Name: ___________________________    ID: _______________________

Instructions:

• Answer the questions directly on the exam pages.

• Show all your work for each question. Providing more detail including comments and explanations can help with assignment of partial credit.

• If the answer to a question is a number, unless the problem says otherwise, you may give your answer using arithmetic operations, such as addition, multiplication, “choose” notation and factorials (e.g., “9 \times 35! + 2” or “0.5 \times 0.3/(0.2 \times 0.5 + 0.9 \times 0.1)” is fine).

• If you need extra space, use the back of a page.

• No books, notes, calculators or other electronic devices are allowed. Any cheating will result in a grade of 0.

• If you have questions during the exam, raise your hand.

<table>
<thead>
<tr>
<th>Question</th>
<th>Value</th>
<th>Points Earned</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>
Question 1. (10 points) Indicate whether each of the following statements is TRUE or FALSE. No justification required.

1.1 (2 points): \[\sum_{i=1}^{n} \frac{1}{i^2} = \Theta(n^4)\].

1.2 (2 points): A graph with \(n\) vertices and \(n-1\) edges is either disconnected or a tree.

1.3 (2 points): For every \(n\) there exists a directed graph on \(n\) vertices with \(\Omega(n^2)\) edges that has a topological ordering.

1.4 (2 points): In a connected weighted graph, the edge with maximum weight is never in the minimum spanning tree.

1.5 (2 points): The recurrence \(T(n) = 4T(n/2) + O(n)\) solves to \(T(n) = \Theta(n^3)\).
Question 2. (10 points)

2.1 (5 points): Recall the scheduling problem where we have several tasks with lengths $t(i)$ and deadlines $d(i)$ and we want to order the tasks to minimize lateness where, if task $i$ is completed at time $f(i)$, then lateness is defined as $L = \max_i \max(0, f(i) - d(i))$. Prove that ordering the intervals by their slack time, i.e., $d(i) - t(i)$ fails to find an optimal solution.

2.2 (5 points): On a stable matching instance, prove that if we run the Gale-Shapley algorithm twice, once with schools proposing and once with students proposing and we obtain the same matching, then the instance has a unique stable solution.
Question 3. (10 points) Alice is planning her course schedule for her time at UMass. There are \( n \) courses she must take and each course \( c_i \) can have pre-requisites \( P_i \), which is a possibly empty set of courses. However, the department allows students to take a course and its prerequisites in the same semester. In other words, a course \( c_i \) can be taken in semester \( t \) if for all \( c_j \in P_i \) the semester in which Alice takes \( c_j \) is at most \( t \).

On the other hand, Alice can take at most 3 courses in a semester.

1. Prove that if the pre-requisite graph has a cycle of length 4, then there is no way for Alice to find a schedule satisfying all the pre-requisites.

2. Prove that if every course is involved in at most one cycle of length at most 3, then a valid schedule must exist.
Question 4. (10 points) Given two lists, $L_1$ of length $n$ and $L_2$ of length $m$. We say that $L_2$ is a subsequence of $L_1$ if we can remove elements from $L_1$ to produce $L_2$. This means that there exists indices $i_1 < \ldots < i_m$ such that $L_1[i_j] = L_2[j]$ for each $j$. Design an algorithm that detects if $L_2$ is a subsequence of $L_1$ and outputs the indices $i_1, \ldots, i_m$ if $L_2$ is a subsequence of $L_1$. 
Question 5. (10 points) Suppose we have a complete $k$-ary tree with $n$ leaves (suppose $n = k^d$ for some integer $d$). Each leaf $v$ is associated with a weight $w(v)$. The weight of an internal node is defined to be the sum of the weights of all leaves that are descendants of this node. So the weight of the root $r$ is $w(r) = \sum_{\text{leaves } v} w(v)$. Design and analyze an algorithm to compute the weight of every internal node.