CSCI 411 - Advanced Algorithms
Assignment 2

August 30, 2024

Solutions to the written portion of this assignment should be submitted via PDF to Canvas. Make sure to justify your answers. C++ code should be submitted on INGInious. Both parts of the assignment are due before **October 6th at 11:59 pm**. Discussions for problems 1 and 2 will be held in class the week before the due date.

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

Written Problems

1. **Group problem 2.1:** Let $G = (V, E)$ be a directed graph with weighted edges. We are interested in the set of vertices $I = \{v | v \in V, \exists u \in V \text{ s.t. } d(u, v) = -\infty\}$. That is, the set of vertices v such that there is at least one path of length $-\infty$ ending at v.
   (a) (4 pts) Describe an intuitive approach for generating the set I.
   (b) (8 pts) Write pseudocode for a function `findSetI(G)` which returns the set I for a given graph.
   (c) (3 pts) Analyze the asymptotic run time of your algorithm.
2. **Group problem 2.2:** Recall the recent exciting news in space exploration: An expedition to Mars has uncovered evidence of an ancient alien civilization! Among the discovered artifacts is what appears to be a partial dictionary. To facilitate a linguistic analysis of the Martian language, we found a way to determine one alphabetic ordering of the individual characters and to determine whether or not this ordering was unique. Now we need to count the total number of possible orderings.
   (a) (4 pts) Describe an intuitive solution to this problem.
   (b) (10 pts) Write pseudocode for a function `numOrders(G)` which returns the total number of possible alphabetic orders of the characters in the graph `G`. This graph is generated from the dictionary as discussed in class.

(c) (1 pt) Analyze the asymptotic runtime of your algorithm. Be aware that computer scientists believe that this is a computationally complex task in general. The asymptotic runtime might be quite high!

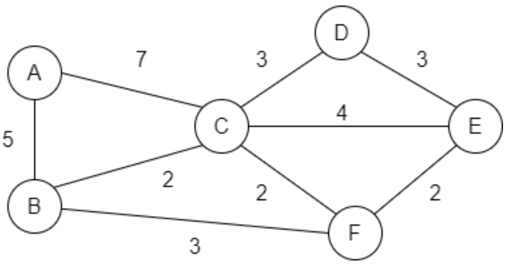
3. There are n outposts, labeled $0, \ldots, n-1$, on the surface of a distance planet. Outpost i is located at coordinates (x_i, y_i) and is equipped with a transmitter of strength s_i. This means that outpost i can send a message s_i miles in any direction. Transmitters can receive any message that reaches the location of the associated outpost. A large long-lasting electrical storm is preventing messages from being sent directly from an orbiting space station to these outposts. Instead, messages must be sent to the ground physically where they can then be communicated between the outposts. Physical travel from the space station to the planet's surface and back is costly. What is the maximum number of outposts that can be contacted with a single trip? Another question, which we will not explore here but which is worth thinking about, is what is the fewest number of outposts that need to be contacted in order for all outposts to receive their messages?

   (a) (10 pts) Describe an intuitive approach for determining this number.

   (b) (15 pts) Write pseudocode for a function `maxContacts(locs, S)` which takes as input a list of outpost locations and a list of transmitter strengths and returns the maximum number of outposts that can be contacted with a single trip to the planet's surface.

   (c) (5 pts) Analyze the asymptotic runtime of your algorithm.

4. For each of the following graphs, generate two minimum spanning trees, one using Kruskal's algorithm and one using Prim's algorithm starting at node A as described in class. List the edges of these trees in the order in which they were selected. If at any point there are multiple edges that may be added on the next step, use an asterisk to indicate that you made a choice. For example, one solution for this graph



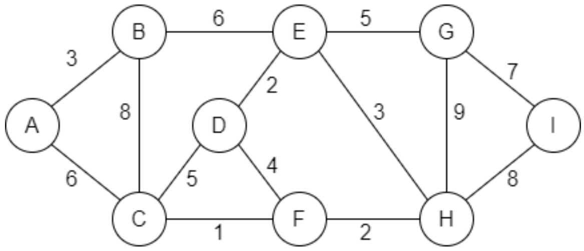
   would be:

   Kruskal's: *{B,C}, *{E,F}, {C,F}, *{D,E}, {A,B}
   Prim's: {A,B}, {B,C}, {C,F}, {E,F}, *{C,D}

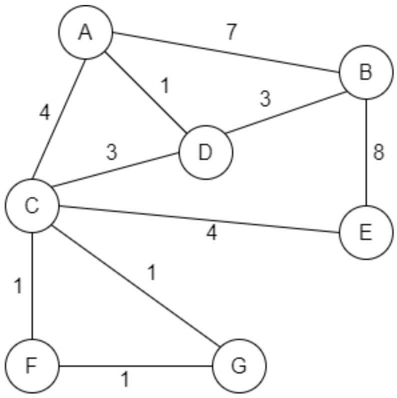
   Here is another possible solution:

   Kruskal's: *{E,F}, *{C,F}, {B,C}, *{C,D}, {A,B}
   Prim's: {A,B}, {B,C}, {C,F}, {E,F}, *{D,E}

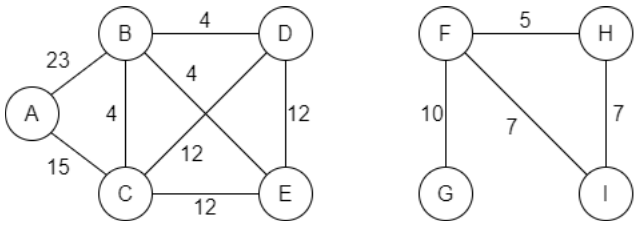
   (a) (10 pts)



(b) (10 pts)



(c) (10 pts)



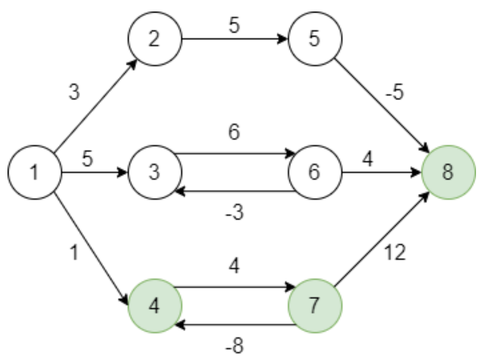
Coding Problems

1. (10 pts) Write a C++ implementation of the pseudocode you developed for problem (1b) and submit to INGInious as assignment_2_q_1.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
     - On one line print a list of the vertices in the set I in ascending order.
     - You may use the `sort` function in the `algorithm` library.

   In the following examples, green nodes belong to I.

   Example 1:



   Input:

   8 11

   1 2 3

   1 3 5

   1 4 1

   2 5 5

   3 6 6

   6 3 -3

   4 7 4

   7 4 -8

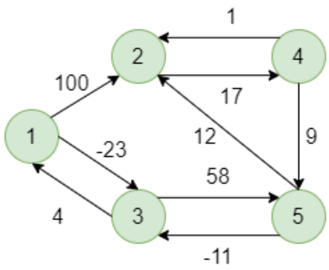
   5 8 -5

   6 8 4

7 8 12

Expected output:
4 7 8

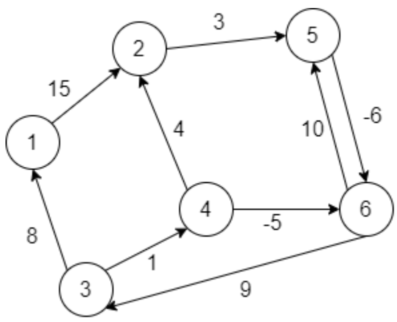
Example 2:



Input:
5 9
1 2 100
1 3 -23
3 1 4
2 4 17
4 2 1
3 5 58
5 3 -11
5 2 12
4 5 9

Expected output:
1 2 3 4 5

Example 3:



Input:

6 9

1 2 15

2 5 3

3 1 8

4 2 4

3 4 1

4 6 -5

6 3 9

5 6 -6

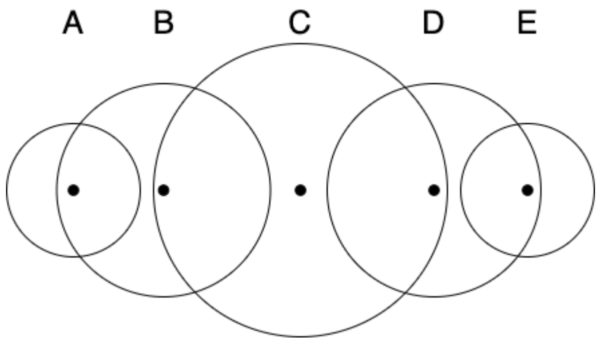
6 5 10

Expected output (empty):

2. (10 pts) Write a C++ implementation of the pseudocode (with some modifications) you developed for problem (3b) and submit to INGInious as assignment_2_q_2.cpp.

   - Input will come from cin
     - The first line will contain a single integer n, the number of outposts.
     - The next n lines will each contain three space separated integers, x, y, and s.
     - Outpost i is at coordinates (x_i, y_i) with a transmitter of strength s_i.
   - Print output to cout
     - A single integer indicating the maximum number of outposts that can be contacted as a result of a single trip to the surface.
   - Example input and output files are available on Canvas.

   Below is a cartoon example of how transmitters from five outposts, A, B, C, D, and E, interact. Notice that all outposts can receive their messages with a single trip from the space station to outpost C. Once this outpost receives messages, it can send them along to outposts B and D. Finally, these two outposts can send messages to A and E respectively. In particular, the solution in this case would be 5.



3. (10 pts) There are many ways in which we might define the "importance" of a node in a graph. One idea is to consider the total number of shortest paths that pass through the node. This is the idea behind betweenness centrality. In particular, the betweenness centrality of node u is

$$betweenness(u) = \sum_{s \neq u \neq t \in V} \frac{\sigma_{st}(u)}{\sigma_{st}}$$

where σ_{st} is the number of shortest paths from s to t and $\sigma_{st}(u)$ is the number of these paths passing through u.

As a step towards calculating this value for a given node, implement a function `countShortestPaths(G, s, u)` that determines the number of shortest paths from s to u in G. How might this function be used to determine the betweenness centrality of u? Submit your implementation to INGInious as assignment_2_q_3.cpp.

- Input will come from cin
  - The first line will contain four integers, n, m, s, and u, separated by spaces.
  - n is a number of vertices, m is a number of edges, s is the start vertex, and u is the destination.
  - The next m lines will each contain two integers, v and w, separated by a space.
  - Each of these pairs represents an undirected edge $\{v, w\}$.
- Print output to cout
  - A single integer indicating the number of shortest paths from s to u.
- Example input and output files are available on Canvas.