Homework 2
Due Date: February 5, 2019 at 11:59PM on Gradescope.

Question 1 (Greedy Gambling)

The local casino is opening up a new game involving \( n \) treasure chests 1, \ldots, \( n \) in a private room. Before a new gambler enters the room all treasure chests are emptied and then single gold bar of value \( V \) is randomly placed into one of treasure chests, while a worthless rock of equal weight is placed into the other treasure chests. The probability that the gold bar is placed in treasure chest \( i \) is given by \( p_i \). Each treasure chest \( i \) is labeled with the probability \( p_i \) and a cost \( c_i \) indicating that the gambler may pay the key master \( c_i \) to obtain the key to open treasure chest \( i \). If the treasure chest contains the gold bar then the gambler may keep it, otherwise the gambler may choose to exit the room (quit) or buy another key.

A strategy consists of sequence of intended actions \( a_1, \ldots, a_n \) where each action \( a \in [n] \) is of the form e.g., buy the key for chest \( a \) and open it. All of the \( a_i \) values are assumed to be distinct since it does not make sense to buy the same key twice. If box \( a_i \) (\( i \leq n \)) contains the gold bar then the total gain (or loss) is given by \( V - \sum_{j=1}^{i} c_{a_j} \) i.e. since the gambler does not need to pay to open boxes \( a_{i+1}, \ldots, a_k \). Thus, the expected gain is given by

\[
\mathbb{E}[\text{Gain}(a_1, \ldots, a_n)] = V - \sum_{i=1}^{n} p_i \left( \sum_{j=1}^{i} c_{a_j} \right)
\]

Note that the gambler may also play the special strategy \( \text{QUIT} \) (never open any box) which always has gain 0. However, if the gambler opens one treasure chest then the gambler must continue paying to open treasure chests until the gold bar is found. The task of the greedy gambler is to find a strategy which maximizes the expected gain.

1. Suppose the gambler sorts the treasure chests such that \( p_1 \geq p_2 \geq \ldots \geq p_n \), and then outputs the strategy 1, \ldots, \( n \) if and only if \( \mathbb{E}[\text{Gain}(1, \ldots, n)] \geq 0 \) — otherwise we output the strategy \( \text{QUIT} \). Provide a short counter-example showing that the gambler might miss the optimal strategy.

2. Suppose the gambler sorts the treasure chests such that \( c_n \geq c_{n-1} \geq \ldots \geq c_1 \), and then outputs the strategy 1, \ldots, \( n \) if and only if \( \mathbb{E}[\text{Gain}(1, \ldots, n)] \geq 0 \) — otherwise the gambler output the strategy \( \text{QUIT} \). Provide a short counter-example showing that the gambler might miss the optimal strategy.

3. Provide a greedy algorithm to find the optimal strategy and analyze the performance of your algorithm. You should prove that your algorithm is correct.
Question 2

1. Consider a variant of the interval scheduling problem in which every meeting request
   \( j \in \{1, \ldots, n\} \) has a unique finish time \( f_j \in \{1, \ldots, 2n\} \), while start times are arbitrary
   (subject to \( s_j < f_j \)). Design a linear time algorithm to solve this variant of the interval
   scheduling problem.

2. You are given an array \( A \) of \( n \) integers and a parameter. Given integers \( i < j \) define
   \( \text{score}(i, j) = 0 \) if \( |j - i| > t \) and \( \text{score}(i, j) = \sum_{k=i}^{j} A[k] \) otherwise. Find a \( O(n \log t) \)
time algorithm to output a pair \((i, j)\) of integers with maximum possible score. You
   should prove that your algorithm is correct and analyze the running time.

Question 3

\( n \) vertical cylinders \( c_1, \ldots, c_n \) whose length are infinite but the area of the base of each
   cylinder is one square meter i.e. \( 1 \text{ m}^2 \). Two cylinders \( c_i \) and \( c_j \) might be connected with
   a horizontal pipe at height \( h_{i,j} \) (or there might not be any pipes between them). Suppose
   there is a tap in the first cylinder \( c_1 \). We turn on the tap at time 0 which pours down one
   cubic meter of water per hour into the first cylinder \( c_1 \) i.e., the flow rate is \( 1 \text{ m}^3/\text{hr} \). If in
   an arbitrary cylinder \( c_i \) water reaches the height of a pipe \( h_{i,j} \) (and the water in cylinder \( j \)
   has not yet reached the height \( h_{i,j} \)) then water flows inside the pipe and reaches the cylinder
   connected to \( c_i \) through that pipe (we assume that it takes a negligible time that water
   passes through any pipe and gets to the other cylinder). Give an algorithm that outputs
   the first time \( t_i \) that water reaches each cylinder \( c_i \). As a simple example, suppose \( n = 3 \)
such that \( c_1 \) is connected to \( c_2 \) with a pipe at height \( h_{1,2} = 1 \), \( c_1 \) is connected to \( c_3 \) at height
   \( h_{1,3} = 2 \) and \( c_2 \) is connected to \( c_3 \) at height \( h_{2,3} = 4 \). Then, we observe that water reaches
   \( c_1, c_2 \) and \( c_3 \) at times \( t_1 = 0, t_2 = 1 \) and \( t_3 = 3 \). You should prove that your algorithm is
   correct, and analyze its running time.

Question 4

Consider the following suggested algorithms for finding a minimum spanning tree in a con-
   nected weighted graph (\( w \) is considered as the weight function). Which algorithm is correct?
   (For each algorithm, you need to prove that its output is a minimum spanning tree or give
   a graph as a counterexample for which the output of algorithm is not a minimum spanning
   tree). Moreover, for each algorithm explain how to implement it in the most efficient way
   even if it is not correct and does not give us necessarily a minimum spanning tree.

1. Algorithm A
   
   sort the edges into non increasing order \( (w(e_1) \leq w(e_2) \cdots \leq w(e_m)) \)
   
   Initialize \( T \) to be equal to the set of all edges \( E \)
   
   for \( i = 1 \) to \( m \) do
      
      if \( T - \{e_i\} \) is a connected graph then
         
         \( T = T - \{e_i\} \)
      
   end if
   
   end for
2. Algorithm B
Initialize $T = \emptyset$
consider some arbitrary order on edges $\{e_1, \ldots, e_m\}$
for $i = 1$ to $m$
do
if there is no cycle in $T \cup \{e_i\}$ then
$T = T \cup \{e_i\}$
end if
end for
return $T$

3. Algorithm C
Initialize $T = \emptyset$
consider some arbitrary order on edges $\{e_1, \ldots, e_m\}$
for $i = 1$ to $m$
do
$T = T \cup \{e_i\}$
if there is a cycle $C$ in $T$ then
let $e'$ be an edge with maximum weight on cycle $C$
$T = T - \{e'\}$
end if
end for
return $T$