There are six problems on the exam, each worth 20 points; you can **pick any five** to complete. If you do all six, the problem with the lowest score will be dropped. There is also an extra credit problem which will count for some unspecified number of points $0 < p < 10$.

In preparing your solutions to the exam, you are allowed to use any sources, including your textbook, other students and professors, previous homeworks and solutions, or any sources on the Internet. You may ask me for feedback on potential solutions, but I will not give you any hints.[1] Of course, I am also happy to answer general questions, go over homework problems, or answer clarifying questions about exam problems.

[1] Think of me as a polynomial certifier for exam problems.

The exam will take place in MC Reynolds 317 from 2–5pm on Monday, May 8. You are not allowed to bring any notes, textbooks, calculators, or any other resources with you to the exam. Bring only something to write with; I will provide a fresh copy of the exam, paper for writing your solutions, and scratch paper.

As usual, to "design and analyze" an algorithm means to (a) describe the algorithm, (b) prove/justify its correctness, and (c) analyze its asymptotic running time. Full credit will only be given for the most efficient possible algorithms. Algorithms must be clearly explained (using pseudocode if appropriate) in sufficient detail that another student could take your description and turn it into working code. You may freely cite any theorems proved in class (without proof), or use algorithms covered in class as subroutines.

Question 1 (20 points)**.** You are driving from Conway to Millinocket, Maine, to attend your third cousin's wedding. You are on a budget, so you want to spend the least amount of money possible. You have a map which shows all the cities in the whole country as well as all the roads connecting the cities. Each road has a cost (*e.g.* the amount of money you would spend on gas, tolls, bribes, *etc.* to drive on that road). Note, however, that some roads are one-way and some are two-way; two-way roads may have a different cost in each direction (for example, driving *up* a road into the mountains uses more gas than driving *down* the same road). Each city also has a cost: for each city you know (from past experience) exactly how much money you would spend there, *e.g.* on food, amusement parks, and other local attractions. Although you are on a budget, you are also impulsive; you always stop in each city you come to, and the only way you can avoid spending money in a city is to avoid visiting it altogether. Design and analyze an algorithm to find the cheapest route from Conway to Millinocket.

Figure 1: Millinocket

Question 2 (20 points)**.** Your car has no parking brake,[2] so every time you stop along your journey you need to park in a place where the ground slopes upward in both directions, so your car will not roll away. More formally, given an array $A[0 \ldots n-1]$ of real numbers representing the altitude in feet of each position along a road, position i is a safe place to park your car if and only if $A[i-1] > A[i] < A[i+1]$. (Note that $i = 0$ and $i = n-1$ are never considered safe, since we have no data about what lies beyond them!)

[2] It was stolen by a moose the last time you visited Millinocket.

Assume that a given array A describes a road which starts out going downwards, has exactly one safe parking place, and then goes back up. That is, $A[0] > A[1] > \cdots > A[i] < \cdots < A[n-2] < A[n-1]$. Design and analyze an algorithm to find the safe parking place when given such an array A.

Question 3 (20 points)**.** A palindrome is any string that is equal to its reversal, like `I`, or `DEED`, or `RACECAR`, or `AMANAPLANACATACANALPANAMA`.

Design and analyze an algorithm to find the length of the longest subsequence of a given string that is also a palindrome. For example, the longest palindrome subsequence of

`MAHDYNAMICPROGRAMZLETMESHOWYOUTHEM`

is `MHYMRORMYHM`, so given that string as input, your algorithm should output the number 11.

Figure 2: `AMANAPLANACATACANALPANAMA`

Question 4 (20 points)**.** Let X be a set of n intervals on the real line. A *proper coloring* of X assigns a color to each interval so that any two overlapping intervals are assigned different colors. Figure 3 shows an example of a proper coloring. Design and analyze an algorithm to compute the minimum number of colors needed to properly color X, with a running time better than $\Theta(n^2)$. Assume that your input consists of an array $I[1..n]$, where each item in the array is a pair of real numbers $I[i].left$ and $I[i].right$ specifying the endpoints of interval i.

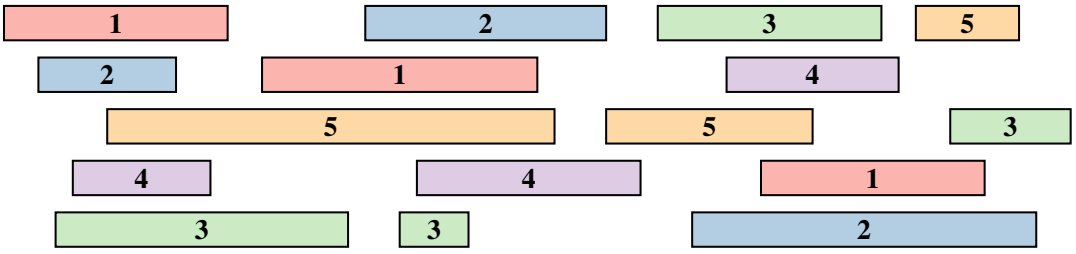

Figure 3: A proper coloring of a set of intervals

Question 5 (20 points)**.** Suppose that instead of powers of two, we represent integers as a sum of distinct Fibonacci numbers. In other words, instead of an array of bits, we keep an array of *fits* (*fi*bonacci dig*its*), where each fit is either 0 or 1, and the i^{th} least significant fit indicates whether the sum includes the i^{th} Fibonacci number F_i. For example, the fit-string 101110_F represents the number

$$F_6 + F_4 + F_3 + F_2 = 8 + 3 + 2 + 1 = 14.$$

Design and analyze an algorithm to increment a fit-string in $\Theta(1)$ amortized time, assuming that a fit-string counter starts at 0 and is repeatedly incremented. As with the example above, assume that $F_1 = F_2 = 1$, that the array of fits is indexed starting from 1, and that the i^{th} index of the array stores the i^{th} fit. Also observe that a given number n does not necessarily have a unique representation as a fit-string, but it does not matter what representation your algorithm uses.

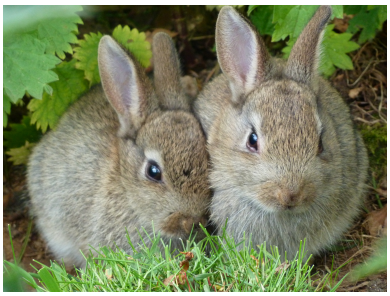

Figure 4: F_3

Question 6 (20 points)**.** A *dominating set* for an undirected graph $G = (V, E)$ is a set of vertices $D \subseteq V$ such that for every vertex $v \in V$, either

1. $v \in D$, or
2. v is adjacent to at least one member of D.

For example, the marked black vertices form a dominating set in the graph at the top of Figure 5; every other vertex is adjacent to at least one of the marked vertices. On the other hand, the marked vertices in the bottom graph do not form a dominating set, since the lower-right vertex is not adjacent to any marked vertex.

The DOMINATING SET problem asks, given a graph G and a positive integer k, does G have a dominating set of size $\leq k$? Prove that DOMINATING SET is NP-complete.

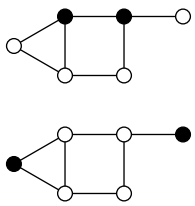

Figure 5: Examples of dominating and non-dominating sets

Question 7 (Extra credit)**.** For your birthday, your fairy godmother has given you a strange gift: a mysterious black box with a slot on one end and two lights on top, one green and one red. When you put a drawing of an undirected graph into the slot, one of the two lights comes on: green if the graph is 3-colorable, and red if it isn't.[3] Curiously, a light always comes on right away, no matter how many vertices and edges the input graph has.

Describe a polynomial-time algorithm, making use of your magic box, which takes a graph G as input and constructs a valid 3-coloring for G (or correctly reports that G is not 3-colorable).

[3] Your fairy godmother didn't actually tell you this; how you figured it out is an exciting tale of adventure, involving a circumnavigation of the globe, a talking unicorn named Charles, and a list of all the real numbers, but unfortunately the tale is too long to fit on this exam.

Acknowledgment Questions 3, 4, and 5 are adapted from Jeff Erickson (`http://jeffe.cs.illinois.edu/teaching/algorithms/`).