Algorithms HW 8 – Dakota Krogmeier

1. A cycle is when a path in a directed graph begins and ends at the same vertex and a directed graph only has a cycle if a DFS search shows a back edge.
   1. Suppose that A has a cycle. Vj is the first vertex found, then we can assume that all other Vu are reachable, meaning they are all descendants in the tree thus the edge, (Vj-1, Vj) is a back edge.
   2. Suppose (u,v) is a back edge. u is a descendant of v in the DFS tree, so there is a path from v to u. Because of this, v is an ancestor of u, so the path from v to u and the back edge (u,v), complete the cycle.
2. Key Idea: The key idea is to keep track of when a vertex is being discovered and when it is being finished. The value placed on each vertex will determine which classification of edge it is, assuming that its not a tree edge.

Algorithm ClassifyEdges(G,G’):

Input: A directed graph, G = (V,E) and a corresponding MST for G,G’ = (V,E)

Output: Lists of edges categorized as one of the following: Tree, Forward, Back, Cross

Process: ClassifyEdges(G,G’):

Def dfsVisit(g,v){

results = 0;

resultsAdd = results + 1;

startTime = resultsAdd;

if results:

then edge is tree edge;

for each vertex n in G(v);

if we don’t visit node;

run dfsVisit;

if start time of v is < start time of n:

edge is forward;

if v finish time > n finish time:

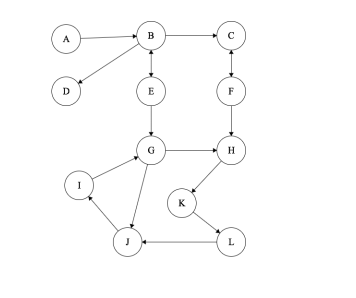
edge is backward;

else;

edge is cross;

3. a. False

b. False

c. 

4. 