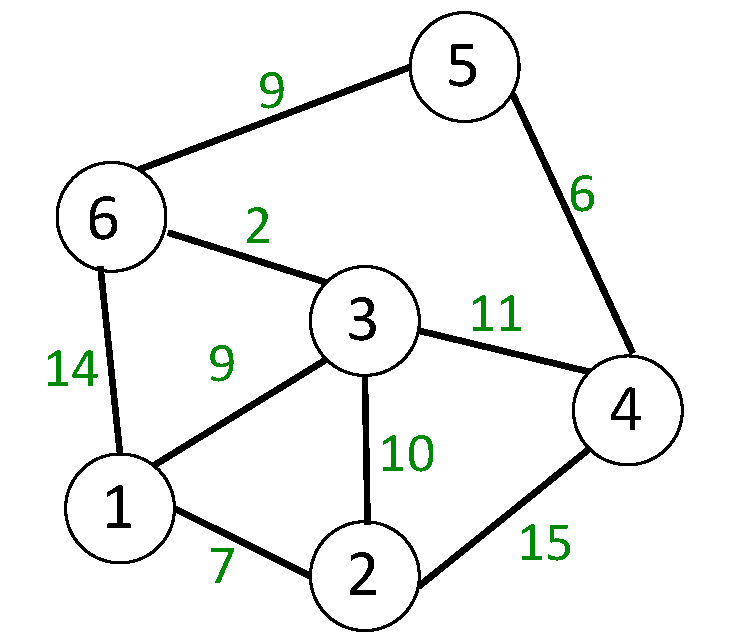
**Algorithms (CSCI 323 & 700)**

**Spring 2016 - Homework #8**

**(Due at the beginning of class on 4/20/2016)\***

1. Show how Kruskal’s Algorithm would build a Minimum Spanning Tree for the graph below. Apply an appropriate sort for the edge-weights, then perform the six MAKE-SET operations, finally do the necessary number of FIND-UNION operations. Compare the MST obtained with the one obtained in last week’s homework using Prim’s Algorithm.



Use the “Dynamic Programming” (DP) approach to solve each of the following three classic DP problems. It is expected that you will use the internet and other resources to find ideas toward solutions. However, you should also understand the answers so that you can reutilize them as needed when such resources are not available. (Hint: you may find the first similar to the Maximum Subarray Problem, the second similar to the Change-Making Problem, and the third similar to Floyd’s APSP Algorithm, all discussed in class.)

1. Consider a rectangular m x n grid which whose cells each contains either a 0 or 1. Design an O(m\*n) algorithm that will find the largest square k x k sub-grid whose cells contain *only* 1s. (Hint: use an auxiliary array that keeps track of the largest all-1s square whose upper-left corner begins at that position.)
2. Consider a set of n items having respective weights (w1, w2, …, wn) and values (v1, v2, …, vn). Design an algorithm that finds the subset of these items that has maximal value, but without exceeding some integer total weight constraint W. The algorithm should run in O(n\*W), where again, W is the integer upper bound on the total weight (capacity) of the container. Note that in this “0-1” version of the problem, you can either include an item or exclude it; you cannot include multiple copies of a given item. (Hint: as you gradually allow yourself to use one more item, that will also reduce the remaining allowed weight remaining in the container.)
3. A polygon is a geometric figure in the plane with at least three straight sides and angles between them. (With 3 sides, it is a triangle; with 4 sides, it is a quadrilateral; with 5 sides, it is a pentagon; with n sides, it is an n-gon.) We would like to “triangulate” the polygon, i.e. draw lines between vertices of the polygon so that the entire polygon is now divided up into triangles. Each of the triangles has a weight associated with it: if a triangle has vertices A, B, C, then the weight of triangle ABC is defined as d[A, B] + d[B, C] + d[C, A], where d is the geometric distance between the two points. Design an algorithm that finds the optimal triangulation, i.e. the division of the polygon into triangles so that the sum of the weights of all the triangles is the minimum possible.

\* Only if you will not be able to attend class on the due date, submit your solutions - *before 6:00 p.m. on the due date* - to the instructor at LT.CS320@yahoo.com