) {
       hanoi(numberOfDisks - 1, start, spare, destination);
       move top disk from peg start to peg destination;
       hanoi(numerOfDisks - 1, spare, destination, start); }
}
```

Give a recursive formula for the number of moves and solve the equation.