```
CS 519-005, Algorithms (MS/MEng-level), Winter 2018
HW9 - Graph Algorithms (part 2), DP (part 4)

Due Monday March 12, 11:59pm.
No late submission will be accepted.

PLEASE SET UP AN INTERNAL DEADLINE ON FRIDAY MARCH 9 TO SAVE MORE TIME FOR HW10!

Include in your submission: report.txt, dijkstra.py, tsp.py.
dijkstra.py will be graded for correctness (1%).

Textbooks for References:
[1] CLRS Ch. 22 (graph), Ch. 15 (DP)
[2] my DP tutorial (up to page 16):
    http://web.engr.oregonstate.edu/~huanlian/slides/COLING-tutorial-anim.pdf
[3] DPV Ch. 3, 4.2, 4.4, 4.7, 6 (Dasgupta, Papadimitriou, Vazirani)
    https://www.cs.berkeley.edu/~vazirani/algorithms/chap3.pdf
    https://www.cs.berkeley.edu/~vazirani/algorithms/chap4.pdf
    https://www.cs.berkeley.edu/~vazirani/algorithms/chap6.pdf
[4] KT Ch. 6 (DP)
    http://www.aw-bc.com/info/kleinberg/assets/downloads/ch6.pdf
[5] KT slides: Greedy II (Dijkstra)
    http://www.cs.princeton.edu/~wayne/kleinberg-tardos/
[6] Wikipedia: Traveling Salesman Problem
[7] Wikipedia: Held-Karp Algorithm (1962) for TSP


***Please answer time/space complexities for each problem in report.txt.

1. [WILL BE GRADED]
   Dijkstra (see CLRS 24.3 and DPV 4.4)

   Given an undirected graph, find the shortest path from source (node 0)
   to target (node n-1).

   [UPDATE]
   Edge weights are guaranteed to be non-negative, since Dijkstra doesn't work
   with negative weights, e.g.

       3
   0 ------ 1
     \    /
    2 \  / -2
       \/
       2

   in this example, Dijkstra would return length 2 (path 0-2),
   but path 0-1-2 is better (length 1).

   [UPDATE]
   For example (return a pair of shortest-distance and shortest-path):

       1
   0 ------ 1
     \    /  \
    5 \  /1   \6
       \/   2  \
       2 ------ 3

   >>> shortest(4, [(0,1,1), (0,2,5), (1,2,1), (2,3,2), (1,3,6)])
   (4, [0,1,2,3])

   [UPDATE] the (2,3) edge should be (2,3,2) not (2,3,1).

   Filename: dijkstra.py


2. Traveling Salesman Problem (TSP).
```

```
   Given an undirected graph of n nodes (0..n-1) representing a road network,
   the traveling salesman has to start from city 0 and visit each city
   once and only once, and return to city 0. Find the minimum-length tour (cycle)
   that satisifies these conditions (this is also called "Hamiltonian Cycle").

   Write the subproblem definition, recurrence relation, and space/time complexities in
report.txt.

   Input: same as Dijkstra
   Output: (cycle_length, cycle_list)
   Tiebreaking: arbitrary

   e.g., for the above example in Dijkstra, one possible best cycle is 0-1-3-2-0, with a
cost of 14.

   >>> tsp(4, [(0,1,1), (0,2,5), (1,2,1), (2,3,2), (1,3,6)])
   (14, [0,1,3,2,0])

   If we add an edge (3,0,1), then the best cycle cost reduces to 5:

   >>> tsp(4, [(0,1,1), (0,2,5), (1,2,1), (2,3,2), (1,3,6), (3,0,1)])
   (5, [0,1,2,3,0])

   Note: This problem can be solved by either Viterbi (recommended) or Dijkstra.
         The classical Edmonds-Karp TSP algorithm is an instance of the former.

   Additional real-world examples: # from a map of germany:
https://stackoverflow.com/questions/11007355/data-for-simple-tsp

   >>> tsp(11, [(0,1,29),(0,2,20),(0,3,21),(0,4,16),(0,5,31),(0,6,100),(0,7,12),(0,8,4),
(0,9,31),(0,10,18),
                (1,2,15),(1,3,29),(1,4,28),(1,5,40),(1,6,72),(1,7,21),(1,8,29),(1,9,41),
(1,10,12),
                (2,3,15),(2,4,14),(2,5,25),(2,6,81),(2,7,9),(2,8,23),(2,9,27),(2,10,13),
                (3,4,4),(3,5,12),(3,6,92),(3,7,12),(3,8,25),(3,9,13),(3,10,25),
                (4,5,16),(4,6,94),(4,7,9),(4,8,20),(4,9,16),(4,10,22),
                (5,6,95),(5,7,24),(5,8,36),(5,9,3),(5,10,37),
                (6,7,90),(6,8,101),(6,9,99),(6,10,84),
                (7,8,15),(7,9,25),(7,10,13),
                (8,9,35),(8,10,18),
                (9,10,38)]))

   (253, [0, 8, 10, 1, 6, 2, 5, 9, 3, 4, 7, 0])

   (Viterbi: 0.0s; Dijkstra: 0.3s)

   Random examples:

   >>> tsp(16, [(1, 2, 0), (11, 5, 5), (9, 8, 4), (6, 1, 4), (5, 13, 5), (12, 11, 4), (14,
8, 0), (0, 11, 3), (10, 12, 3), (5, 5, 1), (7, 0, 1), (10, 5, 1), (11, 5, 3), (13, 11, 4),
(11, 11, 3), (5, 12, 5), (14, 7, 3), (8, 15, 4), (11, 14, 3), (11, 14, 3), (7, 10, 5), (5,
8, 3), (9, 9, 5), (13, 9, 5), (6, 15, 4), (11, 2, 2), (0, 6, 5), (3, 1, 4), (1, 8, 4), (7,
3, 4), (4, 8, 1), (6, 1, 3), (1, 1, 2), (11, 5, 1), (0, 2, 0), (2, 0, 0), (0, 11, 2), (4,
5, 5), (5, 0, 3), (1, 7, 1), (1, 0, 2), (3, 9, 2), (15, 0, 2), (14, 1, 2), (12, 4, 3), (7,
2, 5), (10, 3, 0), (14, 4, 4), (12, 15, 4), (10, 4, 2), (8, 8, 4), (13, 0, 5), (4, 1, 2),
(1, 4, 1), (5, 3, 3), (7, 1, 1), (7, 14, 0), (8, 2, 4), (7, 11, 2), (13, 8, 4), (0, 4, 0),
(12, 13, 1), (3, 2, 1), (3, 3, 0), (5, 7, 0), (6, 0, 4), (14, 14, 2), (12, 6, 5), (6, 13,
3), (0, 1, 3), (5, 3, 5), (15, 11, 0), (3, 11, 2), (11, 9, 0), (13, 3, 0), (9, 6, 5), (0,
14, 0), (13, 15, 3), (6, 2, 0), (9, 0, 2), (9, 2, 1), (15, 6, 0), (11, 12, 5), (14, 4, 2),
(12, 3, 2), (3, 3, 0), (10, 12, 1), (3, 0, 4), (15, 1, 5), (15, 9, 2), (14, 4, 2), (8, 15,
4), (15, 13, 3), (9, 12, 1), (5, 15, 4), (8, 13, 5), (2, 3, 0), (11, 5, 4), (4, 13, 0),
(2, 1, 1)])

   (6, [0, 4, 8, 14, 7, 5, 10, 3, 13, 12, 9, 11, 15, 6, 2, 1, 0])

   (Viterbi: 2.1s, Dijkstra: 0.9s)


   Filename: tsp.py
```

```
Debriefing (required!): --------------------------

0. What's your name?
1. Approximately how many hours did you spend on this assignment?
2. Would you rate it as easy, moderate, or difficult?
3. Did you work on it mostly alone, or mostly with other people?
4. How deeply do you feel you understand the material it covers (0%—100%)?
5. Which part(s) of the course you like the most so far?
6. Which part(s) of the course you dislike the most so far?

This section is intended to help us calibrate the homework assignments.
Your answers to this section will *not* affect your grade; however, skipping it
will certainly do.
```