Foundations of Computing

Instructions: This exam consists of 7 problems, each of equal weight. Please write answers on the exam. If you would like to attach additional pages, include examinee number on each page. To keep answers concise, it is permissible to refer to algorithms in books and cite page numbers.
1. Give a context-free grammar $G$ for the language

$$L = \{ w \in \{a, b\}^* \mid N_a(w) \geq 1 \}.$$

Outline a proof of correctness for your grammar.
2. a. Describe a non-deterministic finite automaton $M$ with 3 states and 4 transitions so that $L(M) = L$, where

$$L = \{ w \in \{a, b\}^* \mid w = xaa \text{ for some } x \in \{a, b\}^* \}.$$

b. How many states does the minimum-state deterministic finite automaton for $L$ have?

c. Use the Myhill-Nerode theorem to prove that your deterministic finite automaton for part (b) is minimal.
3. Let $f(x)$ be a continuous, differentiable, real-valued function, defined for $x \geq 0$. Assume that $f$ has a unique maximum at some $x = m > 1$ and further that $f$ is strictly increasing ($f'(x) > 0$) for $1 \leq x < m$ and $f$ is strictly decreasing ($f'(x) < 0$) for $x > m$. Write a procedure to find $\lceil m \rceil$ where the only operation you may perform on $f$ is to evaluate the derivative, $f'(x)$, at integer values. The goal is to minimize the number of evaluations of $f'$. Analyze the worst case number of evaluations of $f'$ as a function of $\lceil m \rceil$. 
4. The language NOT-ALL-EQUAL-3SAT consists of all 3-CNF boolean formulas $\phi$ which have a satisfying assignment giving each clause of $\phi$ at least one true and at least one false literal.

Show that 3-CNF-SAT is polynomial-time reducible to NOT-ALL-EQUAL-3SAT. (Hint: Given $\phi \in \text{3CNF-SAT}$, introduce a new (global) variable $b$ and for each clause $C_i$ of $\phi$, a new variable $d_i$. Transform each clause $C_i = (x_i \lor y_i \lor z_i)$ of $\phi$ into the conjunction $(x_i \lor y_i \lor d_i) \land (z_i \lor \neg d_i \lor b)$.)
5. Show that the language class $\mathcal{P}$ (languages that can be recognized in polynomial time) is closed under the Kleene star operation. Hint: Given that $L \in \mathcal{P}$ and given an input $y = y_1 y_2 \cdots y_n$ (each $y_i$ is a symbol of $y$), fill a Boolean table $A[1..n, 1..n]$ where $A[i, j]$ indicates whether $y_k \cdots y_j$ is in $L$. Finally, use dynamic programming to fill a table $B[1..n]$ where $B[i]$ indicates whether $y_1 \cdots y_i$ is in $L^*$. 
6. Given a set of keys $K$, a hash table $T$ of $n$ buckets $B[0], \ldots, B[n-1]$, and two hash functions $h_1$ and $h_2$ which map $K$ to $\{0, 1, \ldots, n-1\}$, double hashing is a method of assigning keys to buckets as follows. For each key $k$, define a probe sequence, $h^{(0)}(k), h^{(1)}(k), h^{(2)}(k), \ldots$, where

$$h^i(k) = (h_1(k) + i \cdot h_2(k)) \mod n.$$ 

To insert a key $k$ into hash table $H$, use the probe sequence of $k$ to search $B[h^{(0)}(k)], B[h^{(1)}(k)], B[h^{(2)}(k)], \ldots$ for an empty bucket for $k$. Key $k$ is stored in the first empty bucket encountered.

*Ordered double hashing* is a variation of double hashing in which, as the probe sequence for a key $k$ is followed, if a key $k'$ is encountered in one of the buckets such that $k' > k$, replace $k'$ in the bucket by $k$ and proceed to insert $k'$ (by ordered double hashing) as dictated by the probe sequence of $k'$.

a. Consider a hash table $T$ with 10 buckets. Consider two-digit keys in the range $[0 \ldots 99]$. Use $h_1(k) = \text{(left digit in decimal representation of } k)$ and $h_2(k) = \text{(right digit in decimal representation of } k)$. Thus $h_1(9) = h_1(2) = 0$ and $h_2(2) = h_2(12) = 2$. Insert keys in the sequence 99, 12, 16, 45, 21, 37, 43, 2, 23. Show the hash table resulting from this sequence starting from an initially empty table. Use the following techniques.

- (i) Double hashing
- (ii) Ordered double hashing

b. A hash function $h$ is monotonic if whenever $k_1 < k_2$ then $h(k_1) \leq h(k_2)$. Under the assumptions that $h_1$ is monotonic (but no restriction on $h_2$) and $T$ is constructed by ordered double hashing, describe a procedure for determining the least key in $T$ (efficiency matters). Briefly argue the correctness of your algorithm.

c. Under the assumptions that $h_2 = 1$ (but no restriction on $h_1$) and $T$ is constructed by double hashing, describe a procedure for determining the least key in $T$ (efficiency matters). Briefly argue the correctness of your algorithm.
7. a. Let $G$ be a connected, undirected graph and let $C$ be a cycle in $G$ of length 3.

(i) Suppose that $T$ is a depth-first search spanning tree of $G$. Give an example of how each of the following can occur or prove that it can never occur.

- All edges of $C$ are in $T$.

- No edge of $C$ is in $T$.

- $C$ could have exactly one edge in common with $T$.

- $C$ could have exactly two edges in common with $T$.

(ii) Suppose that $T'$ is a breadth-first search spanning tree of $G$. Give an example of how each of the following can occur or prove that it can never occur.

- All edges of $C$ are in $T'$.

- No edge of $C$ is in $T'$.

- $C$ could have exactly one edge in common with $T'$.

- $C$ could have exactly two edges in common with $T'$.