1. (25 points) Give a solution to the recurrence

\[ T(n) = \begin{cases} 
3T(n/2) + n^2 - 2n, & \text{if } n > 1 \\
1, & \text{if } n = 1 
\end{cases} \]

An asymptotic solution (using \( \Theta \)-notation) would suffice.
2. (30 points) Assume you are given two arrays $A[1..n]$ and $B[1..n]$, $n \geq 1$, and you need an algorithm to produce $m, 1 \leq m \leq 2n$, the number of distinct elements of $A$ and $B$. If $n=6$ and $A=(22, 19, 22, 22, 3, 22)$ and $B=(3, 4, 5, 6, 5, 17)$ then the output would be $m=7$ because of the distinct elements 22, 19, 3, 4, 5, 6 and 17.

a) Show an $O(n \lg n)$ upper bound on the complexity of this problem.

b) Show that a lower bound on the complexity of this problem is $\Omega(n)$. 
3. (15 points) \( a \) How many binomial trees are in a (min)-binomial heap of 82 elements?

\( b \) If you know that a (min)-binomial heap has 82 distinct elements, how many of the nodes could contain the minimum element?

\( c \) If you know that a (min)-binomial heap has 82 distinct elements, how many of the nodes could contain the maximum element?
4. (30 points) Given any array $A[1..n]$ and any integer $k$, $n \geq k \geq 1$, we seek the $k^{th}$ smallest member of $A$. Consider the following two algorithms:

**ALGORITHM A:**

    HEAPSORT $A$
    return $A[k]$

**ALGORITHM B:**

    INSERTIONSORT $A$
    return $A[k]$

Tell whether each of the following statements is true or not, and justify your response.

- **a** ALGORITHM A will solve the problem in worst-case time in $O(n + k \lg n)$.

- **b** ALGORITHM B will solve the problem in worst-case time in $O(n + k \lg n)$.

- **c** ALGORITHM B will solve the problem in best-case time in $O(n + k \lg n)$. 
1. Using the Master Theorem with $a=3$, $b=2$ and $f(n) = n^2 - 2n$ we see that 
$\log_b a = \log 3 < 2$ so choosing $0 < \varepsilon < 2 - \log 3$ we have $f(n) = n^2 - 2n \in \Omega\left(n^{\log 3+\varepsilon}\right)$ and $T(n) \in \Theta\left(n^2\right)$.

2.a We use an auxiliary data structure $C[1..2n]
\begin{align*}
&\text{MERGESORT}(A) \quad O(n \log n) \\
&\text{MERGESORT}(B) \quad O(n \log n) \\
&\text{MERGE}(A \text{ and } B \text{ into } C) \quad O(n) \\
&m \leftarrow 1 \quad O(1) \\
&\textbf{for } i \leftarrow 1 \textbf{ to } n-1 \textbf{ do} \\
&\quad \text{if } C[i] \neq C[i+1] \textbf{ then } m \leftarrow m + 1 \quad O(n) \\
&\textbf{return } m \quad O(1)
\end{align*}

b Assume there exists an algorithm to solve this problem with time complexity not in $\Omega(n)$. Then in solving this problem for inputs $A$ and $B$, some element of at least one of them was not examined. Define $A^*$ and $B^*$ to be identical to $A$ and $B$ except that one unexamined element is changed to be some element which doesn't belong to either $A$ or $B$. But the algorithm gets the same answers on $A^*$ and $B^*$ as it got on $A$ and $B$, and hence it must construct the same $m$, which is now wrong. By contradiction, an algorithm to solve this problem in sublinear time can not exist.

3. a Three binomial trees, with 64, 16 and 2 elements.
   b The minimum element could be in any one of the three roots of the binomial trees.
   c The maximum element must be in the leaf of a tree, and the three trees contain 32, 8 and 1 leaf successively. Hence the maximum element could be in any of the 41 leaves.

4. a False For $k=1$, the time constraint is $O(n)$, and HEAPSORT takes worst-case time in $O(n \log n)$
   b False For $k=n$, the time constraint is $O(n \log n)$, and INSERTIONSORT takes worst-case time in $O(n^2)$.
   c True The best-case execution time of INSERTIONSORT is $O(n)$, and for any $k$ between 1 and $n$, the time constraint is at least linear. That is, $n \in O(n + k \log n)$.