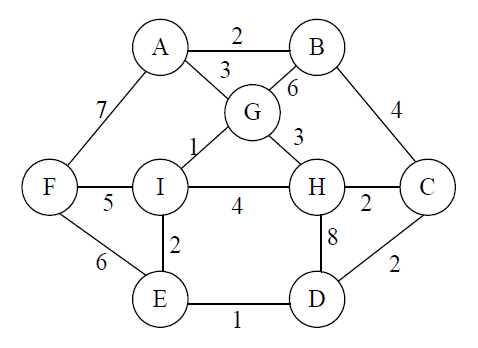
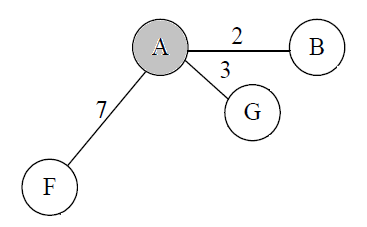
**CMSC 451 Homework 5**

1. Execute Prim’s minimum spanning tree algorithm by hand on the graph below showing how the data structures evolve specifically indicating when the distance from a fringe vertex to the tree is updated. Clearly indicate which edges become part of the minimum spanning tree and in which order. Start at vertex A.

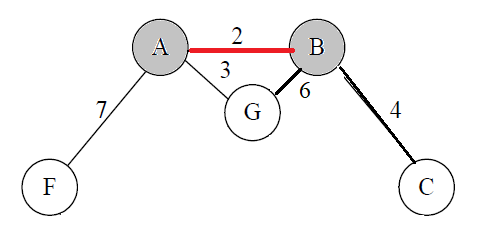
|  |  |  |
| --- | --- | --- |
| A | F | 0 |
| B |  |  |
| C |  |  |
| D |  |  |
| E |  |  |
| F |  |  |
| G |  |  |
| H |  |  |
| I |  |  |



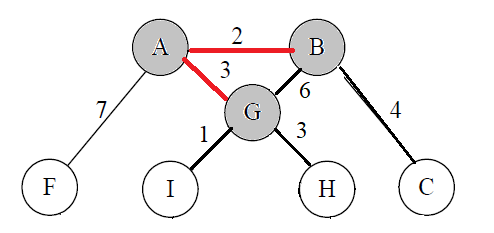
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | F | 2 | Minimum |
| C |  |  |  |
| D |  |  |  |
| E |  |  |  |
| F | F | 7 |  |
| G | F | 3 |  |
| H |  |  |  |
| I |  |  |  |



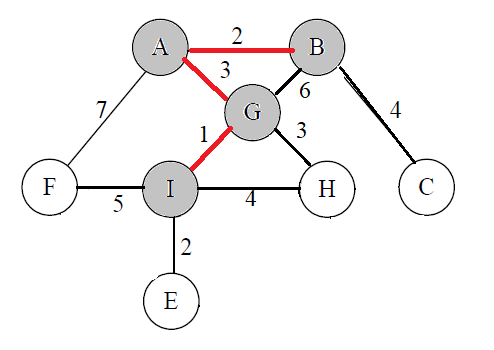
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | F | 4 |  |
| D |  |  |  |
| E |  |  |  |
| F | F | 7 |  |
| G | F | 3 | Minimum |
| H |  |  |  |
| I |  |  |  |



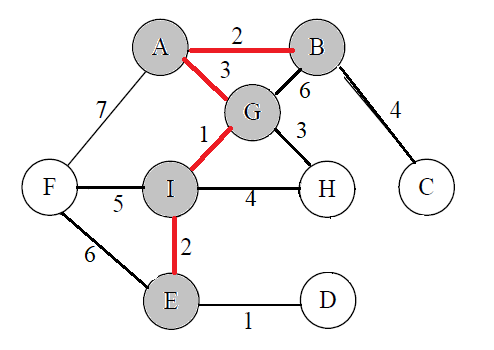
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | F | 4 |  |
| D |  |  |  |
| E |  |  |  |
| F | F | 7 |  |
| G | T |  |  |
| H | F | 3 |  |
| I | F | 1 | Minimum |



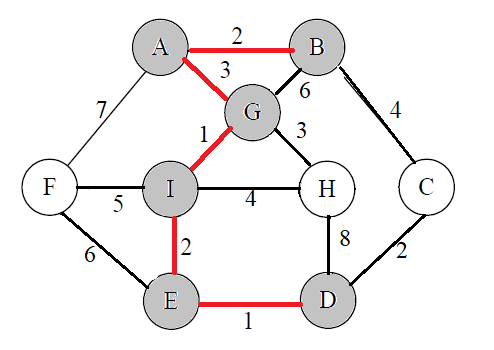
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | F | 4 |  |
| D |  |  |  |
| E | F | 2 | Minimum |
| F | F | 5 | Updated |
| G | T |  |  |
| H | F | 3 |  |
| I | T |  |  |



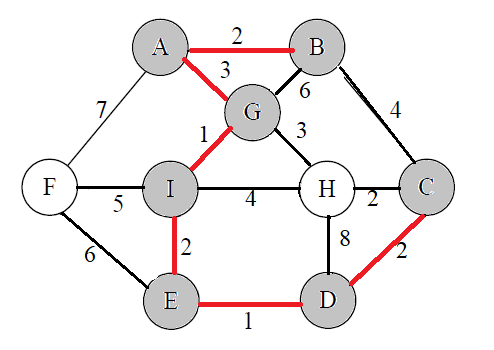
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | F | 4 |  |
| D | F | 1 | Minimum |
| E | T |  |  |
| F | F | 5 |  |
| G | T |  |  |
| H | F | 3 |  |
| I | T |  |  |



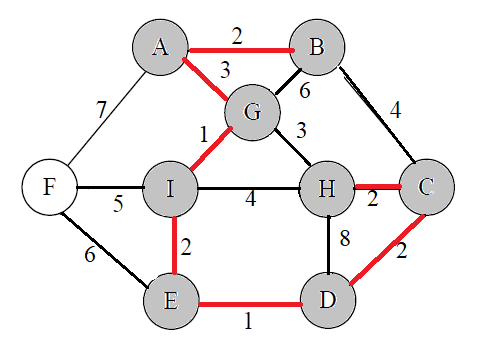
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | F | 2 | Updated,  Minimum |
| D | T |  |  |
| E | T |  |  |
| F | F | 5 |  |
| G | T |  |  |
| H | F | 3 |  |
| I | T |  |  |



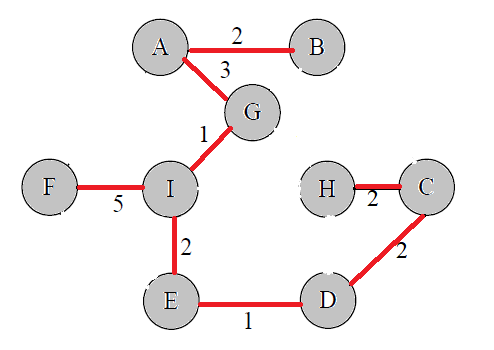
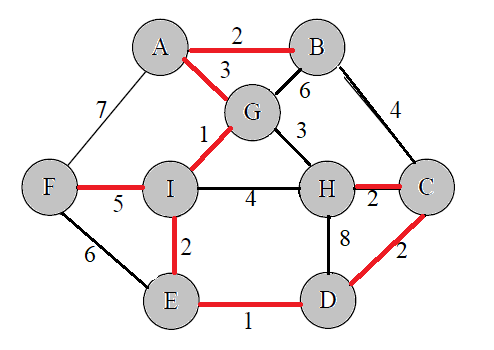
|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | T |  |  |
| D | T |  |  |
| E | T |  |  |
| F | F | 5 |  |
| G | T |  |  |
| H | F | 2 | Updated,  Minimum |
| I | T |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | T |  |  |
| D | T |  |  |
| E | T |  |  |
| F | F | 5 | Minimum |
| G | T |  |  |
| H | T |  |  |
| I | T |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| A | T |  |  |
| B | T |  |  |
| C | T |  |  |
| D | T |  |  |
| E | T |  |  |
| F | T |  |  |
| G | T |  |  |
| H | T |  |  |
| I | T |  |  |



2. Execute Kruskal’s algorithm on the weighted tree shown below. Assume that edges of equal weight will be in the priority queue in alphabetical order. Clearly show what happens each time an edge is removed from the priority queue and how the dynamic equivalence relation changes on each step and show the final minimum spanning tree that is generated.

**Choose AB 1 (A, B)**

**Choose GH 1 (A, B) (G, H)**

**Choose JM 1 (A, B) (G, H) (J, M)**

**Choose DE 2 (A, B) (D, E) (G, H) (J, M)**

**Choose EF 2 (A, B) (D, E, F) (G, H) (J, M)**

**Choose HM 2 (A, B) (D, E, F) (G, H, J, M)**

**Skip DF 3**

**Choose FM 3 (A, B) (D, E, F, G, H, J, M)**

**Choose KL 3 (A, B) (D, E, F, G, H, J, M) (K, L)**

**Choose AD 4 (A, B, D, E, F, G, H, J, M) (K, L)**

**Skip EM 4**

**Choose AC 5 (A, B, C, D, E, F, G, H, J, M) (K, L)**

**Skip GM 5**

**Skip BC 6**

**Choose KM 6 (A, B, C, D, E, F, G, H, J, K, L, M)**

**Skip BG 7**

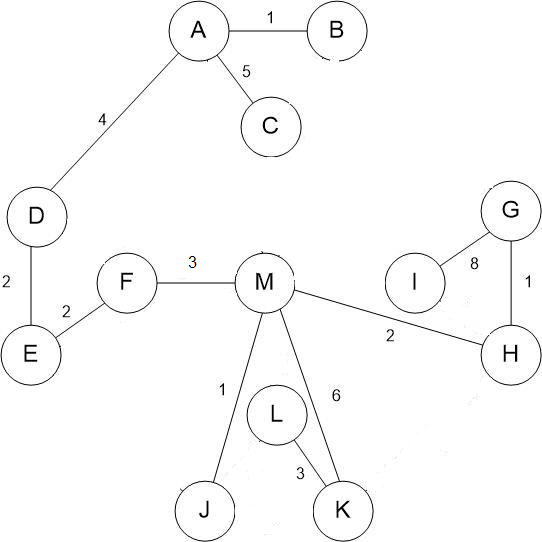
**Skip CM 7**

**Skip JK 7**

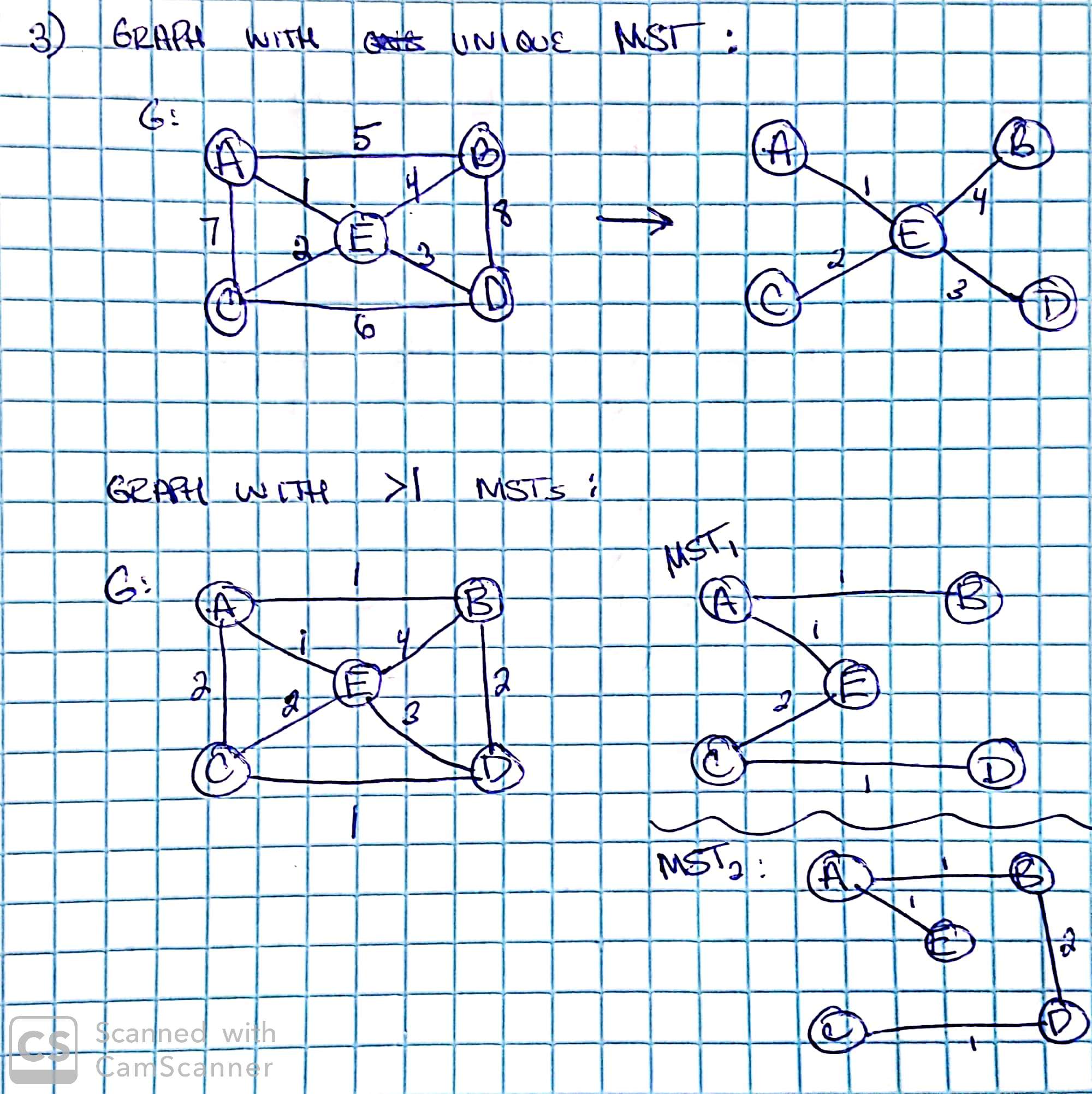
**Skip JL 7**

**Skip AM 8**

**Choose GI 8 (A, B, C, D, E, F, G, H, I, J, K, L, M)**



3. Give an example of a weighted graph for which the minimum spanning tree is unique. Indicate what the minimum spanning tree is for that graph. Give another example of a weighted graph that has more than one minimum spanning tree. Show two different minimum spanning trees for that graph. What determines whether a graph has more than one minimum spanning tree?



**A graph will have more than one spanning tree if any of the vertices have more than one edge of the same weight that will still make a minimum spanning tree. Any easy way to test this is with Prim’s algorithm. When the next subtree is added to the MST-in-progress, if there is more than one minimum in that step, then the choice for which edge is arbitrary and usually implemented by alphabetical order. If we simply choose the other edge next time, then we will have another MST for the same graph.**

4. Given the following adjacency lists (with edge weights in parentheses) for a directed graph:

A: B(5), C(3), D(1)

B: C(1), D(3)

C: B(3), D(7), E(1)

D: A(6), C(3)

E: F(5)

F: D(3), A(4)

Execute Dijkstra’s shortest-path algorithm by hand on this graph, showing how the data structures evolve, with *A* as the starting vertex. Clearly indicate which edges become part of the shortest path and in which order.

**Solution is picture on following pages.**

