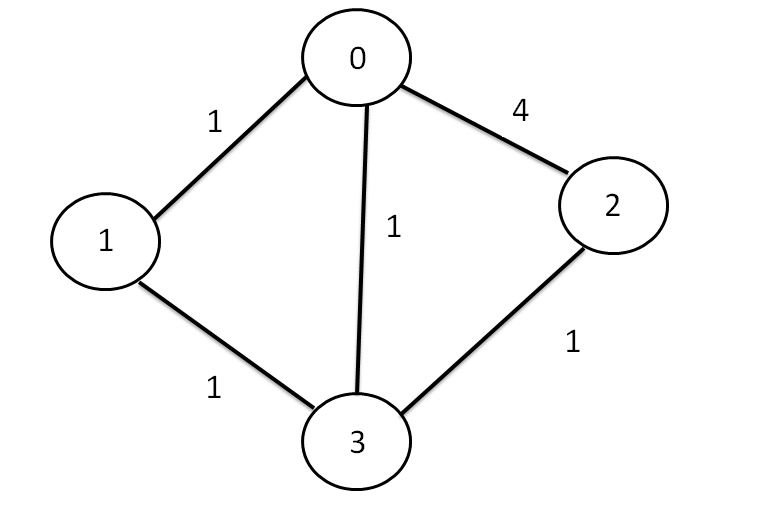
QUIZ 6

* Give a graph where Prim's and Kruskal's algorithms generate different kinds of minimum spanning trees. Explain how these trees are generated. If you cannot find an example, explain why these algorithms always generate the same minimum spanning tree given any graph.



Example:

This is an example where Kruskal and Prim algorithms yield different MSTs. I have implemented the graph above in the MST Test algorithm and here are the results.

MST Prim

0 <- 1 : 1.000000

0 <- 3 : 1.000000

3 <- 2 : 1.000000

3.0

MST Kruskal

0 <- 1 : 1.000000

3 <- 2 : 1.000000

3 <- 1 : 1.000000

3.0

Both result in total weight of 3, however they are different trees. This happens when the user generates a graph with all or most edges having the same weight attached to them. For Prim it adds all the edges connected to the vertex that has the least weight to a PriorityQueue and then polls edges from that PriorityQueue which have the least weights and then adds the edge(s) with the next least weights to build the tree until the PQ is empty. In my example, first we add the edge from 1 to 0 followed by 3 to 0, even though 3 to 1 is also a valid option. After we add 3 to 0, 3 to 1 is not an option anymore (cycle) so we add 2 to 3.

For Kruskal we add all the edges to the PQ first and then poll edges with the least weight and merge trees. First we choose 1 to 0 and then 3 to 2 even though 0 to 3 and 1 to 3 are also options. This is followed by 3 to 1 with 0 to 3 ignored creating a different tree than Prim’s algorithm.

Both algorithms, due to the Java implementation of Priority queue: “If multiple elements are tied for least value, the head is one of those elements -- ties are broken **arbitrarily**.”, yield different trees when most edges carry identical weights.

One other scenario would be where 0, the starting vertex and the edges connected to it have relatively greater weights than other edges. In this case Kruskal would start with those edges but Prim would have to start with edges necessarily connected to 0, the starting edge.