Introduction to Algorithms: 6.006
Massachusetts Institute of Technology — April 11, 2018
Instructors: Srini Devadas, Jason Ku, and Nir Shavit — Problem Set 8

Problem Set 8

All parts are due on April 19, 2018 at 11PM. Please write your solutions in the LaTeX and Python templates provided. Aim for concise solutions; convoluted and obtuse descriptions might receive low marks, even when they are correct. Solutions should be submitted on the course website, and any code should be submitted for automated checking on `py.mit.edu/6.006`.

Problem 8-1. [30 points] **Shortest Paths**

(a) [10 points] **Dijkstra Practice:** Below is is a list of triples representing a weighted directed graph G on 8 integer labeled vertices. A triple (i, j, w) represents a directed edge in the graph from vertex v_i to vertex v_j having weight w.

```
G = [(1, 3, 11),
     (1, 5, 34),
     (2, 5, 14),
     (3, 5, 10),
     (3, 8, 22),
     (4, 7,  5),
     (5, 6, 47),
     (5, 7, 18),
     (6, 1,  3),
     (7, 2,  7)]
```

- [2 points] Draw graph G by placing vertices to lie roughly on a circle ordered clockwise by index. Be sure to label edge weights!
- [8 points] Run Dijkstra to compute the shortest path distance $\delta_s(v_i)$ from source vertex $s = v_1$ to every vertex v_i in the graph.

(b) [5 points] **Longest Paths:** A path in a graph is **simple** if it visits any vertex at most once. Fellman Bord claims that one can find a longest simple path between two vertices in an arbitrary weighted graph by negating edge weights and running Bellman-Ford once in the modified graph. Argue that Fellman Bord's algorithm correctly finds a longest simple path, or argue that the algorithm is not correct for all inputs.

(c) [15 points] **Longer Shortest Paths:** Given a connected undirected graph $G = (V, E)$ with positive edge weights, describe algorithms to find a minimum weight path (if one exists) between two specified vertices that uses: (1) **exactly** k edges in $O(k|E|)$ time, and (2) **at least** k edges in $O(k|E| + |V| \log |V|)$ time.

Problem 8-2. [30 points] **Consulting**

For each of the following scenarios, apply a graph algorithm that best solves the problem: describe a graph related to the problem, choose or modify an existing algorithm to apply to the graph, justify your choice of algorithm, and state your algorithm's running time in terms of problem parameters.

(a) [10 points] **Rover Scheduling:** NASA scientists need to plan a task schedule for the Mars rover. Each task requires a certain amount of time to complete. One task may be required to complete prior to another. For example, before transmitting soil sample data to Earth, the samples must have been collected, and the antennae battery must have been charged. Given a list of rover tasks including their durations and dependencies, describe an algorithm to compute the shortest time to complete all tasks.

(b) [10 points] **Secret Correspondence:** A network of spies operate in Easteros. Each spy has a list of associates with whom they communicate regularly. Whenever a particular pair of spies meet, there is a known probability that the information discussed will be leaked. The Mother of Turtles needs to send a secret message via this network from her spy in the capitol to a scouting spy at the boarder of the kingdom. When a message is transferred through the network via a sequence of spies, the **risk** of transferring the message is the maximum probability of **any individual pair of spies** in the sequence leaking the information. Assuming the spy network is sufficiently connected to be able to transmit the message, describe an algorithm to determine a sequence of spies to convey the message that minimizes risk.

(c) [10 points] **Purchasing Parts:** Faylee Krye is the mechanic on board a Sirefly-class spaceship named Ferenity. While docked at a spaceport, she heads into town to purchase six parts for the ship: a power converter, a flux capacitor, a hyper drive, an ion shield generator, a binary motivator, and an air freshener. Faylee has downloaded a map of the town including the location of the parts distributors and time estimates to travel long any particular road in town (you may assume that the number of parts distributors is large). Looking down the list of distributors, Faylee observes that each store sells at most one of the parts, so she will need to visit six different stores to buy everything. Assume that the time spent in any store is negligible compared to transit time. Describe an algorithm to find a route for Faylee to travel from the spaceport and back, purchasing all the needed spaceship parts in the minimum amount of time.

Problem 8-3. [40 points] **Crypto-Exchange**

Chad Chaddingsworth is an unlikeable millennial investor who trades crypto-currencies for a living on the BoinCase exchange. The BoinCase exchange allows users to convert between real currencies and crypto-currencies at posted exchange rates that change each day. For example, yesterday Chad purchased 5 CitBoins for 10 US dollars at an exchange rate of 0.5 CitBoins per US dollar. BoinCase exchange rates are always positive, but not necessarily symmetric, so the exchange rate from US dollars to CitBoins might be smaller (or possibly larger) than 2.0. Given a daily list of BoinCase exchange rates, Chad wants to find a sequence of trades that he can perform that day and make money. More specifically, beginning with x US dollars to invest at the beginning of the day, Chad wants to end the day with more than x US dollars by performing a sequence of conversions on the exchange.

(a) [15 points] Design an algorithm to determine whether Chad can make money on the exchange, and if so, provide a sequence of currencies from US dollars to US dollars for which the sequence of conversions makes money. **Hint:** Products can be transformed into sums using logarithms.

(b) [25 points] Write the Python function `make_money(exchange_rate)` that implements your algorithm from part (a). Currencies are labeled with integers from 0 to $n-1$ inclusive, with 0 label of US dollars. The input `exchange_rate` is a length n list of length n lists, where element `exchange_rate[i][j]` corresponds to the exchange rate for converting currency i into currency j on the exchange. Your function should return a list of integer labels between 0 and $n-1$ inclusive, starting and ending with label 0, corresponding to a sequence of currencies that can be converted in order to make money. If no sequence of conversions can make money in this way, then your function should return `None`. Submit your code online at `py.mit.edu/6.006`.