Name: Jaryd Meek

ID:

Collaborators: Noah Nguyen, Emily Parker

CSCI 3104, Algorithms **Due March 12, 2021**

Problem Set 7 (50 points) **Spring 2021, CU-Boulder**

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.
Advice 2: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

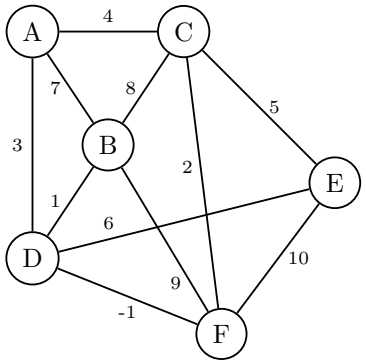
Instructions for submitting your solution:

- The solutions **should be typed** and we cannot accept hand-written solutions. Here's a short intro to Latex.
- You should submit your work through **Gradescope** only.
- The easiest way to access Gradescope is through our Canvas page. There is a Gradescope button in the left menu.
- Gradescope will only accept **.pdf** files.
- It is vital that you match each problem part with your work. Skip to 1:40 to just see the matching info.

Name: Jaryd Meek
ID:
Collaborators: Noah Nguyen, Emily Parker

CSCI 3104, Algorithms **Due March 12, 2021**
Problem Set 7 (50 points) **Spring 2021, CU-Boulder**

1. Consider the following graph:



   (a) Use Kruskal's algorithm to compute the MST. It will suffice to indicate the order in which edges are added to the MST.

   (b) Now use Prim's algorithm starting at node A to compute the MST, again by indicating the order in which edges are added to the MST.

   (c) Is it possible to change the starting node for Prim's algorithm such that it adds edges to the MST in the same order as Kruskal's algorithm? If so, which starting node(s) would work? Justify your answer.

   Note: For parts (a) and (b), let (Node1, Node2) represent the edge between two nodes Node1 and Node2. Therefore, your answer should have the form: (Node1, Node2), (Node4, Node5), etc.

Solution:

(a) (D,F), (B,D), (C,F), (A,D), (C,E)

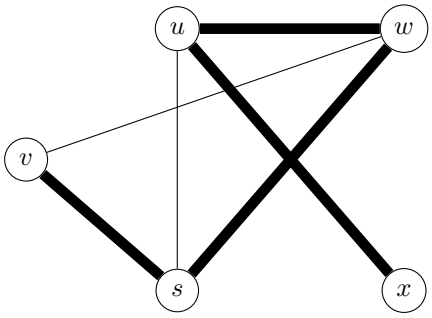
(b) (A,D), (D,F), (D,B), (F,C), (C,E)

(c) Yes, if you start at D, Prim's algorithm will add edges to the the MST in the same order of Kruskal's algorithm.
Performing Prim's algorithm -
Start at D, Check each adjacent edge for the lowest value.
The lowest value is -1, so add vertex F to the MST
Our MST is now (D,F)
Check again for next lowest value connected to our new graph, in this case, it will be 1, connecting D and B
Our MST is now (D,F), (B,D)
Continue to next lowest edge weight attached to our current MST, in this case, it's 2 from C to F
Our MST is now (D,F), (B,D), (C,F)
Continue to next lowest edge weight attached to our current MST, it's 3 from A to D
Our MST is now (D,F), (B,D), (C,F), (A,D)
Continue to next lowest edge weight attached to our current MST, it's 4 from A to C
Since this would create a cycle so skip it,
Continue to next lowest edge weight attached to our current MST, it's 5 from C to E
Our MST is now (D,F), (B,D), (C,F), (A,D), (C,E)
Since this matches our answer from (a), there is a node that will produce the same MST with nodes added in the same order, using Kruskal's algorithm and Prim's algorithm

Name: Jaryd Meek
ID:
Collaborators: Noah Nguyen, Emily Parker

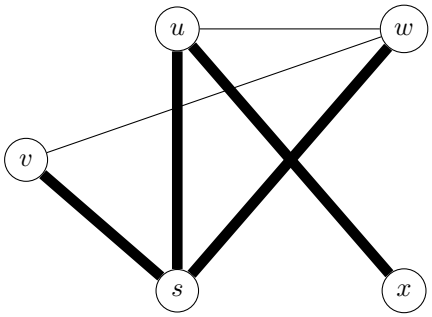
CSCI 3104, Algorithms **Due March 12, 2021**
Problem Set 7 (50 points) **Spring 2021, CU-Boulder**

2. Consider the undirected, unweighted graph $G = (V, E)$ with $V = \{s, u, v, w, x\}$ and $E = \{(s, u), (s, v), (s, w), (u, w), (u, x), (v, w)\}$, and let $T \subset E$ be $T = \{(s, v), (s, w), (u, w), (u, x)\}$. This is pictured below with T represented by wide edges.



Demonstrate that T cannot be output by BFS with start vertex s.

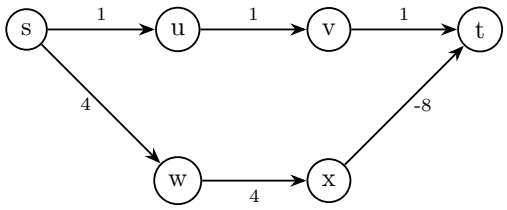
Solution:
T can not be output by BFS with start vertex s since the tree does not visit all nodes at a given depth before going deeper. The vertex (u, s) would have to be visited to be a valid BFS. If it were a valid BFS, The order visited (starting at s) would be (s, v) then (s, u) then (s, w). That would visit all nodes at an equal depth, then you would traverse deeper, and visit (u, x). This would complete the BFS as all nodes would be visited.
The final tree produced by a BFS would look like such.



Name: Jaryd Meek
ID:
Collaborators: Noah Nguyen, Emily Parker

CSCI 3104, Algorithms **Due March 12, 2021**
Problem Set 7 (50 points) **Spring 2021, CU-Boulder**

3. Given the following directed graph $G = (V, E)$ with starting and ending vertices $s, t \in V$, show that Dijkstra's algorithm does *not* find the shortest path between s and t.



Solution:
Since the version of Dijkstra's algorithm stops when it finds the correct node, Dijkstra's algorithm on this graph will go $S \to U \to V \to T$ then stop. Since it terminates before traversing down the lower path (since the final distance will be 3, which is less than the 4 needed to go down the lower path to w), it will never encounter the negative weight, causing it to pick a non-ideal path.

Name: Jaryd Meek
ID:
Collaborators: Noah Nguyen, Emily Parker

4. Given a graph, implement Prim's algorithm via Python 3. The input of the Graph class is an Adjacency Matrix. Complete the Prim function. The Prim fucntion should return the weight sum of the minimum spanning tree starting from node 0.

   The file `graph.py` is provided; use this file to construct your solution and upload with the same name. You may add class variables and methods but **DO NOT MODIFY THE PROVIDED FUNCTION OR CLASS PROTOTYPES.**.

   Here is an example of how your code will be called:

   Sample input:

```
g = Graph([ [0, 10, 11, 33, 60],
            [10, 0, 22, 14, 57],
            [11, 22, 0, 11, 17],
            [33, 14, 11, 0, 9],
            [60, 57, 17, 9, 0]])

assert g.Prim() == 41
```