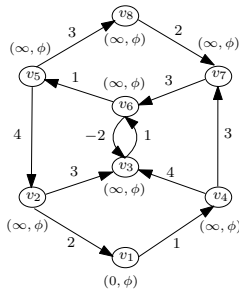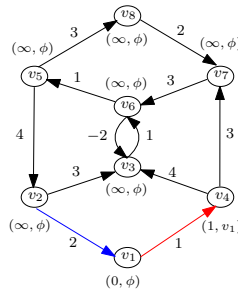# Solution to Problem 1:   Bellman-Ford Algorithm

Edge relaxation order: $(v_1, v_4), (v_2, v_1), (v_2, v_3), (v_3, v_6), (v_4, v_3), (v_4, v_7), (v_5, v_2), (v_5, v_8), (v_6, v_3), (v_6, v_5), (v_7, v_6), (v_8, v_7)$
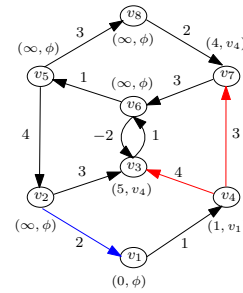
Node Attributes: $(distance\_estimate, parent)$

Successful edge relaxations in RED, failed edge relaxations in BLUE.

In the iteration $|V| = 8$, we observe an update of distance estimate due to edge relaxation.

In other words, we have a negative weight cycle.

Solution to Problem 2:   Dijkstra's Algorithm *50 points*

Data Structure: Min. Priority Queue implemented using min-heaps
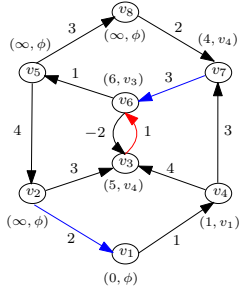
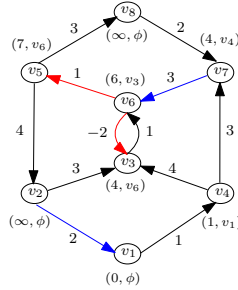Node Attributes: $(distance\_estimate, parent)$
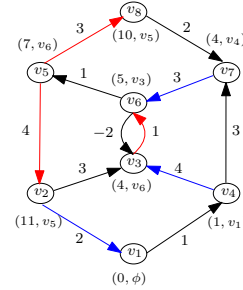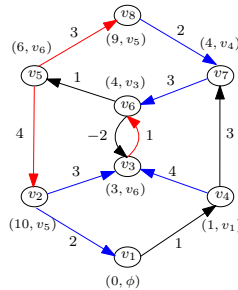

Initialization

Iteration 1

Iteration 2

Iteration 3

Iteration 4
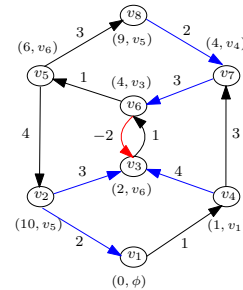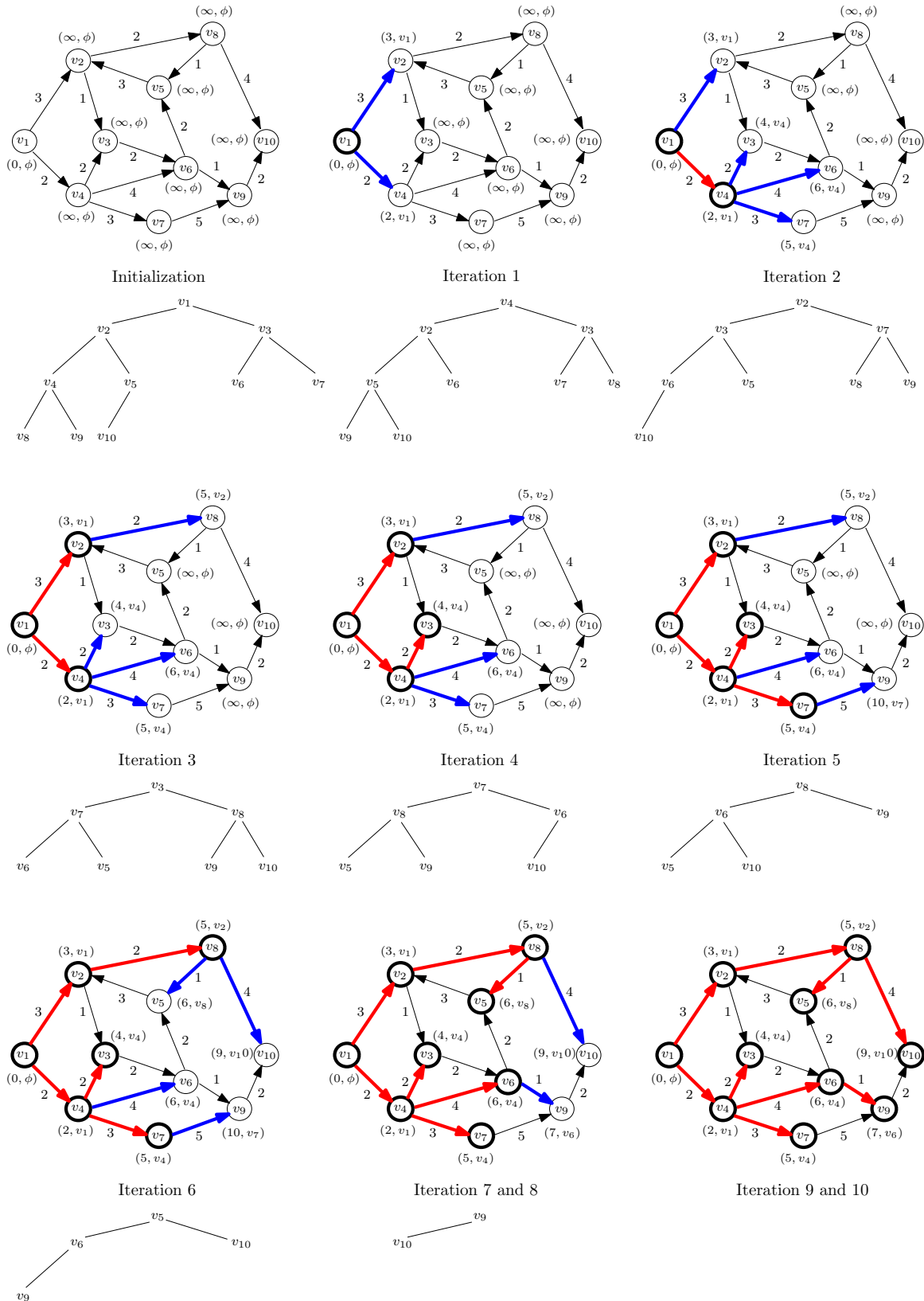

Iteration 5

Iteration 6

Iteration 7 and 8

Iteration 9 and 10

Bonus Problem (10 points): Implement Dijkstra's algorithm on graphs represented as adjacency lists in Python, and validate your code on the example graph given in Problem 2.

Solution: Code is not included in this solution set.