There are 5 questions on this test.

You have to return the test to me or Tu Nguyen by 4:30pm on Tuesday, March 10, 2020. No answers will be accepted after this deadline. You can choose to send your answers back electronically before that deadline or hand them back at the beginning of class on Tuesday, March 10.

Please hand in ONE submission - multiple submissions will not be tolerated.

You are not allowed to discuss any problem with another human being, except the instructor for the course.

You can use a computer only as a word processor; in particular, you cannot consult the internet in regards to this midterm. You CAN use any other resource like the textbook, your notes, books from the library.

You CAN cite any result we have mentioned in class or from the HWs without proof. If you cite a result (e.g., from a book) that was NOT mentioned in class, you should include a complete proof of this fact.

The level of rigor expected is the same as the HW solutions. Make sure you justify all your answers.

There will be partial credit; e.g., if you do not prove part (i) of some question, but use part (i)'s result to prove part (ii), you can get full credit for part (ii).

I attest that I have completed this exam without unauthorized assistance from any person, materials, or device.

Name ____________________________________________

\( \mathbb{N} = \{0, 1, 2, \ldots\} \) will denote the set of natural numbers.

1. \( \textbf{(20 pts)} \) We say that a set \( X \subseteq \mathbb{R}^n \) is bounded if and only if there exists a number \( M \geq 0 \) such that \( X \subseteq \{ x \in \mathbb{R}^n : -M \leq x_i \leq M \ \forall i = 1, \ldots, n \} \).

Let \( P = \{ x \in \mathbb{R}^n : Ax \leq b \} \) where \( A \) is an \( m \times n \) matrix and \( b \in \mathbb{R}^m \) and assume that \( P \) is nonempty. Define the set \( C := \{ r \in \mathbb{R}^n : Ar \leq 0 \} \). Show that \( P \) is bounded if and only if \( C = \{ 0 \} \).

[Hint: \( P \) is bounded if and only if the linear programs \( \max\{ x_i : Ax \leq b \} \) and \( \max\{ -x_i : Ax \leq b \} \) have finite values for all \( i = 1, \ldots, n \).]
2. **Game of Slither.** The game of *Slither* is played on a graph $G = (V,E)$ between two players. The players, called *First* and *Second*, play alternately, with First playing first. At each step the player whose turn it is chooses a previously unchosen edge. The only rule is that at every step the set of chosen edges forms a simple path (i.e., a path with no repeated vertices). The loser is the first player unable to make a legal play at his or her turn.

(i) **(10 pts)** Show that if $G$ has a perfect matching (a matching which leaves no vertex exposed), then First has a winning strategy.

(ii) **(10 pts)** Suppose First’s first move is an edge $uv$ such that $u$ is an inessential vertex of $G$. Show that Second can now force a win. [Recall that a vertex is inessential if there exist a maximum matching that leaves this vertex exposed]

3. **(5 pts)** Let $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ where $A$ is an $m \times n$ matrix and $b \in \mathbb{R}^m$. Let $c \in \mathbb{R}^n$ and suppose $\delta = \max\{c^T x : x \in P\}$. Define $F = \{x \in P : c^T x = \delta\}$, i.e., $F$ is the set of ALL optimal solutions to the linear program. Show that if $x \in F$ and $x_1, x_2 \in P$ such that $x = \frac{1}{2}(x_1 + x_2)$, then $x_1, x_2 \in F$.

4. **(20 pts)** We consider the problem of scheduling a set $J$ of jobs on $M \in \mathbb{N}$ uniform parallel machines. Each job $j \in J$ has a processing requirement $p_j \in \mathbb{N}$ which denotes the number of days needed to process the job on any one of the machines. Each job also comes with a release date $r_j \in \mathbb{N}$, representing the beginning of the day the job $j$ becomes available for processing, and a due date $d_j \geq r_j + p_j$, representing the beginning of the day by the end of which job $j$ must be completed ($d_j$ is also a natural number). We start our count of days from day 0. Any machine can process only one job in a single day, and any job can be processed by at most one machine on a given day. However, we allow preemptions, i.e., we can interrupt a job and process it on different machines on different days (recall that all machines are uniform, i.e., have the same processing speed). The scheduling problem is determine if there exists a feasible schedule of jobs assigned to machines on different days such that all jobs are processed in their respective windows.

Give an algorithm that decides in polynomial time whether a feasible schedule exists or not. The algorithm should be polynomial time in the data, that is polynomial in $|J|, M, \sum_{j \in J} (\log r_j + \log d_j + \log p_j)$.

[Hint: The following construction could be useful: Arrange the numbers $r_j, d_j$ in ascending order and create $2|J| - 1$ intervals of consecutive days defined by these numbers.]

5. **(20 pts)** Let $A$ be an $m \times n$ matrix such that every entry is either 0 or 1. Moreover, in any column the 1’s are consecutive, i.e., every column is of the following form:

$$[0,0,\ldots,0,1,1,\ldots,1,0,0,\ldots,0]^T.$$

Show that $A$ is totally unimodular.
Hint: Use row operations to reduce to a matrix which you know is totally unimodular from the results done in class.