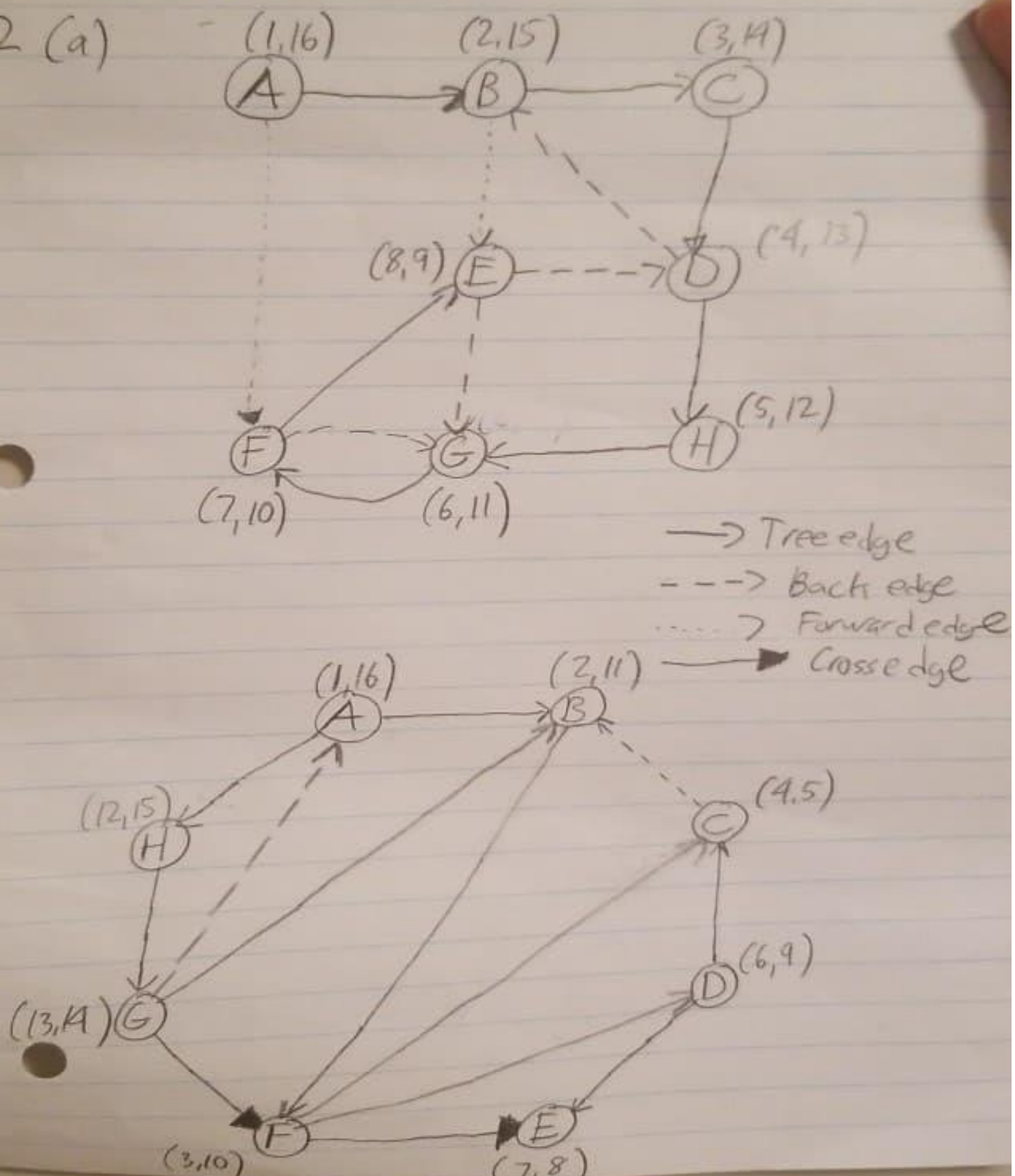
Kevin San Gabriel
301342241
CMPT 307 Assignment 3

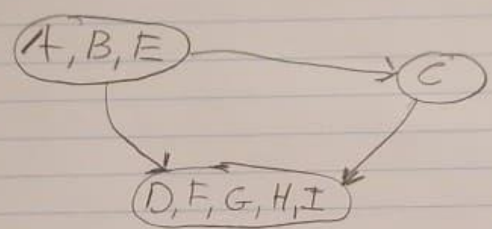
3.2 (a)



c) the metagraph is as depicted previously

d) To make the graph strongly connected, we must add an edge from {E} to {C, D, F, I} and another edge from {E} to {B}. Thus, we need to add a minimum of 2 edges to make the graph strongly connected

ii) a) Similar to the procedure on the graph in i), we can construct a meta graph:



Thus, the order in which we find SCCs is:

{D, F, G, H, I}, {C}, {A, B, E}

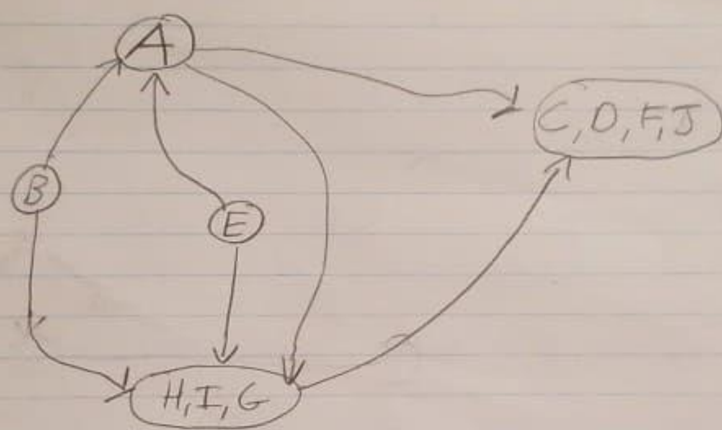
b) Source SCC = {A, B, E} because it is the last explored

Sink SCC = {D, F, G, H, I} because it is the first explored

c) the metagraph is as depicted above

3.4 i) a) After running DFS on the reverse graph of G, we get that the highest post number is 20 on vertex C. Running from vertex C, we ensure we start DFS in a sink SCC.

We can construct the following meta-graph after starting DFS on the next highest post number vertex

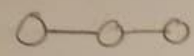


Thus, the order in which the SCCs are found is:

{C,D,F,J}, {H,I,G}, {A}, {E}, {B}

b) Sink SCC = {C, D, F, J} because C has the highest post #

Source SCC = {B} because B has the lowest post #



d) We can make the graph strongly connected if we add an edge from {D, F, G, H, I} to {A, B, E}, thus minimum # edges is 1 to make the graph strongly connected

3.7 a) isBipartite(G):

```
for v in V:
    visited[v] = 0
    if explore(G, v, 1) == 0
        return false
    else
        return true

explore(G, v, color):
    visited[v] = color
    for edge in E:
        if visited[u] == 0 and visited[u] + color == 0.
            return false
        else
            explore(G, u, color * (-1))
```

b) <u>Proof</u> by contradiction

Suppose $G = (V, E)$ has atleast one cycle of odd length and G is bipartite.

Let V_1 be a node in a cycle and V_2 be the last node in that cycle before V_1

If we have 2 colors only, and V_1 is colored with one color, then V_2 must be colored with the same color as V_1

since the cycle is of odd length, Therefore, G cannot be valid for a proper 2-coloring because we have two adjacent vertices V_1 and V_2 that are the same color. This ends our proof by contradiction.

c) We need at most 3 colors because if we have G be a graph that has no odd length cycles, we can create an extension graph of G, namely G' by adding an edge to G. This edge will create an odd length cycle and we need 3 colors for this case.

3.9

We need a function which finds the neighbours of u.

```
find_neighbours (G, u):

    visited[u] = true          v is a neighbour of u
    previsit[u]
    for each edge (u,v) in E:
        twodegree[u] = twodegree[u] + degree[v]
        if not visited[v]
            find_neighbours[v]
    postvisit[u]
```

Firstly, we compute the value of each node by counting the number of entries there are on the adjacency list.

Secondly, we compute twodegree by initializing the degrees of each vertex to 0, then, calling find_neighbour to compute the sum of each neighbour v of each vertex u.

In our algorithm, all nodes are visited to check the neighbours of each vertex. There are n nodes so the loop iterates n times. Thus, we have a running time of O(n).

3.22 Algorithm:

i) run DFS on the graph starting with any node
ii) run DFS again on the graph starting with the highest post number vertex.
iii) if we reach all nodes on the second run of DFS, then yes, there is a vertex $s \in V$ from which all other vertices are reachable, otherwise no there is not.

3.24

```
is_path(G):
    dfs(G)
    add vertices visited by DFS to a list L
    while L not empty:
        remove vertex from list and add to a linearly ordered list
        decrement the indegree of the neighbors of v by 1
        if indegree of any vertex == 0
            add that vertex to L
    for consecutive vertices i, i+1 in the linearly ordered list:
        if there is no edge (i, i+1)
            return false
    return true
```

3.29 statement: binary relation "R" on a set "S"

We need to show that a binary relation R on a set S is reflexive, symmetry and transitive

Assume ^and represent^ elements of S with an edge "x" to "y" if $x, y \in R$

Reflexive

The statement is reflexive if $xRx \ \forall x \in S$.
With our previous assumption, the statement is reflexive because all values of x are in set S

Symmetric

The statement is symmetric if xRy implies yRx
Let us claim that connected components of the graph are the partitions of S namely $S_1, S_2, \ldots S_k$

From our assumption on the statement, the statement is symmetric because $S_1, S_2 \ldots S_k$ are all connected in the same group S on the graph

Transitivity

Our statement is transitive if xRy and yRz implies xRz

we know there are no edges between two components, so, elements from different partitions are not related.

Because the members of different groups are not related, when they are categorized in the same group, we can conclude that our statement is transitive

Because our statement, with the assumption that we assume and represent elements of S with an edge "x" to "y" if x, y ∈ R, has the properties of reflexivity, symmetry and transitivity, we can conclude that the binary relation on S is an equivalence relation when it is disjointed into groups.