Name: Yan, Zi
Course: CIS 502
Assignment: HW7

Question 1

First of all, in order to ensure that we have enough sequence of x and sequence of y to choose from, let $xs = x^n$ and $ys = y^n$, such that both xs and ys will be no shorter than s.

Let $m(i, j)$ denote the feasibility of getting the first $i + j$ symbols after interleaving first i symbols of xs and first j symbols of ys. We know $m(0, 0) = 1$, and we will have the formula,

$$m(i,j) = \begin{cases} 1 & \text{if } (m(i-1,j) = 1 \text{ and } xs[i] == s[i+j]) \text{ or } (m(i,j-1) = 1 \text{ and } ys[j] == s[i+j]) \\ 0 & \text{otherwise} \end{cases}.$$

We will have to search all the combination of i ,j and compute the corresponding $m(i, j)$, such that for $k \in [1..n]$,$i + j = k$. At last, if there is at least one $m(i', j') = 1$, where $i' + j' = n$, we can say s is an interleaving of x and y.

Because the length of s is n, the maximum value of i or j should be n. Therefore, we need to compute a $n \times n$ matrix of m. The runtime is $O(n^2)$.

Question 2

Observation 1: we have to order some gas on day 1. Because the tank is empty at the end of day 0.

Observation 2: we do not need to consider the oil cost. Because for each possible scheme, the total cost of oil is always $\sum_{i=1}^{n} g_i$.

Observation 3: the amount of oil we order should be $\sum_{i=a}^{b} g_i$, where $a \leq b$, so that we can save the most, and it should be no greater than L.

Proof. Suppose not. If we order A gallon oil on day a, we know $A > g_a$ and $A = \sum_{i=a}^{b} g_i + \Delta$, where $a \leq b$. So we have to order again on day $a + b + 1$. But we can actually save $c(a + b)\Delta$ by only ordering $\sum_{i=a}^{b} g_i$ on day a. It contradicts our goal that we want to save as much as we can.

Because the tank size is L, the total amount of oil ordered should be no greater than L. $\square$

Observation 4: If we order oil on day a and use it up on day b, the total cost of storage is $c\sum_{i=a}^{b}(i-a)g_i$. Because for day a there is no storing fee for g_a, for day $a+1$, the storage cost is $c(a+1-a)g_{a+1}$. For day i, the storage cost is $c(i-a)g_i$. Therefore, the total cost is the sum of the storage cost for each day, namely $c\sum_{i=a}^{b}(i-a)g_i$.

For this problem, we have to consider it backwards from day n, because if we consider it forwards, there is no constraint on how many gallon oil we need to order. Let OPT(i) denote from day i to day n, the cost is lowest. Then we have $\text{OPT}(n) = 0$. $\text{OPT}(i) = P + \min\{\sum_{k=i}^{j}(k-1)g_k + \text{OPT}(j)\}$, where $j \geq i$ and j should ensure $\sum_{k=i}^{j} g_k \leq L$. And we need to know OPT(1).

Because the calculation of a sum is $O(n)$, and there are n OPTs, the total runtime is $O(n^2)$.

Question 3

a) For a bipartite graph $G = (V, E)$, each person $p_i \in V$, each night $d_i \in V$, and there is a edge $(p_i, d_j) \in E$ if $d_j \notin S_i$.

If G has a perfect matching, each person p_i will be paired with exact one night d_j, and no person is left alone and no night is left alone. Therefore, for each matching (p_i, d_j), the person p_i is able to cook on night d_j, and this provides a feasible schedule.

If a feasible schedule exists, for a person p_i and a night d_j, there will be a pair (p_i, d_j) as the assignment making p_i cook on night d_j, and each person is assigned to a different night. Therefore, in G, there is always an edge between a p_i and a d_j and each p_i is connected to a different d_j, forming a perfect matching.

b) If either p_i or p_j is able to cook on night d_l, we are done, because we only need to assign the person who is able to cook on night d_l and keep the other one to cook on night d_k. Otherwise, we use the algorithm described below.

We build a graph G described above, and a graph A representing the schedule of Alanis. We remove (p_i, d_k) from A to get A'. We also get the residual graph G_f of A' with respect to G. If we can find a path from p_j to d_l in G_f, we can augment A' with that path, and the result is a perfect matching, namely a feasible schedule, otherwise there will be no perfect matching, and no feasible schedule, either.

We need $O(n^2)$ to build the graph G, A', and G_f, we need another $O(n^2)$ to find the augmentation path. So the total runtime is $O(n^2)$.

Question 4

We build a graph $G = (V, E)$, where there are node s as a source, node t as a sink, nodes x_i representing advertiser i, and nodes u_j representing user j. 1) And there is an edge between s and x_i with lower bound r_i, where $i \in [1..m]$. 2)There is an edge between x_i and u_j with capacity 1 if $X_i \cap U_j \neq \emptyset$, where $i \in [1..m], j \in [1..n]$. 3) And there is an edge between each u_j and t with capacity 1. 4) Only s has a demand of $-\sum\limits_{i=1}^{m} r_i$ and t has a demand of $\sum\limits_{i=1}^{m} r_i$.

Then, if we can find a feasible circulation in G, we can determine that it is possible to assign each user to an ad conforming to the contract, because in the circulation, a pair (x_i, u_j) denotes that an ad i will be shown to user j, and the lower bound r_i of edge (s, x_i) guarantees that the ad i will be shown to users at least r_i times. And each edge u_j, t has capacity 1, so each user is shown at most one ad.

In the other way around, if there is a feasible assignment between ads and users, we can construct a feasible circulation as follows. We put the edge (x_i, u_j) in the circulation if ad x_i is shown to user u_j. Because each ad x_i is shown to at least r_i users, each edge (s, x_i) has at least a flow of r_i. And each user is shown at most one ad, so edge (u_j, t) will have at most a flow of 1.

Consequently, there is a feasible circulation in the graph G if and only if it is possible to assign each ad to some users conforming to the contract.

It takes us $O(nm)$ to build the graph G. $O(nm)$ is needed to find a feasible circulation, if possible. So total runtime is $O(nm)$