1. Greg is asked to give a definition of NP. Here is Greg’s answer: NP is a class of languages; a language \( L \subseteq \Sigma^* \) belongs to NP if and only if
\[
(\exists \text{ finite alphabet } \Sigma_1)(\exists L_1 \subseteq \Sigma_1^*)(L_1 \in \mathbb{P} \text{ and } (\forall x \in \Sigma^*)(x \in L) \iff (\exists w \in \Sigma_1^*)(x, w) \in L_1)).
\]
(a) (6 points) Prove that instead of NP, Greg defined a vastly larger complexity class; let’s call it GREG. Prove: HALTING \( \in \) GREG.
(b) (4 points) Modify Greg’s definition to get a correct definition of NP. Make as few changes as possible. (Very little needs to be changed.)
(c) (G only, 3 points) What exactly is GREG?
(d) (4 points) Prove that HALTING is NP-hard by giving a Karp-reduction from 3-COL to HALTING.

2. (2+5 points) Alice and Bob want to communicate using public-key cryptography. Which of them needs to publish a key (a) if Alice wants to send an encrypted message to Bob; (b) if Alice wants to send Bob a digitally signed document? Describe, in case (b), how the scheme is used to sign the document and why it will work if Bob wants to take Alice to court?

3. Let FACT denote the decision version of the factoring problem discussed in class.
   (a) (2 points) Define FACT.
   (b) (5 points) Suppose FACT can be recognized in time \( T(n) \). Prove that then, \( n \)-digit integers can be factored into their prime factors in time \( O(n^cT(n)) \). Estimate the constant \( c \).
(c) **(G only, 5 points)** Prove that if FACT ∈ coNPC then NP = coNP. 
You may use the old result that the set of primes is in NP but not 
the recent result that it is in P.

*Comment added after the test.* “coNPC” was a typo, it was in-
tended to be NPC. However, the problem stands as stated (it is 
easier than the problem originally intended).

4. **(Branch-and-bound) (4+3+3 points)** (a) Describe, in pseudocode, an 
algorithm to find α(G), the maximum size of independent sets in the 
graph G, in time O(c^n), where c < 2 is a constant; make the constant 
as small and possible. (b) State the recurrent inequality satisfied by 
the time complexity of the algorithm. (c) Analyse the recurrence; state 
the equation satisfied by c. You do not need to calculate c.

5. (a) **(3 points)** Assuming 3-COL ∈ NPC, prove that 4-COL ∈ NPC. 
(k-COL is the set of k-colorable graphs.)

(b) **(G only, 5 points)** Assuming 3-SAT ∈ NPC, prove that (0,1)-ILP 
∈ NPC.

6. **(G only, 3+3+3+4 points)**

(a) Define how DFS classifies the edges of a digraph.

(b) State the “white path theorem.”

(c) State the “topological sort” problem (input-output).

(d) Give a linear-time algorithm which takes a digraph as input and 
returns either a topological sort or a directed cycle.

7. **(4 points)** Describe in pseudocode the algorithm that computes the 
number of connected components of an undirected graph in linear time. 
Do not include the code for BFS.

8. **(3 points)** Recall that a digraph is strongly connected if every vertex is 
accessible from every vertex. State a very simple algorithm that decides 
in linear time whether or not a graph is strongly connected. You may 
use BFS as a building block but any other algorithm you use, whether 
discussed in class or not, you need to describe in pseudocode.
9. (G only, 4 points) Let \( f_1, f_2, \ldots \) be functions, \( f_k : \mathbb{N} \to \mathbb{R} \). (\( \mathbb{N} \) is the set of nonnegative integers.) True or false: if for all \( k \), \( f_k(n) = O(n) \) then \( \sum_{k=1}^{n} f_k(n) = O(n^2) \). Prove your answer.

10. Recall that Prim’s algorithm solves the min-cost spanning tree problem by growing the tree from a vertex.

   (a) (2 points) State the three data structure operations required for an implementation of Prim’s algorithm.

   (b) (2 points) State the number of times each operation is called in terms of \( n = \) the number of vertices and \( m = \) the number of edges of the input graph.

   (c) State the total cost under (a) (2 points) the array implementation (b) (2 points) the heap implementation (c) (G only, 2 points)) the Fibonacci tree implementation of the required priority queue datastructure.

   (d) (2 points) Which implementation is best for dense graphs (graphs with \( \Theta(n^2) \) edges)?

11. (G only, 5 points) Given the \( n \)-digit integers \( k \) and \( m \), calculate \( (F_k \mod m) \) in polynomial time. (\( F_k \) is the \( k \)-th Fibonacci number.) Hint: use \( 2 \times 2 \) matrices.