Name: ______________________________

Entry number: ______________________________

- Always try to give algorithm with best possible running time. The points that you obtain will depend on the running time of your algorithm. For example, a student who gives an $O(n)$ algorithm will receive more points than a student who gives an $O(n^2)$ algorithm.
- You are required to give proofs of correctness whenever needed. For example, if you give a greedy algorithm or an algorithm using network flow for some problem, then you should also give a proof why this algorithm outputs optimal solution.
- You may use any of the following known NP-complete problems to show that a given problem is NP-complete: 3-SAT, INDEPENDENT-SET, VERTEX-COVER, SUBSET-SUM, 3-COLORING, 3D-MATCHING, SET-COVER, CLIQUE, HAMILTONIAN-CYCLE, HAMILTONIAN-PATH.
- Some of the questions might be related. Make sure you read all questions.
- **Use of unfair means will be severely penalized.**

There are 5 questions for a total of 30 points.

---

1. (3 points) Given integers a and b that can be represented in m and n bits respectively, design an algorithm to compute a^b. Discuss running time in terms of m and n. You may assume that multiplying two numbers is a unit time operation irrespective of the size of the numbers.

Solution: Here is the algorithm that computes a^b time $O(n)$ time:

```
Power($a, b$)
  If ($b == 0$) reutrn(1)
  $y \leftarrow Power(a, \lfloor b/2 \rfloor)$
  If ($b$ is even) return($y \cdot y$)
  else return($y \cdot y \cdot a$)
```

Let $T(m, n)$ denote the running time of the algorithm. Then we have $T(m, n) = T(m, n-1) + O(1)$ and $T(m, 1) = O(1)$, which gives $T(m, n) = O(n)$.

2. (3 points) Consider the following problem:

LONG-PATH: Given a weighted, directed graph $G = (V, E)$, two vertices $s, t \in V$ and a number W, determine if there is a *simple path* between s and t such that the sum of weights of edges in this path is $\geq W$.

Recall that a simple path is a path that does not have any vertices repeated. Show that LONG-PATH is NP-Complete.

Solution: A simple path of of weight at least W acts as a short certificate and the certifier just checks this path. So, LONG-PATH $\in NP$. To show that LONG-PATH is NP-complete, we will show that HAMILTONIAN-PATH $\leq_p$ LONG-PATH. Here is an algorithm that solves the HAMILTONIAN-PATH problem given using an algorithm, *Long-Path* for solving the LONG-PATH problem:

```
Check-Ham-Path(G)
1.  Construct $G'$ from $G$ by giving weight of 1 to all edges of $G$.
2.  For all pairs of vertices $s, t \in V$
3.      If (Long-Path($G', s, t, n-1$) == 1)then return(1)
4.  return(0)
```

Claim 1. If G has a hamiltonian path, then *Check-Ham-Path* returns 1.
Proof. Since G has a hamiltonian path there exists vertices s, t and a path from s to t that visits all vertice exactly once. This path has length $n-1$ in G'. So the algorithm returns 1 in one of the iterations.

Claim 2. If *Check-Ham-Path* returns 1, then G has a hamiltonian path.
Proof. Since the algorithm returns 1, there is a pair of vertices s, t such that there is a simple path of length $n-1$ in G'. This means that this path visits all vertices exactly once. This means that this path is a hamiltonian path.

3. Given an undirected, unweighted graph $G = (V, E)$, consider the following four Linear Programs:

LP1	**LP2**	**LP3**	**LP4**
Maximize $\sum_{e \in E} y_e$	*Maximize* $\sum_{e \in E} y_e$	*Minimize* $\sum_{v \in V} x_v$	*Minimize* $\sum_{v \in V} x_v$
Subject to:	*Subject to:*	*Subject to:*	*Subject to:*
For every $u \in V$,	For every $u \in V$,	For every $(u, v) \in E$,	For every $(u, v) \in E$,
$\sum_{e:e \text{ contains } u} y_e \leq 1$,	$\sum_{e:e \text{ contains } u} y_e \leq 1$,	$x_u + x_v \geq 1$,	$x_u + x_v \geq 1$,
and	and	and	and
$\forall e \in E, y_e \in \{0, 1\}$	$\forall e \in E, 0 \leq y_e \leq 1$	$\forall v \in V, 0 \leq x_v \leq 1$	$\forall v \in V, x_v \in \{0, 1\}$

Let F_1, F_2, F_3, F_4 denote the value of the optimal solutions of **LP1**, **LP2**, **LP3**, and **LP4** respectively. Answer the following:

(a) (2 points) Show that F_1 is the cardinality of maximum matching in G. A matching in a graph is defined to be a subset $S \subseteq E$ of edges such that each vertex is present as an endpoint in at most one of the edges in S. A maximum matching is a matching with maximum cardinality.

Solution: We first show that if there is a matching of size k in G, then there is a feasible solution of **LP1** with objective value k. Let M be a matching of size k. Consider the following solution to **LP1**: $y_e = 1$ iff $e \in M$. Note that this assignment to variables satisfies all the constraints, since for any vertex u if $y_e = 1$ for $e = (u, v)$, then $y_{e'}$ for all edges $e' \neq e$ incident on u is 0 since e is part of a matching.

Next, we show that if there a feasible solution to **LP1** with objective value k, then there is a matching in G of size k. Indeed, consider any feasible solution $\{y'_1, ..., y'_{|E|}\}$ to **LP1** with objective value k. Let M' contain edge e iff $y'_e = 1$. Note that M' is a matching in G (otherwise some constraint fails), and since $\sum_{e \in E} y'_e = k$, we have $|M'| = k$.

From the above, we know that there is a matching in G of size F_1. A matching of larger size than F_1 cannot exist since it will contradict with the fact that F_1 is the optimal solution to **LP1**.

(b) (3 points) We know that F_4 is the cardinality of minimum vertex cover of G. Argue that $F_1 \leq F_4$.

Solution: Let M be any matching in G and let S be any vertex cover of G.
Claim. For every $(u, v) \in M$, at least one of u and v is in S.
Proof. This is true since otherwise the edge (u, v) is not covered by S.
Note that since each vertex occurs in at most one edge in M, the above claim implies that $|M| \leq |S|$. Let M_{max} denote a maximum matching and let S_{min} be a minimum vertex cover of G. Then from above claim, we have that $F_1 = |M_{max}| \leq |S_{min}| = F_4$.

(c) (4 points) Show that $F_1 \leq F_3$ and that $F_2 \leq F_4$.

Solution: This follows from weak LP duality that we discussed in a tutorial. We will show that $F_1 \leq F_2$, $F_3 \leq F_4$, and $F_2 \leq F_3$. The first two inequalities follow from the fact that **LP2** and **LP3** are relaxed versions of **LP1** and **LP4** respectively (since if integer constraints are relaxed, the optimal value can only get better). $F_2 \leq F_3$ follows from weak duality. Let $Y' = (y'_1, ..., y'_{|E|})$ be any feasible solution of **LP2** with objective value F'_2 and $X' = (x'_1, ..., x'_{|V|})$ be any feasible solution to **LP3** with objective value F'_3. Consider any edge $e = (u, v) \in E$ and consider the inequality w.r.t. this edge in **LP3**, i.e., $x_u + x_v \geq 1$. We know that $x'_u + x'_v \geq 1$ since X' is a feasible solution to **LP3**. We multiply both sides of the inequality by y'_e to get the following: $y'_e \cdot x'_u + y'_e \cdot x'_v \geq y'_e$. Note that the inequality does not change because $y'_e \geq 0$. We do this for all inequalities in **LP3** and then combine them to get:

$$\sum_{u \in V} x'_u \cdot \left(\sum_{e:e \text{ } contains \text{ } u} y'_e \right) \geq \sum_{e \in E} y'_e$$

Since Y' is a feasible solution to **LP2**, we have that for any $u \in V$, $\sum_{e:e \text{ } contains \text{ } u} y'_e \leq 1$. Using this, we can replace the quantities within the brackets above with 1 and obtain:

$$\sum_{u \in V} x'_u \geq \sum_{e \in E} y'_e$$

This gives us $F'_2 \leq F'_3$. Note that this is true for any pair of feasible solutions and so it will be true for the optimal solutions which implies that $F_2 \leq F_3$.

(d) (2 points) Note that **LP1** is an instance of a problem that is NP hard (this was discussed in the lectures). On the other hand, there is an efficient algorithm (known as Edmond's algorithm) that computes the maximum matching in any given graph in time $O(|E|\sqrt{|V|})$. Why doesn't this imply that **P** = **NP**? Give reasons.

Solution: The NP-hardness is in the *worst-case* sense. There might be a large number of instances which might be easy to solve. The ILP instances arising from the matching problem are a small subset of instances of the problem that are easy to solve. This does not imply that ILP problem is easy.

(e) (4 points) Consider the following greedy algorithm for computing a minimum vertex cover of G: Iteratively pick and store an edge (u, v) from G and remove edges adjacent to u and v. Note that this gives a *maximal* matching in G. Let S be all the vertices occurring (as endpoints of some edge) in this matching. Argue that S is a vertex cover and that $|S| \leq 2 \cdot F_4$. This means that the greedy algorithm is a 2-approximation algorithm.

Solution: The algorithm gives a *maximal* matching M in G. Let S be the vertices occurring in M.

Claim. S is a vertex cover.
Proof. For the sake of contradiction there is an edge (u, v) that is not covered. This means that $u \notin S$ and $v \notin S$. This implies that neither u not v occur in any any edge in M. But then this means that M is not a maximal matching since we can add the edge (u, v) to M to get a matching of larger cardinality.

Let OPT denote any minimum vertex cover of G. We know that for every edge $(u, v) \in M$, either $u \in OPT$ or $v \in OPT$ (or both), otherwise the edge (u, v) is not covered. This means that $|M| \leq |OPT| = F_4$. Now, S is of size $2|M|$. So, we get that $|S| = 2 \cdot |M| \leq 2 \cdot F_4$.

4. (5 points) There is an $n \times n$ grid in which some cells are empty and some are filled. The empty/filled cells are given by an $n \times n$, 0/1 matrix F. Cell (i, j) is empty iff $F[i, j] = 0$. You have unbounded supply of 2×1 tiles (called dominoes). Each domino could be placed on the empty cells of the grid in horizontal and vertical manner (note that you need two consecutive empty cells on the grid for doing this). The problem is to determine if the grid can be covered by placing these 2×1 dominos such that each empty cell is covered by exactly one domino. Do you think this problem is in **P**? You may assume that **P** $\neq$ **NP** for this problem. Give reasons.

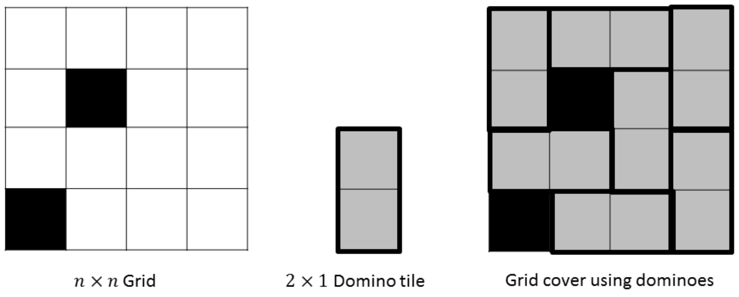

Figure 1: Example of a 4×4 grid that can be covered using 2×1 domino tiles.

Solution: You can reduce this problem to the problem of finding a maximum matching in a general graph. You know from problem 3(d) that there is an algorithm (Edmond's algorithm) that solves the problem in polynomial time. Hence, this problem is in **P**. Here is the reduction to graph matching. Given F, we create a graph G with n^2 vertices, one vertex corresponding to each cell in the grid. Every vertex corresponding to an empty cell has edges to vertices corresponding to its neighboring empty cells (neighbor in the grid). The vertices corresponding to full cells do not have any incident edges. Let m be the number of empty cells in the grid. Now, we show the following:

Claim. There exists a domino tiling of the grid iff the maximum matching in G is of size $m/2$.
Proof. If there is a domino tiling, then we consider the matching where the pair of adjacent vertices on which a domino tile is placed is part of the matching. Since the grid can be tiled, $m/2$ tiles can be placed which means that there are $m/2$ edges in the matching. Also note that since the graph has m connected vertices, it cannot have matching of size $> m/2$.

In the other direction, if the maximum matching is of size $m/2$, then we just place tiles on adjacent cells of the grid indicated by the edges in the matching. This way we cover the entire grid.

5. (4 points) There are n men with heights $m_1, ..., m_n$ and n women with heights $w_1, ..., w_n$. You have to match men to women for a dance such that the average difference in height of each pair, is minimized. Design an algorithm to solve this problem. Discuss running time.

Solution: We will show that the following greedy algorithm minimizes the average difference in height:

> Separately, sort the men and women according to their heights (say increasing order). Match the first man in the sequence to the first woman, second man in the sequence to the second woman, and so on.

The running time of this algorithm is clearly $O(n \log n)$.

We will now prove optimality. For analysis assume that $m_1 \leq m_2 \leq ... \leq m_n$ and $w_1 \leq w_2 \leq ... \leq w_n$. Consider any optimal solution σ. This solution is just a permutation of $1...n$. This means that as per the optimal solution, M_1 is matched with $W_{\sigma(1)}$, M_2 is matched with $W_{\sigma(2)}$, and so on. Let i be the smallest index such that $\sigma(i) \neq i$. Let j be the index such that $\sigma(j) = i$. Clearly, $j > i$. Consider another solution σ_1 that is the same as σ, except $\sigma'(i) = i$ and $\sigma'(j) = \sigma(i)$. We will show that the average height difference as per σ' is less than equal to the average height difference as per σ. Let the average height difference as per σ and σ' be denoted by $H(\sigma)$ and $H(\sigma')$ respectively. We have:

$$H(\sigma) - H(\sigma') = \frac{1}{n} \cdot (|m_i - w_{\sigma(i)}| + |m_j - w_i| - |m_i - w_i| - |m_j - w_{\sigma(i)}|)$$

We consider the following 6 cases:

1. $w_i \leq w_{\sigma(i)} \leq m_i \leq m_j$: In this case, $H(\sigma) - H(\sigma') = 0$.
2. $w_i \leq m_i \leq w_{\sigma(i)} \leq m_j$: In this case, $H(\sigma) - H(\sigma') = (1/n) \cdot 2 \cdot (w_{\sigma(i)} - m_i) \geq 0$.
3. $m_i \leq w_i \leq w_{\sigma(i)} \leq m_j$: In this case, $H(\sigma) - H(\sigma') = (1/n) \cdot 2 \cdot (w_{\sigma(i)} - w_i) \geq 0$.
4. $w_i \leq m_i \leq m_j \leq w_{\sigma(i)}$: In this case, $H(\sigma) - H(\sigma') = (1/n) \cdot 2 \cdot (m_j - m_i) \geq 0$.
5. $m_i \leq w_i \leq m_j \leq w_{\sigma(i)}$: In this case, $H(\sigma) - H(\sigma') = (1/n) \cdot 2 \cdot (m_j - w_i) \geq 0$.
6. $m_i \leq m_j \leq w_i \leq w_{\sigma(i)}$: In this case, $H(\sigma) - H(\sigma') = 0$.

So, in all the cases $H(\sigma') \leq H(\sigma)$. We continue with this exchange operation and at the end we obtain our greedy solution.