Data Structures Lab 12 Graphs 2 Name:_____

Objectives: To understand how a graph can be used to solve graph algorithms.

To start the lab: Download and unzip the file lab12.zip

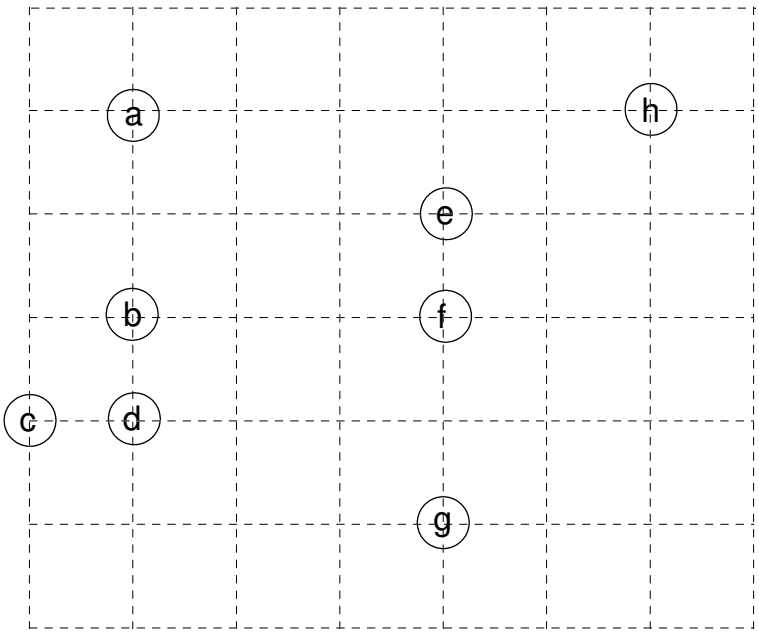
Part A: In IDLE open the binary heap class file: `lab12/binheap.py`. The changes discussed in lecture are included:

- `PriorityQueueEntry` class, and
- additional `BinHeap` methods: `__contains__` and `decreaseKey`.

a) Look at the `buildHeap` method. Explain how it takes a list of values and builds them into a heap.

b) Run the `lab12/make_min_spanning_tree.py` program which uses Prim's algorithm on the graph from lecture. Does it give the expected output?

c) Predict the order of edges added by Prim's algorithm if we start at vertex "e":



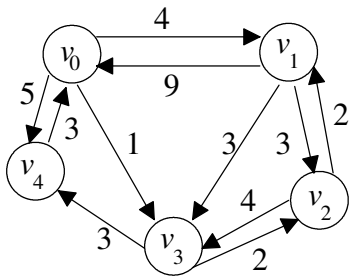
d) Modify the `lab12/make_min_spanning_tree.py` program to verify your prediction. NOTE: This is a very easy modification. You just need to start Prim's algorithm starting at the vertex labeled "e".

After you have answered the above questions and completed the code, raise your hand and explain your answers.

Part B: The textbook's Dijkstra's Algorithm code (Listing 7.11 p. 341) needs to be updated similarly to Prim's algorithm.

a) Modify the `dijkstra` method in the `lab12/graph_algorithms.py` file similar to Prim's so that it uses `PriorityQueueEntry` objects among other corrections.

b) Run the `lab12/test_dijkstra.py` program which uses Dijkstra's algorithm on the graph from lecture. Does it give the expected output?



After you have fixed the dijksta method in `lab12/graph_algorithms.py`, raise your hand and demonstrate your code.

EXTRA CREDIT Opportunities:

1. Add code to the end of the `test_dijkstra.py` program to print the shortest paths from v_0 to each of the other vertices. One line of output might look something like:
   "Shortest path from v0 to v4 is v0 > v3 > v4 with a total distance of 4"

2. As implemented the decreaseKey method must do a O(n) search for the item whose value is being decreased. To avoid this search, modify the PriorityQueueEntry object to maintains an "index" data attribute indicating its index in the BinHeap's heapList. Several of the BinHeap methods must also be modified to keep this index data attribute up to date.