COMP 271 Design and Analysis of Algorithms
Fall 2008 Final Exam Solutions

1. 1.1 (c) 1.2 (d) 1.3 (b) 1.4 (a)

2. **[OMITTED FROM SYLLABUS]**
   2.1 (a) 2.2 (b) 2.3 (a) 2.4 (b)

3. Scan the middle row and find the minimum number x. Check if x is smaller than both his neighbors above and below it. If so we return x as a local minimum and terminate the algorithm. Otherwise at least one of these two neighbors is smaller than x. Without loss of generality assume the neighbor above x is smaller than x. Then in the upper half of the array, there is at least one number that is smaller than any number of in the middle row, therefore the upper half must contain at least one local minimum. Next we recursively solve the problem in the upper half. When we go into the recursions, we alternate between considering the middle row and the middle column.

4. First sort the graph topologically. For each vertex u, let $d[u]$ be the length of the longest path from s to u. We have the recurrence: $d[s] = 0, d[u] = \infty$ for any u before s in the topological order, and $d[u] = \max_{v,(v,u)\in E} d(v)+1$. Then fill the array d in the topological order. To find the path, keep another array $b[u]$, which records the previous neighbor of u on the longest path from s to u.

5. Let $o[i]$ be the length of the longest oscillating subsequence that ends at a_i and has an odd length; let $e[i]$ be the length of the longest oscillating subsequence that ends at a_i and has an even length. We have the recurrence: $o[1] = 1, o[i] = \max\{0, \max_{j<i, a_j>a_i} e[j]\} + 1, e[i] = \max\{-\infty, \max_{j<i, a_j<a_i} o[j]\} + 1$. Then we compute $o[1], e[1], o[2], e[2], \ldots$. The final answer is $\max_i\{o[i], e[i]\}$.

6. Put the first base station at $x + 4$ where x the coordinate of the first house. Remove all the houses that are covered and then repeat. Correctness: Let X be the solution returned by this greedy algorithm, and let Y be an optimal solution. Consider the first base station where Y is different from X. Suppose the base station in X is located at x and the one in Y is located at y. By the greedy choice, we must have $x > y$. Now move y to x in Y. The resulting Y still covers all houses. Repeated applying this transformation will convert Y into X.

7. **[OMITTED FROM SYLLABUS]**
   (a) The certificate is simply the set of objects S. (b) For any input to Partition $a_1, \ldots, a_n$, we set $w_i = v_i = a_i$, $V = W = \frac{1}{2}\sum_{i=1}^n a_i$. Correctness proof: If Partition is "yes", then the set S is just B. If 0-1 Knapsack is "yes", then for the set S we have $\sum_{o_i\in S} a_i \leq V$ and $\sum_{o_i\in S} a_i \geq V$, so $\sum_{o_i\in S} a_i = V$. Thus B is simply the set S. (c) The running time of the DP is exponential in the input size of 0-1 Knapsack.