Instructions: This exam consists of 6 problems, all of equal weight. Please write answers on the exam. If you would like to attach additional pages, include your examinee number on each page. To keep answers concise, it is permissible to refer to algorithms in books and cite page numbers.
1. (a) Design a deterministic finite automaton (DFA) for the language $L \subseteq \{a,b\}^*$, where every string $w$ in $L$ has the same number of $a$’s and $b$’s ($N_a(w) = N_b(w)$) and, for every prefix $x$ of $w$, $N_b(x) \leq N_a(x) \leq N_b(x) + 2$.

(b) If your DFA in part (a) does not have the minimum number of states, design a minimum-state DFA for the same language $L$ and prove that the number of states is minimum. **Note:** It is not sufficient to go through the steps of a minimization algorithm. Your proof needs to be based directly on the Myhill-Nerode theorem.
2. Let \( b_i \) be the binary representation of the integer \( i \) with no leading zeros (for example, \( b_5 = 101 \)). Design a context-free grammar for the language \( L \subseteq \{0,1,\#\}^* \), where every \( w \) in \( L \) has the form \( b_i\#REV(b_{i+1}) \) for some \( i \geq 1 \) (so, for example, \( 101\#011 = b_5\#REV(b_6) \) is in \( L \)). For every variable in your grammar other than the start symbol, give a precise description of the set of strings generated from that variable. That is, if \( A \neq S \) is a variable, give a description (using precise English and/or appropriate notation) of \( \{w \mid A \Rightarrow^* w\} \).
3. Prove that the following question (decision problem) about Turing machines is undecidable: Given a one-tape Turing machine $M$ with tape alphabet $\Gamma = \{0, 1\}$, does $M$, when started with a blank input tape, ever reach a configuration in which the tape has $1^p$, for some prime number $p$, as a substring? Hint: 2 is a prime number, but 1 is not.
4. The hierarchy of Acme Widget Company can be described as a tree structure in which each node represents an employee and the parent of a node represents the immediate supervisor of that employee. Each employee also has a *congeniality index* (CI), a measure of how well they interact socially.

Your task, as an outside consultant, is to come up with an efficient algorithm to select employees to attend a year-end party. The goal is to maximize the total congeniality index of the employees present subject to the constraint that an employee and his/her immediate supervisor are not both invited. Describe the algorithm using high-level pseudocode and give the execution time as a function of the number of employees $n$.

Hint: Work bottom up in the hierarchy, calculating the maximum total congeniality for each subtree, both with and without inviting the employee at the root.
5. Merge sort is an optimal sorting algorithm for small input lists in the sense that it performs no more comparisons than any sorting algorithm that uses only comparisons to access the keys (i.e. its number of comparisons matches the general lower bound for comparison-based sorting algorithms).

(a) Complete the following table, which shows the exact worst-case number of comparisons for merge sort versus the exact worst-case number of comparisons of the sorting lower bound.

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>merge sort</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lower bound</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Find the smallest value of $n$ for which the merge sort entry in the table exceeds the lower bound and come up with a permutation of the integers 1 through $n$ on which merge sort has the worst case performance shown in your table.
6. Let \( n \) be a large fixed integer. Design a data structure that maintains \( n \) integer values \( x_1, \ldots, x_n \) and supports the following operations:

- \( \text{changeKey}(i, dx) \) sets \( x_i = x_i + dx \), no return value
- \( \text{sum}(i,j) \) no side effects, returns \( \sum_{i \leq k \leq j} x_k \)

Describe your data structure and give detailed pseudocode for each of the operations. Each operation should have execution time in \( O(\log n) \) in the worst case. Hint: Consider a balanced binary tree with the values \( x_1, \ldots, x_n \) stored at the leaves. Since \( n \) is fixed, the tree can be constructed in advance and its structure does not need to be modified by the operations.