CS 330 – Fall 2021
Homework 8, Problem 2: Lightest Directed Cycle
Due November 3, 2021

Your goal is to design an algorithm that, given a directed graph G with edge weights, finds the lightest directed cycle in the graph.

Inputs An *directed* graph $G = (V, E)$ with positive edge weights.

Outputs The weight of the lightest cycle in the graph, as well as a list of nodes in the cycle. Any cycle with two or more edges is acceptable (so a cycle of the form u, v, u is ok).

Design an algorithm for this task and program it in Python.

Your algorithm should run in worst-case time $O(nm \log n)$ (assuming $m = \Omega(n)$). A simple strategy is to modify Dijkstra's algorithm to find the shortest cycle from each node s .

Additional optimizations: To get full credit, your algorithm will have to run quickly on graphs where running n full copies of Dijkstra's algorithm would take too long. To do this, you will have to implement additional optimizations. For example, one idea is to ensure that your algorithm never explores paths that are longer than the shortest cycle found so far. The autograder will provide feedback on running time. Of course, you should focus at first on getting an algorithm that is correct!

Starter Code We've provided starter code that consists of

- `simplegraphs.py` (version 2) : A collection of functions for manipulating graphs as python dictionaries. There are procedures for reading a graph from a file that describes it, writing the graph to a file, and other basic manipulations. It includes code for BFS and DFS and Dijkstra's algorithm.
  You can use any of these in your solution. Do not change the file `simplegraphs.py` (since the autograder will include the version "v2.0" that we distributed). You can create your own version and modify the functions there if you like.
- `shortestCycle.py` : A template for your actual solution. There are functions for reading the input and writing the output (see below).
- Test inputs and outputs. You should do your own testing!

Program Format You should submit a file called `shortestCycle.py` containing your solution (we'll give you a template to start from). We will test your program by running the following command:

```
python shortestCycle.py graph_file output_file
```

There are two arguments, each of which is the name of a file

- `graph_file` describes the graph. This uses the format that is read by `simplegraphs.py`. The first two lines of the file list the number of nodes n and the number of directed edges in the graph. The remaining lines list the edges in the graph, assuming the node IDs are $\{1, .., n\}$. Each line lists two vertices (ints) and a weight (float).

- `output_file` is where your program will write its output. Code for writing the output is provided. The first line is an float (the shortest cycle length, or “inf” if no cycle was found) and the third line is a list of the IDs of the nodes in the shortest cycle (an empty list if no cycle was found). The node IDs should be in order, but they can start anywhere on the cycle. (The autograder checks that they form a directed cycle of the correct weight.)