CMPS 2200 Introduction to Algorithms – Fall 14

11/4/14

7. Homework

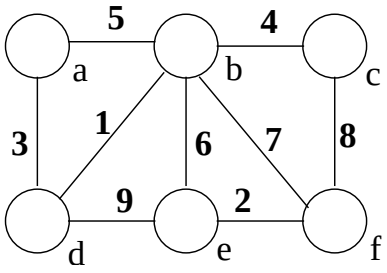
Due **11/13/14** at the beginning of class

Remember, you are allowed to turn in homeworks in groups of two.

1. **Dijkstra (5 points)**
   Run Dijkstra's algorithm on the graph below, with start/source vertex a. (Assume that each undirected graph edge is represented using two directed edges with the same weight.)

   (a) Show all the different stages of the algorithm (vertex weights, tree edges stored in the predecessor array, and the priority queue). Clearly indicate the shortest path tree. You may use a copy of the last page for your convenience.

   (b) What is the shortest path from a to f, and what is its weight?



2. **Kruskal (5 points)**
   Run Kruskal's algorithm on the graph above. Show all the different stages of the algorithm (tree edges and the set of vertex subsets). Clearly indicate the minimum spanning tree. You may use a copy of the next page for your convenience.

3. **Dijkstra and negative edge weights (4 points)**

   Give an example of a directed connected graph with real edge weights (that may be negative) for which Dijkstra's algorithm produces incorrect answers. Justify your answer.

4. **Dijkstra variant (4 points)**
   Consider making the following change to Dijkstra's algorithm:

   ```
   while |Q|>1
   ```

   This means that the while loop runs until the queue consists of one element only. Argue why this change does not impact the correctness of Dijkstra's algorithm, i.e., the outputs of the original algorithm and the modified algorithm are the same.

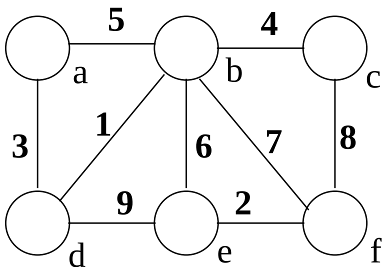
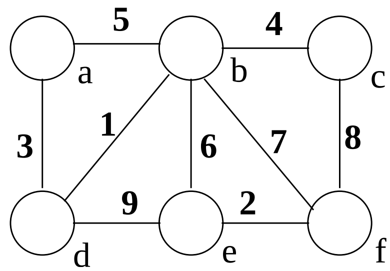
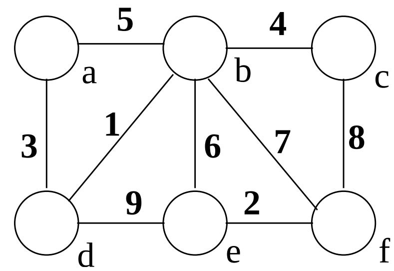
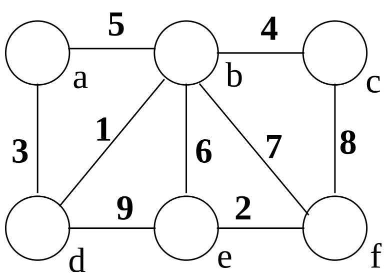
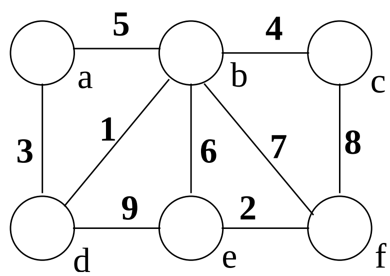
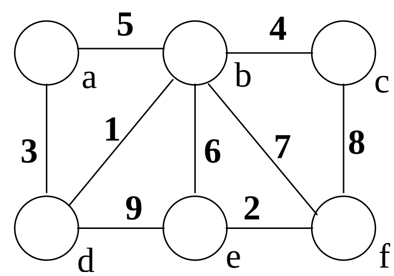
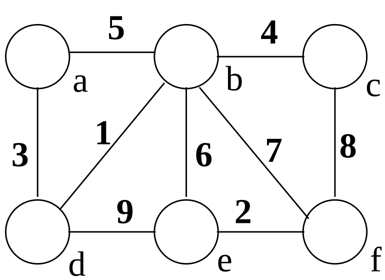
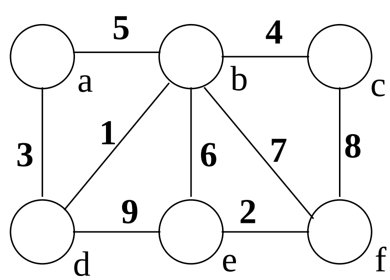
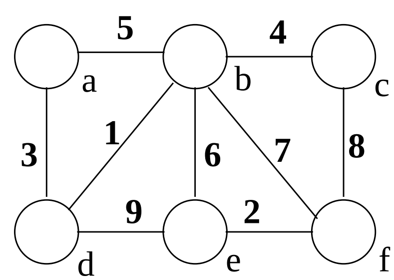
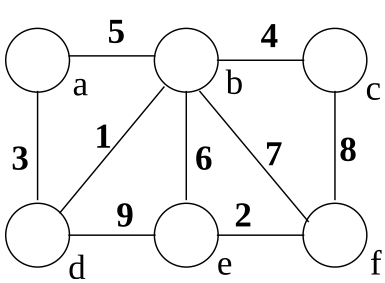
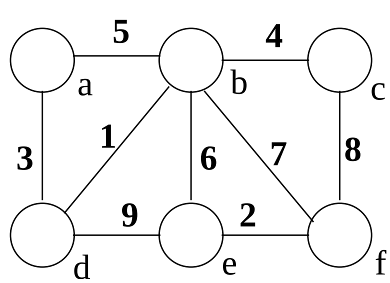
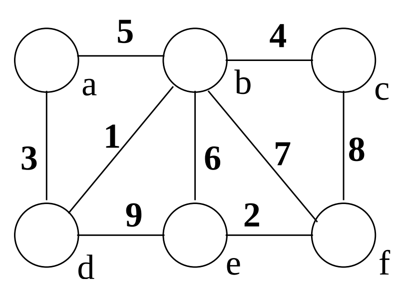
FLIP OVER TO NEXT PAGE ⟹

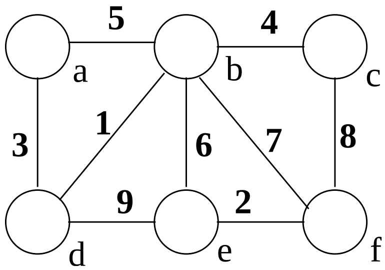
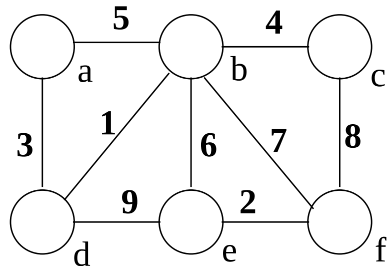
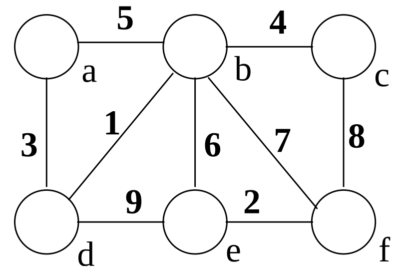
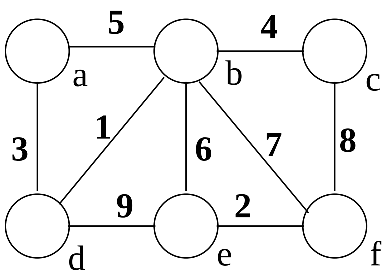
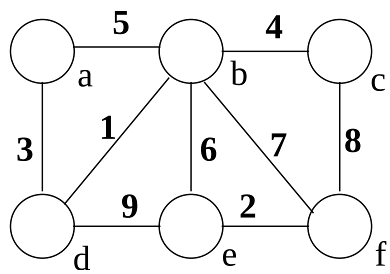
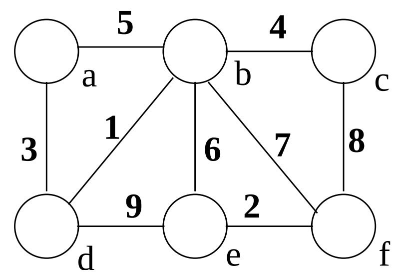
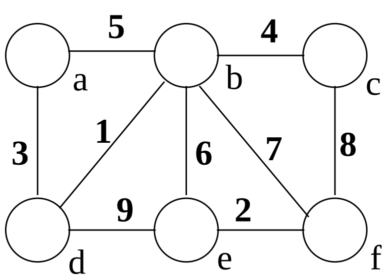
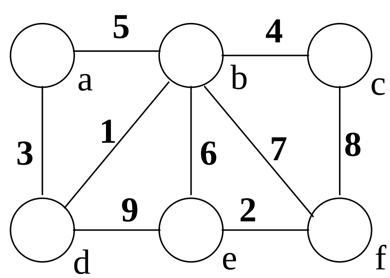
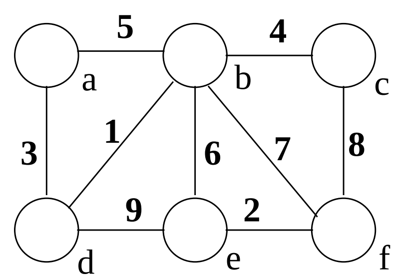
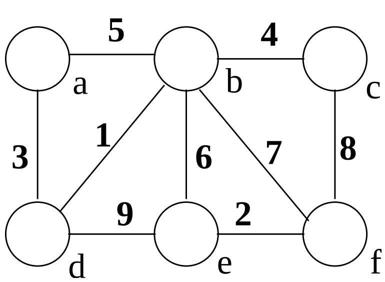
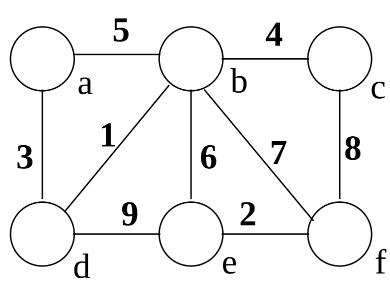
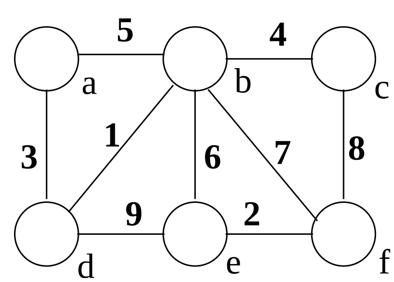
5. **Faster Prim (5 points)**
Let $G = (V, E)$ be a connected undirected graph with edge weights $w : E \to \mathbb{R}$. Assume all edge weights are integers between 1 and 10.

   (a) Show that Prim's algorithm can be implemented to work in $O(|V| + |E|)$ time in this case.
   (Hint: Suggest a data structure based on bucketing vertices with the same weight, that replaces the priority queue and analyze the runtime.)

   (b) (1 point) Argue why or why not the same trick works for Dijkstra's algorithm.

FLIP OVER TO NEXT PAGE ⟹


5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f


5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f

5
4
a
b
c
3
1
6
7
8
9
2
d
e
f