Student: Joaquin Saldana
Assignment: CS325 Summer 2017 / HW5

1. Considered the weighted graph, this is the order:
    a. AE (4), EB (7), BC (13), CD (14), CG (23), GF (30), DH (42) and the total weight is 42.
    b. A, E, B, C, D, F, G, H
        i. A = 0, B = 5, C = 11, D = 12, E = 4, F = 15, G = 15, H = 24
2. Pseudocode to determine if a directed acyclic graph (DAG) has a Hamiltonian path.

    Create a linked list and topologically sort the directed acyclic graph

    ```
    G = []

    for i -> n
        sort G
    ```

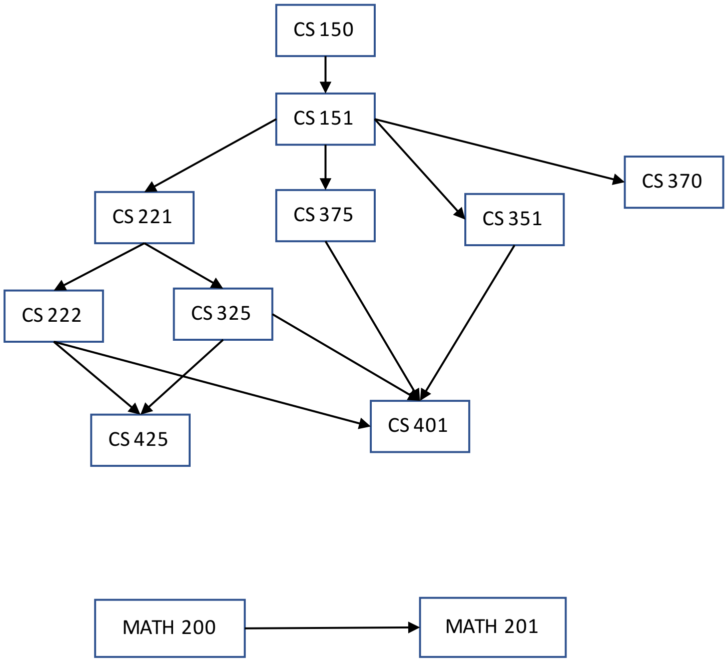
    Once the sort is complete, you know the edges go from lower index vertices to higher index vertices for a Hamiltonian path to exists. In other words: (1,2), (2,3), …., (n-1, n)

    ```
    While(The list is not empty)
        If 1 node w/ 0 child nodes is present in list
                Remove the node and links to in the list
        Else
                Return False

    Return true
    ```

    As a result, the run time should be O(V+E).

3. Below is the list of courses and prerequisites for a factious CS degree:
    a. Below is DAG for the courses listed in the table:



b. Topological sort of the graph:
   i. Q1: MATH 200, CS150
   ii. Q2: MATH 201, CS151
   iii. Q3: CS370, CS351, CS375, CS221
   iv. Q4: CS222, CS325,
   v. Q5: CS425, CS401

c. You can do so w/ the following order:

   (MATH200, CS150), (MATH 201, CS151), (CS221, CS351), (CS222, CS325), (CS375, CS425), and (CS401, CS425)

d. The longest path would be taking one class at a time and using BFS

   MATH200, MATH 201, CS150, CS151, CS370, CS351, CS221, CS222, CS375, CS325, CS425, and CS401

4. Algorithm, w/ a runtime of O(V+E), to determine the 2 coloring of a graph if one exits or terminate w/ the message that the graph is not two colorable.
   a. Below is the pseudocode:

```
G ← Empty list that will contain the colored vertexes

// The first loop coloring the vertices

While vertices uncolored exists
```

```
    Select an uncolored vertex, v
    Color(v)
Return G

// Function that will color the vertices so long as their present in the linked list

function Color(vertex v)
    if v is not colored
            for each vertex j w/ an edge from m to n
                    if m == color 1 and m == color 2
                            return false // graph is not two colorable
                    if m == color 1
                            n = color 2
                    if m == color 2
                            n == color 1
                    if m color is unknown
                            n = color 1
                            color(m)
            append n to G
```

b. The running time should be O(V + E) because most operations are in constant time

5. Fire station question

   a. To find the fastest route from the fire station to each of the intersections should be Dijkstra's algorithm. It would work as follows:

   G has access to all intersections minus A and B, w/ C being the longest route w/ a weight of 9. To get to either A or B it will need to go through various intersections some with heavy weights than others.

   A = 13, B = 6, C = 9, D = 7, E = 2, F = 8, G = 0, and H = 3

   b. Use Dijkstra's algorithm on every vertex and obtain the lowest weight to determine which vertex to use

   c. G might be the best location since it has access to all intersection minus A and B

Extra Credit:

The best locations for the fire stations would be C and H. At C, it can reach intersections A, G, D, and F directly w/ the furthest distance being 6. While with H it can reach stations G, B, and E w/ the longest distance being 7.