Problem 1. We have shown that weak duality holds for standard LPs of the form

$$\min c^\top x \qquad \text{s.t. } Ax = b,\ x \geq 0$$

Suppose instead that our LP involves all types of constraints (equality, lower-bound, and upper-bound). That is, for an LP of the form

$$\begin{aligned}
&\min c^\top x \\
&\text{subject to} \\
&\sum_j A_{ij} x_j \leq b_i \qquad \text{for } i \in 1, \ldots, k_1 \\
&\sum_j A_{ij} x_j \geq b_i \qquad \text{for } i \in k_1 + 1, \ldots, k_2 \\
&\sum_j A_{ij} x_j = b_i \qquad \text{for } i \in k_2 + 1, \ldots, m \\
&x_i \geq 0 \qquad \text{for all } i
\end{aligned}$$

the corresponding dual will be

$$\begin{aligned}
&\max \sum_{i=1}^{k_1} b_i y_i + \sum_{k_1+1}^{k_2} b_i y_i' + \sum_{k_2+1}^{m} b_i y_i'' \\
&\text{subject to} \\
&\sum_{i=1}^{k_1} a_{ij} y_i + \sum_{k_1+1}^{k_2} a_{ij} y_i' + \sum_{k_2+1}^{m} a_{ij} y_i'' \geq c_j \\
&\qquad\qquad y_i \leq 0 \qquad \text{for } i \in 1, \ldots, k_1 \\
&\qquad\qquad y_i' \geq 0 \qquad \text{for } i \in k_1 + 1, \ldots, k_2 \\
&\qquad\qquad y_i'' \quad \text{UIS} \quad \text{for } i \in k_2 + 1, \ldots, m
\end{aligned}$$

Problem 2. Rewriting the max-flow linear program in matrix form, we want to find

$$\begin{aligned} z = \max & \sum f_P \\ \text{subject to} & \\ \mathbf{f}^\top A &\leq \mathbf{u} \\ f_P &\geq 0 \qquad \text{for all paths } P \end{aligned}$$

where $\mathbf{f} = [f_1, \ldots, f_p]$ is the vector of path flows, A is a $P \times m$ boolean matrix where each row represents a path and each column represents an edge (such that A_{ij} is 1 if the i-th path includes edge j and 0 otherwise), and $\mathbf{u} = [u_1, \ldots, u_m]$ is the vector of edge capacities.

In other words, we want to maximize $\mathbf{f} \cdot \mathbf{1}$ such that $\mathbf{f}^\top A \leq \mathbf{u}$ and $f_p \geq 0 \; \forall \; P$. If we consider this a dual, then the corresponding primal (which is the dual of the dual) is:

$$\min \mathbf{u}^\top \mathbf{x} \quad \text{s.t.} \quad A\mathbf{x} \geq \mathbf{1}, \; x_i \geq 0$$

where $\mathbf{x}$ is a vector of length m. Looking at the matrix product constraint, we see that the dot product of each row of A (corresponding to a path P) and $\mathbf{x}$ must be at least 1, which means we can rewrite the new linear program as

$$\begin{aligned} w = \min & \sum_{i=1}^{m} u_i x_i \\ \text{subject to} & \\ \sum_{e \in P} x_e &\geq 1 \qquad \text{for all paths } P \\ x_e &\geq 0 \qquad \text{for all edges } e \end{aligned}$$

If we think of the x_e as the "weights" of the edges, then the first constraint says that all paths must have at least 1 weight, and the second constraint says that all edges must have non-negative weight. This is essentially the min-cut problem: we want to find the minimum cut that spans all possible paths (i.e. splitting G into two disjoint sets) while minimizing the cost of the edges crossing the cut. Thus, by setting $x_e = u_e$ for edges crossing the cut and $x_e = 0$ otherwise, the dual linear program will try find the minimum s-t cut in the network graph.

Problem 3. (a) The first constraint ensures that flow is conserved at every node in the graph. The second constraint ensures that the total flow in the graph is equal to 1. Lastly, the third constraint ensures that each flow f_{ij} is non-negative. Under these constraints, the linear program seeks to minimize $\sum c_{ij} f_{ij}$, i.e. the total cost of the flow in the graph. The objective function ensures that the optimal solution of the linear program under these constraints results in a decomposition of the flow into minimum mean cycles. To show that this is true, take any two cycles in $\{f_{ij}\}$ with total cost c_1, c_2 and length l_1, l_2. WLOG, suppose the mean of the first cycle is less than that of the second, so $\frac{c_1}{l_1} < \frac{c_2}{l_2}$. Then any flow f through these cycles will result in $\frac{c_1}{l_1} f < \frac{c_2}{l_2} f$, so the cycle with the lesser mean will always result in the lower contribution to the objective function. Thus, the linear program correctly finds a circulation that consists of minimum mean cycles.

(b) The dual linear program is as follows:

$$\begin{aligned} z &= \max \lambda \\ \text{subject to} & \\ y_j - y_i + \lambda &\leq c_{ij} \qquad \text{for all } i, j \\ y_i &\quad \text{UIS} \qquad \text{for all } i \\ \lambda &\quad \text{UIS} \end{aligned}$$

(c) We can rewrite the constraint $y_j - y_i + \lambda \leq c_{ij}$ as $(c_{ij} - \lambda) + y_i - y_j \geq 0$, which corresponds to the reduced cost of edge (i, j) if we remove λ from the cost of the edge. In other words, the constraint seeks to determine the maximize λ we can remove from all the edges such that the price function is still feasible for the graph. Notice that if we sum the reduced costs of the edges in the graph, each $-y_j$ term is cancelled out by the following y_i term, so the telescoped sum is $\sum_{i,j} c_{ij} - \lambda$. Thus, for the minimum mean cycle C its sum must be equal to 0 (since otherwise we could just increase λ to further minimize the reduced costs), so we have

$$\sum_{(i,j) \in C} c_{ij} - \lambda = 0$$

This implies $c = l\lambda$, where $c = \sum_{(i,j) \in C} c_{ij}$ and l is the length of the cycle. Therefore, since the minimum mean cycle bounds λ at $\frac{c}{l}$, it follows that by maximizing λ to reach this vertex, the dual LP also solves the minimum mean cycle problem.

(d) Since c_{ij} are integers, we can use binary search to find the optimal λ over a search area. On each iteration, we create a temporary graph G' with λ subtracted from all the edge costs, and look for the presence of negative cost cycles). The search space is $\lambda \in [0, \max(c_{ij})]$. Also, we know that the difference between the smallest and next smallest mean costs for two cycles with total cost c_1, c_2 and length l_1, l_2 is bounded by

$$\frac{c_1}{l_1} - \frac{c_2}{l_2} = \frac{c_1 l_2 - c_2 l_1}{l_1 l_2} \leq \frac{1}{m^2}$$

since the minimum difference is 1 and the maximum length of a cycle is m, the number of edges in the graph. Thus, we can end the binary search once our window becomes smaller than $\frac{1}{m^2}$, knowing that we have found the optimal λ. Since each iteration takes $O(mn)$ time to detect cycles, the total runtime of this algorithm is $O(mn \log C)$ where $C = \max(c_{ij})$.

Problem 4. (a) We have that for a primal linear program in standard form,

$$\min c^\top x \qquad \text{s.t. } Ax = b,\ x \geq 0$$

the corresponding dual is

$$\max b^\top y \qquad \text{s.t. } A^T y \leq c$$

From strong duality, we also know that the two optima are equal. Thus, we can formulate a linear program

$$\begin{aligned} c^\top x &= b^\top y \\ \text{subject to} & \\ Ax &= b \\ A^\top y &\leq c \\ x &\geq 0 \end{aligned}$$

Then, solving the original LP is reduced to checking whether this new LP is feasible (since any feasible solution will satisfy the objective, which is the optimal solution for both the original and dual). Lastly, we can convert the new LP to standard form by rewriting $c^\top x = b^\top y$ as $c^\top x - b^\top y = 0$ and $A^\top y \leq c$ as $A^\top (y^+ - y^-) + s = c$, leaving us with a form that matches $\min 0 \cdot x$ s.t. $Ax = b, x \geq 0$.

(b) The dual is

$$\max b^\top y \qquad \text{s.t. } A^T y \leq 0,\ y \text{ UIS}$$

(c) If the primal is feasible, then its optimal objective value is 0 (otherwise, if the primal is infeasible, then its optimal objective value is ∞). By strong duality, the dual also has an optimal objective at 0, which is achieved by $y = 0$. Therefore, this is the optimum solution.

(d) We can devise an algorithm as follows:

- Given the input LP, convert it to the form described in part (a).
- Optimize this new LP using the oracle algorithm given to us (and the fact that there is a trivial dual solution as shown in part (c)).
- Convert the optimum of the new LP back to its original form.

Since transforming the input LP to and from the form described in part (a) takes time relative to the size of the matrix $O(mn)$, the main factor is the oracle call. Thus, the overall runtime of the algorithm is $O((m+n)^k)$.

(e) In class, we showed that from the basis of a primal optimum it is possible to quickly construct a dual optimum. However, the reverse is not necessarily true; knowing the dual optimum may not assist with constructing a primal optimum efficiently, as shown in the given algorithm.

Problem 5. (a) Bob wants to solve $\max_{\sum y_j=1} \min_{\sum x_i=1} xAy$. Since x is known (because Bob has knowledge of Alice's mixed strategy), $f(y) = xAy$ is a function of y, and thus the problem reduces to the following linear program:

$$\max f(y) \qquad \text{s.t.} \sum y_j = 1,\ y_j \geq 0\ \forall y_j$$

which can be solved by Bob to determine his pure response (i.e. by setting the y_j that corresponds to the maximum element of xA in their dot product to 1, which will maximize xAy).

(b) We have just shown that if Bob knows about Alice's strategy, then he will choose the maximum element of xA to be his pure strategy. Knowing this, Alice's optimal strategy becomes

$$\min_{\sum x_i=1} \max_{\sum y_j=1} xAy = \min_{\sum x_i=1} \max\{xA_1, \ldots, xA_n\}$$

where A_i is the i-th column of A. We can rewrite this as the following linear program:

$$\begin{aligned} z &= \min c \\ \text{subject to} & \\ xA_i &\leq c \qquad \text{for all } i \\ \sum_{i=1}^{n} x_i &= 1 \\ x_i &\geq 0 \qquad \text{for all } i \end{aligned}$$

Conversely, if Bob's strategy is known, then Alice will choose a pure strategy corresponding to the minimum element of Ay, i.e. $\min\{A_1y, \ldots, A_ny\}$ where A_i is the i-th row of A. Thus, Bob's optimal strategy becomes

$$\max_{\sum y_j=1} \min_{\sum x_i=1} xAy = \max_{\sum y_j=1} \min\{A_1y, \ldots, A_ny\}$$

which can be rewritten in the form of a linear program as

$$\begin{aligned} w &= \max d \\ \text{subject to} & \\ A_iy &\geq d \qquad \text{for all } i \\ \sum_{i=1}^{n} y_i &= 1 \\ y_i &\geq 0 \qquad \text{for all } i \end{aligned}$$

(c) Previously, we showed that since Bob always plays optimally against Alice, it is in her best interest to minimize her potential losses. That is, once Alice realizes that Bob's strategy always results in potential winnings (for Bob) of $\max\{xA_1, \ldots, xA_n\}$, Alice should try to minimize this quantity c, and the linear program determines this by finding the optimal c such that each $xA_i \leq c$.

Likewise, once Bob realizes that Alice's strategy always results in a minimum net win of $\min\{A_1y, \ldots, A_ny\}$, he will want to maximize this quantity d while ensuring that $d \leq A_iy$ for all A_iy, which the linear program solves.

(d) From part (d), we have that Alice's linear program is

$$\max c \qquad \text{s.t. } xA_i \leq c \ \forall i, \ \sum_{i=1}^{n} x_i = 1, \ x \geq 0 \ \forall i$$

The dual of this is

$$\min d \qquad \text{s.t. } A_i y \geq d \ \forall i, \sum_{i=1}^{n} y_i = 1, \ y_i \geq 0 \ \forall i$$

which is exactly the linear program for Bob's optimal strategy. By strong duality, then, the two optimum outcomes for Alice and Bob are equal.