COMP 5970/6970 HW 1: 5 questions 5 points 5% Credit

**Due before 11:59 PM Tuesday January 29**

Instructions:

1. This is an individual assignment. You should do your own work. Any evidence of copying will result in a zero grade and additional penalties/actions.
2. Enter your answers in this Word file. Submissions must be uploaded **as a single file** (Word or PDF preferred, but other formats acceptable as long as your work is LEGIBLE) to Canvas before the due date and time. Don’t turn in photos of illegible sheets. **If an answer is unreadable, it will earn zero points.** Cleanly handwritten submissions (print out this assignment and write answers in the space provided, with additional sheets used if needed) scanned in as PDF and uploaded to Canvas are acceptable.
3. **Submissions by email or late submissions (even by minutes) will receive a zero grade.** No makeup will be offered unless prior permission to skip the assignment has been granted, or there is a valid and verifiable excuse.

**Multiple Choice Questions (5 points)**

*In the following questions, circle the correct choice. If more than one answer is correct, circle all that apply. In those cases, partial credit will be given to partially correct answers. No explanation needed. Incorrect answers or unanswered questions are worth zero points.*

1. “Any problem that can be solved with a greedy algorithm can also be solved with dynamic programming.” The statement is:

[a] True

[b] False

2. “Dynamic programming can be used to find an approximate solution to an optimization problem, but cannot be used to find a solution that is guaranteed to be optimal.” The statement is:

[a] True

[b] False

3. In dynamic programming algorithm, we drive a recurrence relation for the solution to one subproblem in terms of solution to others and reuse the solutions to smaller subproblems in order to solve a larger problem. Suppose the recurrence relation for a dynamic programming algorithm is of the form:



The number of subproblems is:

[a] 3n

[b] 2n

[c] n2

[d] none of the above

4. In bottom-up dynamic programming algorithm, we need a traversal order such that all needed subproblems are solved before solving the original problem. Suppose the recurrence relation for a dynamic programming algorithm is of the form:



A valid traversal order is:

[a] Solve A(i, j) for (i from 0 to n: for (j from 0 to n))

[b] Solve A(i, j) for (i from 0 to n-2: for (j from i+2 to n))

[c] Solve A(i, j) for (i from 0 to n-2: for (j from i to n))

[d] none of the above

5. Consider two vertices, s and t, in some directed acyclic graph G = (V, E). Let’s assume that a dynamic programming algorithm is developed to determine the number of paths in G from s to t. The running time of the algorithm is:

[a] O (VE)

[b] O (V+E)

[c] O (VlgV+E)

[d] none of the above