1. The degree of a vertex of a graph is the number of neighbors of that vertex. The degree of a graph is the maximum degree of any vertex of the graph. Suppose that $G$ is a connected graph with $n$ vertices, whose degree and diameter are both $d$. Prove that $d = \Omega \left( \frac{\log n}{\log \log n} \right)$.

2. Solve the recurrence:

$$F(n) = \frac{n}{\log n} F(\log n) + n$$

3. Use amortized analysis to prove that the following pseudo-code takes $O(k)$ time to execute. The function `Unknown` has Boolean type, and returns whatever the adversary decides each time it is called.

```plaintext
1: i = 0
2: for j = 1 to k do
3:   while i > 0 and Unknown do
4:     i = i - 1
5:   end while
6:   i = i + 1
7: end for
```

4. Using the result from Problem 3, prove that the REDUCE subroutine of SMAWK takes $O(n + m)$ time to reduce the case of an non-strictly monotone matrix, for $m \geq n$, to the case of an non-strictly monotone matrix.

5. Give a solution to the range query problem of size $n$ that takes $O(n \log^* n)$ preprocessing time and $O(1)$ time for each query, and which uses $O(n \log^* n)$ space.

6. Given a sequence of integers, it is possible to find the longest monotone increasing subsequence in $O(n \log n)$ time, where $n$ is the length of the sequence. Explain how that is done.

   Walk through your algorithm for the example sequence:

   At each step, illustrate the current state of the data structure.

7. Explain Johnson’s algorithm for the all-pairs shortest path problem on a sparse weighted directed graph of $n$ nodes and $m$ edges where there are no negative cycles (although there could be negative edges). Give the asymptotic time complexity in terms of $n$ and $m$. 

8. Prove that the monotone Boolean circuit problem is $\mathcal{P}$-complete, assuming that the Boolean circuit problem is $\mathcal{P}$-complete.

9. Prove that the tropical product of two Monge matrices is Monge.

10. Give an $O(n^2)$-time algorithm which computes the tropical product of two non-Monge matrices.

11. Consider a $3 \times n$ weighted grid graph, $G$. Let $s$ be the upper left corner node. Prove that there is a parallel algorithm in the CREW PRAM model that solves the single-source minpath problem in $G$, where the source node is $s$, with time complexity $O(\log n)$ using $\frac{n}{\log n}$ processors. You don’t have to actually give the algorithm. You only have to convince me that the algorithm exists.

   Be careful. The least cost path from $s$ to the opposite corner could be a path of length $3n + 1$ that uses every node.

12. Solve the following recurrences.

   (a) $F(n) = F\left(\frac{n}{2}\right) + 6F\left(\frac{n}{4}\right) + 3F\left(\frac{n}{6}\right) + n^2$

   (b) $F(n) = F\left(\frac{4n}{9}\right) + F\left(\frac{n}{9}\right) + 1$

   (c) $F(n) = F\left(\frac{4n}{9}\right) + F\left(\frac{n}{9}\right) + n$

13. The following 0-1 problems are all obviously in the class $\mathcal{NP}$.

   P1: SAT.
   P2: 3-SAT.
   P3: 3-in-1 SAT.
   P4: Independent Set.
   P5: Clique.
   P6: Knapsack.
   P7: Traveling Salesman.
   P8: Dominating Set.
   P9: Integer Programming.

   One of the following will be on the test:

   (a) Assuming that P1, P2, and P3 are $\mathcal{NP}$-complete, prove that P4 is $\mathcal{NP}$-complete.

   (b) Assuming that P1, P2, P3, P4, and P5 are $\mathcal{NP}$-complete, prove that P6 is $\mathcal{NP}$-complete.

   (c) Assuming that P1, P2, P3, P4, P5, P6 and P7 are $\mathcal{NP}$-complete, prove that P8 is $\mathcal{NP}$-complete.