Name: _____ **Entry number**: _____

COL351 Quiz 4 Duration: 1 hour

Read the following instructions before you begin writing.

1. Keep a pen, your identity card, and optionally a water bottle with you. Keep everything else away from you, at the place specified by the invigilators.
2. Do not detach sheets. Write your entry number and name on every page. (We will detach sheets prior to grading.)
3. Answer only in the designated space. Think before you use this space. No additional space will be provided for writing answers. Blank pages will be provided for rough work on demand, but they cannot be used for writing answers.
4. No clarifications will be given during the exams. If something is unclear or ambiguous, make reasonable assumptions and state them clearly. The instructor reserves the right to decide whether your assumptions were indeed reasonable.
5. You may use any result discussed in class or tutorials without proving it. Similarly, you can use any algorithm discussed in class or tutorials as a "black-box" i.e., without reproducing any details of how it works.

---

1. (8 points) A *cycle cover* of a directed graph $G = (V, E)$ is a collection of directed cycles in G such that every vertex of G belongs to exactly one cycle. Equivalently, a cycle cover is a bijection $\sigma : V \longrightarrow V$ such that for all $v \in V$, $(v, \sigma(v)) \in E$. Design a polynomial time algorithm that, given a directed graph G, returns a cycle cover of G if one exists; otherwise returns "NO". Write a short proof of correctness and polynomial running time.

Construct a bipartite graph $G' = (L, R, E')$ as follows.

1. For each $v \in V$, include vertices v_l in L and v_r in R.
2. For each $(u,v) \in E$, include an edge $\{u_l, v_r\}$ in E'.

If $C \subseteq E$ is a cycle cover of G, then $\{\{u_l, v_r\} \mid (u,v) \in E\}$ is a perfect matching of G'.

If $M \subseteq E'$ is a perfect matching of G', then $\{(u,v) \mid \{u_l, v_r\} \in M\}$ is a cycle cover of G.

Algorithm: Construct G' and test whether it has a perfect matching $\longrightarrow$ poly time because we saw how to find a max-cardinality matching in poly time, in class.

Name: _____ Entry number: _____

COL351 Quiz 4 Duration: 1 hour

2. (12 points) The famous daily soap, "Kyunki Saas Bhi Kabhi Bahu Thi", is all set to launch its new season, speculated to run for the next ten years (at least). The television channel is anticipating an enormous viewership and a huge number of advertisers. They have, therefore, approached a super-smart person – you – to design an algorithm that assigns advertisement slots in a commercial break to advertisers with the goal of maximizing the revenue paid by the advertisers. You get a cut in this revenue, of course.

   More formally, the commercial break in an episode consists of m 30-sec advertising slots labelled $1, \ldots, m$. Each advertiser j (where $j \in \{1, \ldots, n\}$, $n \geq m$) specifies a subset $S_j \subseteq \{1, \ldots, m\}$ of slots where their 30-sec advertisement can be placed, and the amount $v_j > 0$ they propose to pay if their advertisement is shown in one of the slots in S_j. Repeating any advertisement is annoying to the viewers, and must be avoided. Assume $\bigcup_{j=1}^{n} S_j = \{1, \ldots, m\}$, which means for every slot there exists some advertiser interested in it.

   Design a **strongly polynomial time** algorithm to compute a revenue-maximizing assignment $a_1, \ldots, a_m$, where $a_i \in \{1, \ldots, n\}$ is the identity of the advertiser who gets the i'th slot (a_i's must be all distinct; the revenue earned by this assignment is $\sum_{j \in \{a_1, \ldots, a_m\}} v_j$). Clearly show your calculation of running time, and write a short proof of correctness.

Construct bipartite graph $G=(L, R, E)$, where $L=\{1 \ldots n\}$, $R=\{1 \ldots m\}$, $E = \{(j,i) \mid i \in S_j\}$. We are looking for a max-weight matchable subset of L (where v_j's are the weights). We proved in the tutorial that the set of matchable subsets of one side of a bipartite graph is a matroid, and we also know that Kruskal's algorithm finds a max-weight independent set.

∴ Algorithm is as follows:

1. Sort advertisers in nondecreasing order of v_j's.
2. $L' \leftarrow \phi$
3. For each j in the sorted order
   If $L' \cup \{j\}$ is matchable
   $L' \leftarrow L' \cup \{j\}$.
4. Find a matching in G that matches L' and return it.

Running time: $O(n \log n)$ for sorting

$O(n)$ iterations of the loop.

To test whether $L \cup \{j\}$ is matchable: $O(n)$ augmentations, each taking time $O(|E|) = O(mn)$ $\therefore$ $O(mn^2)$ time.

To find a matching finally: $O(mn^2)$ time using the same argument as above.

$\therefore$ Running time $= O(n \log n) + O(n \cdot mn^2) + O(mn^2)$

$= O(mn^3)$ → strongly polynomial in input size