CSCI 411 - Advanced Algorithms and Complexity
Coding Assignment Replacement

March 2, 2023

The following coding problems may be used to improve scores for coding problems on assignments 1 to 5. Specifically, scores for the original problems will be considered along with scores for the problems described here. The highest five, totaling at most 115 points, will be used in your grade (you cannot get more than 115 total points). You may complete as few or as many of these problems as you like. Simple skeleton code for these problems will be available on Canvas.

C++ code should be submitted on both Canvas and turnin. This assignment is due before **May 7th at 11:59 pm**. Your primary focus over the course of the next several weeks should be the project. These problems should not take time away from your efforts on the project.

I highly encourage you to collaborate with one another on these problems. However, you must write up your own solutions **independently**. Feel free to communicate via Discord and to post questions on the appropriate forum on Canvas. Do not post solutions. Also, please include a list of the people you work with at the top of your submission.

Coding Problems

1. (20 pts) Implement the Floyd-Warshall algorithm using dynamic programming.
   - Submit a file called floyd-warshall.cpp
   - Input will come from cin
     - The first line will contain two integers, n and m, separated by a space.
     - n is a number of vertices and m is a number of edges.
     - The next m lines will contain three integers, u, v, and w, separated by spaces.
     - Each of these triples represents a directed edge (u, v) and its weight w.
   - Print output to cout
     - Your output should consist of n lines each containing n space separated integers.
     - The ith line represents distance from node i to all other nodes in order of node label.
       - The jth number on the ith line is the distance from node i to node j.
       - If no path exists from node i to node j, represent the distance with the string `"INF"`.
2. (30 pts) Implement a dynamic programming algorithm for finding a minimum vertex cover of a tree following appropriate pseudocode from problem (2b) in assignment 5.

- Submit a file called tree_vc.cpp
- Input will come from cin
  - The first line will contain two integers, n and m, separated by a space.
  - n is a number of vertices and m is a number of edges.
  - The next m lines will contain two integers, u and v, separated by a space.
  - Each of these pairs represents an undirected edge $\{u, v\}$.
- Print output to cout
  - Your output should consist of a single integer representing the size of the vertex cover discovered.

3. (20 pts) Let $T = (V, E)$ be a tree. The degree of a node $v \in V$, denoted $deg(v)$, is its number of neighbors. A *leaf* is any vertex with degree 1 while a *major vertex* has degree 3 or greater. $v \in V$ is an *exterior major vertex* if it has degree 3 or greater and it is closer to at least one leaf than any other major vertex. Put another way, $v \in V$ is an exterior major vertex if $deg(v) \geq 3$ and $\exists \ell \in V$ such that $deg(\ell) = 1$ and, $\forall u \in V$ such that $deg(u) \geq 3$, $d(v, \ell) \leq d(u, \ell)$.

   Write code to partition the leaves of a tree with respect to exterior major vertices. That is, for each exterior major vertex $v \in V$, find all leaves $\ell \in V$ such that $d(v, \ell) \leq d(u, \ell)$ for all other exterior major vertices $u \in V$.

   - Submit a file called leaf_partition.cpp
   - Input will come from cin
     - The first line will contain two integers, n and m, separated by a space.
     - n is a number of vertices and m is a number of edges.
     - The next m lines will contain two integers, u and v, separated by a space.
     - Each of these pairs represents an undirected edge $\{u, v\}$.
   - Print output to cout
     - For each exterior major vertex print a single line of space separated integers.
     - The first integer in each line represents the exterior major vertex itself.
     - The remaining integers represent the leaves associated with this exterior major vertex in sorted order.
     - Lines should appear in sorted order with respect to exterior major vertex labels.

4. (20 pts) Given a string S, generate a Huffman encoding (see Section 16.3 - Huffman codes (page 428-435) of *Introduction to Algorithms*).

   - Submit a file called huffman_encoding.cpp
   - Input will come from cin
     - There will be a single line containing a sequence of space separated strings.
   - Print output to cout
     - The number of bits (0s and 1s) required to represent the input string using a Huffman encoding.
     - Be sure to include whitespace characters (spaces) in the encoding.

5. Implement any single coding problem for which you did not originally include a submission. You may only provide a submission for one such problem.