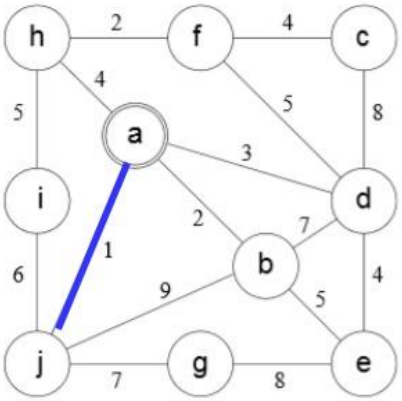
CS 325 Spring 2018 – HW 5

Sheng Bian

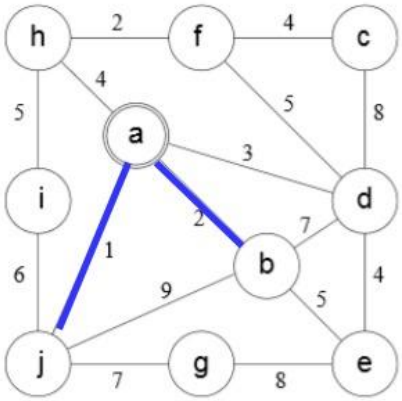
Problem 1:

The weight of the minimum spanning tree is 32. The steps are shown as below.

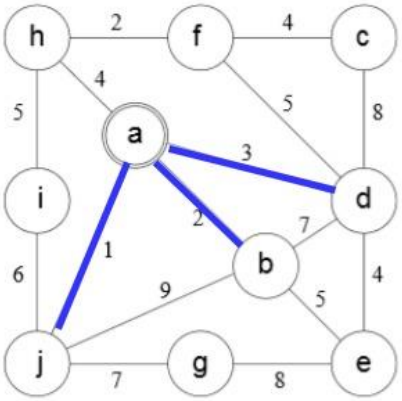
(a)



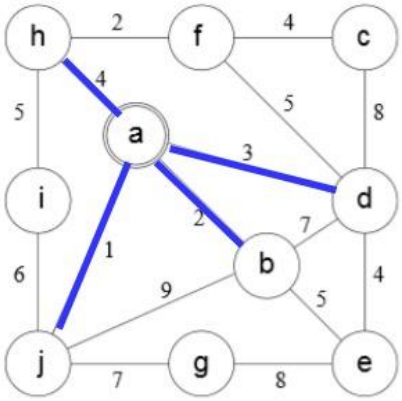
(b)



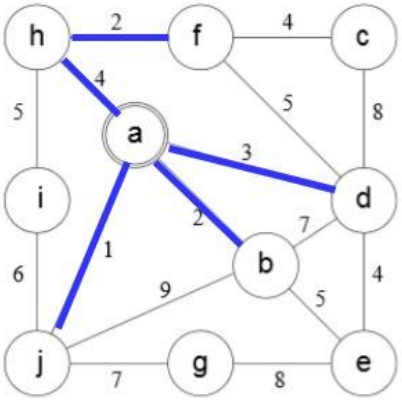
(c)



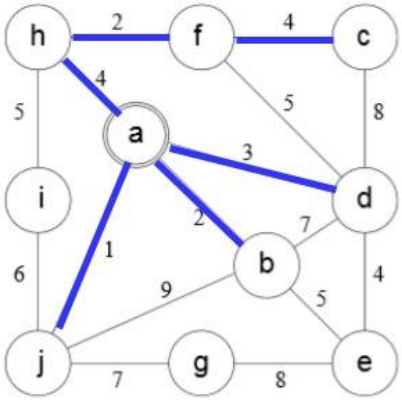
(d)



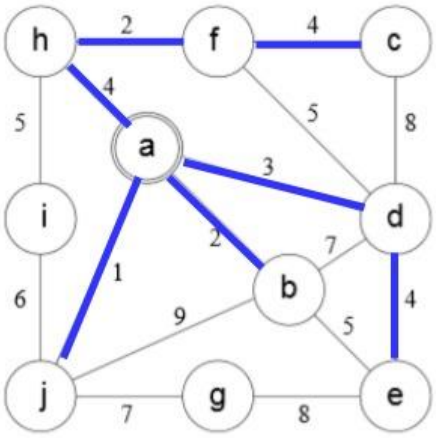
(e)



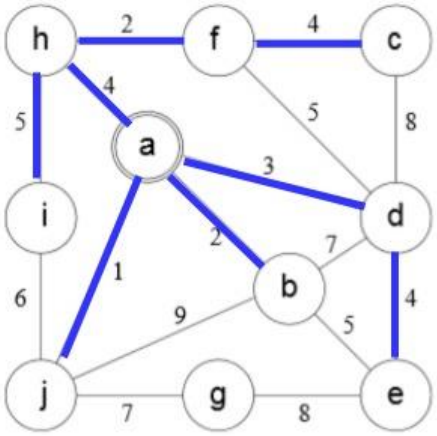
(f)



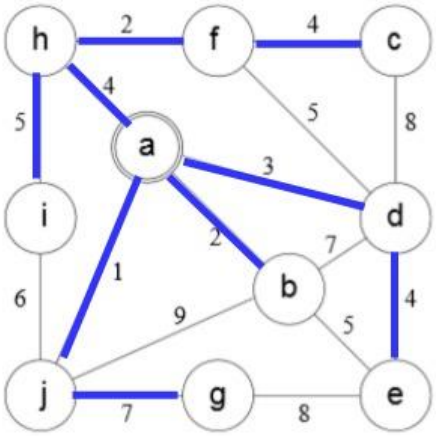
(g)



(h)



(i)



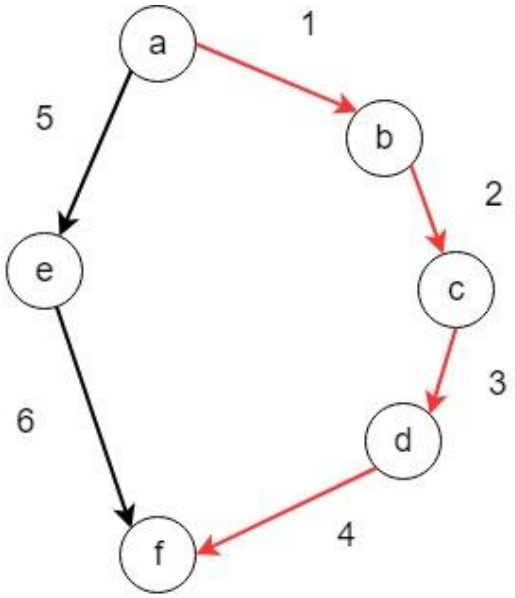
Problem 2:

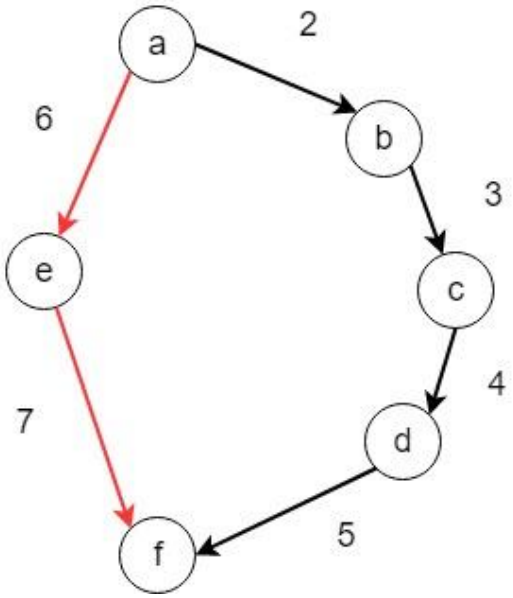
(a)

The minimum spanning tree doesn't change. We can use Kruskal's algorithm to prove it. When we use Kruskal's algorithm to find the minimum spanning tree, we will add the edges one at a time, in increasing weight order. Since we increase each edge weight, the edges and the order to add into minimum spanning tree don't change. Since Kruskal's algorithm is proved correct, we can prove the minimum spanning tree doesn't change.

(b)

The shortest paths may change. The example is given below. The original shortest path is a->b->c->d->f. After each edge weight is increased by 1, the new shortest path is a->e->f.





Problem 3:

(a)

We can modify Breadth-First Search to solve this problem. Ignore any edges of weight less than W. The pseudocode is described below.

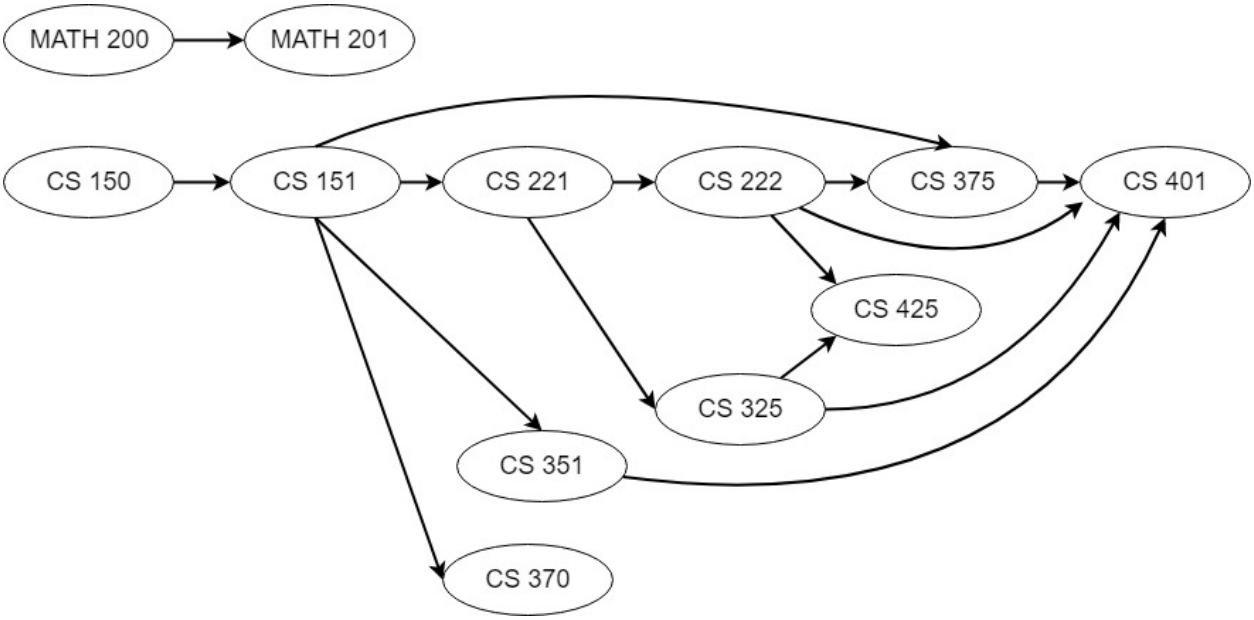
```
BFS(G, s)
        initialize vertices;
        Q = {s};
        while (Q not empty)
                u = DEQUEUE(Q);
                for each v in G.Adj[u]
                        if weight (v, u) >= W
                                if (v == t)
                                        return found
```

```
                    if (v.color == WHITE)
                    v.color = GREY;
                    v.d = u.d + 1;
                    v.p = u;
                    ENQUEUE(Q, v);
        u.color = BLACK;
```

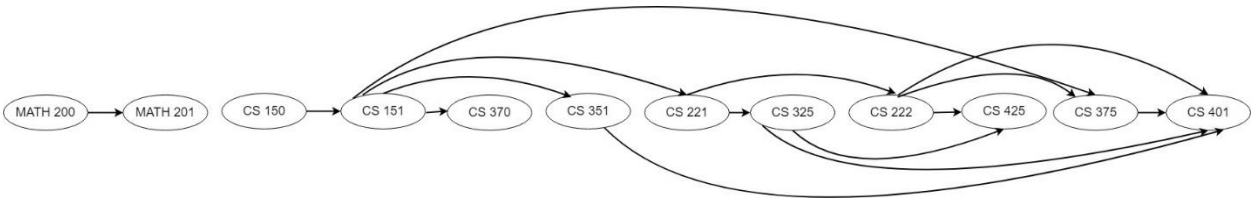
(b)Since we use Breadth-First Search, the running time is still O(V + E).

Problem 4:

(a)



(b)



A topological sort is:

MATH200 -> MATH201 -> CS150 -> CS151 -> CS370 -> CS 351 ->CS221 -> CS325 -> CS222 -> CS425 -> CS375 -> CS401

(c)

Term 1: MATH200, CS150

Term 2: MATH201, CS151

Term 3: CS221, CS351, CS370

Term 4: CS222, CS325

Term 5: CS375, CS425

Term 6: CS401

(d)

The length of the longest path in the DAG is 5. That is CS150, CS151, CS221, CS222, CS375, CS401. We can find the longest path by looking at the DAG I created above. This represents the minimum number of terms (6) required to complete CS degree if we want to meet all the course prerequisite.

Problem 5:

a) It's possible to designate. We can use Breadth-First Search to solve this problem. The pseudocode is below. The code is based on BFS pseudocode from lecture slides.

```
BFS(G, s)                                   //G is the graph and s is the source node
    Initialize vertices
    Q = {s}                                 // Q is a queue initialize to s
    while (Q not empty)
        u = Q.dequeue()                     // Dequeue a vertex from queue
        for each v in G.Adj[u]              // go through the neighbors of u
            if (v.color == WHITE)           // v is unvisited
                ENQUEUE(Q, v)
                v.color = GREY              // mark as visited
                v.d = u.d + 1               // increase distance
                if v.d is even
                    add to team "Babyfaces"
                else
```

add to team “Heels”

if u and v are on the same team

print impossible to designate

exit the program

print team members

b) The running time is O(V + E) because this is the time complexity of Breadth-First Search. Each vertex is enqueued once and dequeued once: O(V). Each adjacency list is traversed once: O(E). Therefore, the running time is O(V + E).