1. [9 points]
   (a) [4 points] Construct a deterministic finite automaton that functions as a 25-cents candy machine. The alphabet consists of three symbols n, d, and q (representing nickel, dime, and quarter, respectively). If more than 25 cents is deposited, no change is returned and no credit is given for overage. For examples, the input strings dnn, q, qn, nnm should be accepted, but the input strings ndn and dd should not be accepted. Give the state-transition diagram of the machine. A brief and precise interpretation of the states and transitions of your machine is required.
(b) [5 points] Let $\Sigma = \{a, b\}$, $M = (Q, \Sigma, \delta, q_0, F)$ be a deterministic finite automaton, and $E_M = \{x \mid x \in L(M), \text{ and } |x| \text{ is even} \}$. Show that $E_M$ is regular by giving explicitly the 5-tuple definition of a deterministic finite automaton accepting $E_M$. 
2. [9 points]

(a) [3 points] Is the following statement true? Whenever a nondeterministic finite automaton $M$ with $\epsilon$-transitions with 5 states is converted into an equivalent deterministic finite automaton $M'$, $M'$ has at most 32 states. Justify your answer.
(b) [6 points] Let $k$ be a positive integer. Prove that the language:

$$\{ x \in \{a, b\}^* \mid \#_a(x) \equiv 0 \pmod{k} \}$$

can not be accepted by any deterministic finite automaton with fewer than $k$ states. Note: $\#_c(u)$ denotes the number of occurrences of a symbol $c$ in a string $u$. 
3. [12 points]

(a) [5 points] Prove or disprove the following statement: For arbitrary regular expressions $r_1$ and $r_2$ over an alphabet $\Sigma$ such that $\epsilon \in L(r_1)$, there exists a regular expression $r$ over $\Sigma$ such that $r = r_1 r + r_2$. 
(b) [7 points] Let $\Sigma = \{a, b\}$. Define a function $f$ recursively from the set of all regular expressions over $\Sigma$ to the set of all natural numbers as follows:

\[
\begin{align*}
  f(\emptyset) &= 0 \\
  f(\Lambda) &= 0 \\
  f(a) &= 1 \\
  f(b) &= 0 \\
  f(r_1 + r_2) &= \min(f(r_1), f(r_2)) \text{ for regular expressions } r_1 \text{ and } r_2 \\
  f(r_1r_2) &= f(r_1) + f(r_2) \text{ for regular expressions } r_1 \text{ and } r_2 \\
  f(r^*) &= 0 \text{ for regular expression } r.
\end{align*}
\]

i. [1 point] What does the function $f$ compute?

ii. [6 points] Use some form of induction to prove your answer.