Ground rules:

- This is an oral presentation assignment. You should work in groups of three. At some point before March 20, your group must sign up for a 1-hour time slot on the signup sheet on the course web page.
- Please write up (and turn in during your presentation) the parts below that specifically require a write-up, as indicated below.
- During the presentations, you may be asked to explain why your algorithm is correct.
- This is a fairly long assignment – so it’s recommended that you start to think about it soon.

Problems:

1 The Triage Before Spring Break

Read problem 20 on page 329 of the textbook.

(a) Solve it, and write-up your algorithm formally in pseudo-code.

(b) Now suppose for each course $i$ you have a minimum desired grade, $g_i$, which is an integer. Your new goal is to maximize the average grade you receive (again as measured by the functions $f_i$) subject to the constraint that you get a grade of at least $g_i$ in course $i$. If there is no way to divide your work that satisfies these constraints, your algorithm should indicate this, e.g. by returning “instance is infeasible”. Write-up your algorithm formally in pseudo-code.

2 The Spring Break Cruise

It’s spring break and you are on a cruise in the Caribbean. There is a set of activities $A$ on the cruise, which you’ve read about. Furthermore, some of the activities happen at the same time, and you can only do one activity at a time. We’ll model this by partitioning $A$ into $\{A_1, A_2, \ldots, A_k\}$ where $A_i$ are the activities that are at time $i$. Thus you can do at most one activity in each $A_i$. For each activity $a \in A$, you have rated it for fun value, denoted $f(a)$. You’ve also estimated how much sun exposure you’ll get if you go to event $a$, say $s(a)$. Now, getting just the right tan is essential to you, so you also have minimum and maximum levels of sun exposure, which we’ll call $s_{\text{min}}$ and $s_{\text{max}}$. All values are non-negative integers. Your goal is select a set of activities $I$ to go to such that your total fun $f(I) := \sum_{i \in I} f(i)$ is maximized, subject to the constraint that you get the right tan (i.e. $s(I) \in [s_{\text{min}}, s_{\text{max}}]$). Of course, you also need to ensure that you only include at most one activity per time slot (i.e. $|I \cap A_i| \leq 1$).

(a) Give an algorithm for this problem, whose running time is polynomial in $|A|$ and $s_{\text{max}}$. Write-up your algorithm formally in pseudo-code.

(b) Now suppose each activity $a \in A$ costs some amount of money $c(a)$ (a non-negative integer) and you have $B$ dollars you’re willing to spend on activities. Give an algorithm that maximizes $f(I)$ under the constraints that $s(I) \in [s_{\text{min}}, s_{\text{max}}]$ and $c(I) \leq B$. Its running time should be polynomial in $|A|$, $s_{\text{max}}$, and $B$. Write-up your algorithm formally in pseudo-code.
3 The Spring Break Road trip

While you are on your cruise, your friends are going on a road trip to San Francisco immediately after midterms. To plan the trip, they have laid out a map of the U.S., and marked all the places they think might be interesting to visit along the way. However, the requirements are:

1. Each stop on the trip must be closer to SF than the previous stop.
2. The total length of the trip can be no longer than $D$.

They want to visit the most places possible subject to these conditions. Unfortunately, they don’t have your algorithmic prowess, and ask for your help in planning their trip.

As a first step, you create a DAG with $n$ nodes (one for each location of interest) and an edge from $i$ to $j$ if there is a road from $i$ to $j$ and $j$ is closer to SF than $i$. Let $d_{ij}$ be the length of edge $(i, j)$ in this graph.

Help out your friends by giving an $O(mn)$-time algorithm to solve their problem. Specifically, given a DAG $G$ with lengths on the edges, a start node $s$, a destination node $t$, and a distance bound $D$, your algorithm should find the path in $G$ from $s$ to $t$ that visits the most intermediate nodes, subject to having total length $\leq D$. Write-up your algorithm formally in pseudo-code.

(Note that in general graphs, this problem is NP-complete: in particular, a solution to this problem would allow one to solve the traveling salesman problem. However, the case that $G$ is a DAG is much easier.)