CS 2011: Practice Final SOLUTION

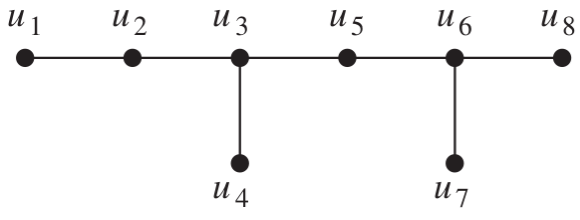
Summer 2018

University of Minnesota

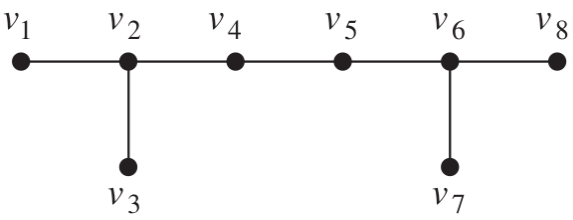
Quiz period: 15 minutes

Points available: 30

Problem 1 (10 pts): Show that the two graphs below **are not isomorphic** to one another. Explain your reasoning.



Graph G



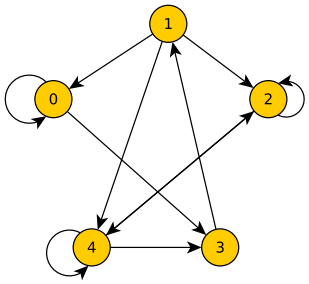
Graph H

SOLUTION: The graphs have an equal number of vertices and edges and their distribution of vertex degrees is identical. However, in G, the path u_1, u_2, u_3 has degree sequence $\{1, 2, 3\}$. The only candidates that start with degree 1 in H are v_1, v_3, v_7, v_8 but no path with that degree sequence starting with these nodes has the sequence desired.. Thus these two graphs are not isomorphic.

Problem 2 (10 pts): The code for Warshall's algorithm to find the transitive closure of a relation is given. Demonstrate this code on the relation which is shown as an adjacency matrix. Show the matrix `Rel` each time line 12 is reached as indicated. For reference, the directed graph version is also shown.

```
 1 bool[][] warshall_tc(bool Rel[][]) {
 2   assert(Rel is a square matrix);
 3   int n = rows(Rel);
 4   bool[][] Rel = copy(Rel);
 5   for(int v=0; v<n; v++){
 6     for(int i=0; i<n; i++){
 7       for(int j=0; j<n; j++){
 8         bool b = Rel[i][v] AND Rel[v][j];
 9         Rel[i][j] = Rel[i][j] OR b;
10       }
11     }
12     // SHOW Rel HERE for v=0,1,2,...
13   }
14 } // Rel is now its transitive closure
```

```
initial Rel
    0 1 2 3 4
  +----------
0 | 1 0 0 1 0
1 | 1 0 1 0 1
2 | 0 0 1 0 1
3 | 0 1 0 0 0
4 | 0 0 1 1 1
```



SOLUTION: Iterations are shown below.

```
v=0 done
    0 1 2 3 4
  +----------
0 | 1 0 0 1 0
1 | 1 0 1 1 1
2 | 0 0 1 0 1
3 | 0 1 0 0 0
4 | 0 0 1 1 1
```

```
v=1 done
    0 1 2 3 4
  +----------
0 | 1 0 0 1 0
1 | 1 0 1 1 1
2 | 0 0 1 0 1
3 | 0 1 1 1 1
4 | 0 0 1 1 1
```

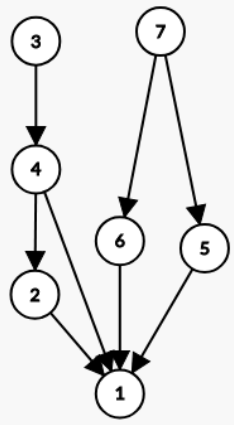
```
v=2 done
    0 1 2 3 4
  +----------
0 | 1 0 0 1 0
1 | 1 0 1 1 1
2 | 0 0 1 0 1
3 | 0 1 1 1 1
4 | 0 0 1 1 1
```

```
v=3 done
    0 1 2 3 4
  +----------
0 | 1 1 1 1 1
1 | 1 1 1 1 1
2 | 0 0 1 0 1
3 | 1 1 1 1 1
4 | 1 1 1 1 1
```

```
v=4 done
    0 1 2 3 4
  +----------
0 | 1 1 1 1 1
1 | 1 1 1 1 1
2 | 1 1 1 1 1
3 | 1 1 1 1 1
4 | 1 1 1 1 1
```

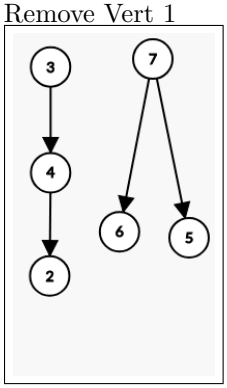
Problem 3 (10 pts): Below is naive code which produces a topological sort of a directed acyclic graph (DAG). Show diagrams of how the code transforms the given DAG at each iteration by drawing its state line 8 of the code. Show your resulting topological sort of the vertices.

```
 1 vertex[] topo_sort(graph G){
 2   int n = number_of_vertices(G);
 3   vertex order[] = new vertex[n];
 4   for(i=n-1; i>=0; i--){
 5     vmin = find a "minimal" vertex in G;
 6     order[i] = vmin;
 7     G = remove vmin from G;
 8     // SHOW GRAPH HERE
 9   }
10   return order;
11 }
```

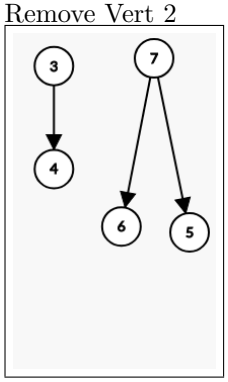


SOLUTION: Topological order {7,3,4,6,5,2,1}; a variety of other solutions possible.

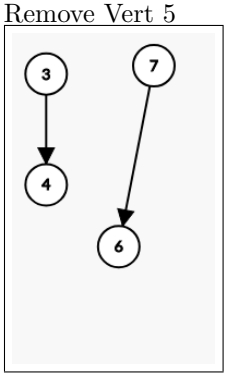
Remove Vert 1



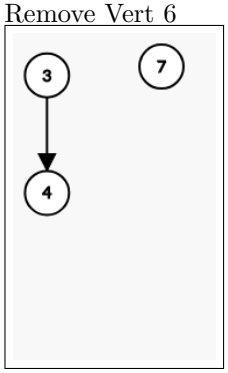
Remove Vert 2



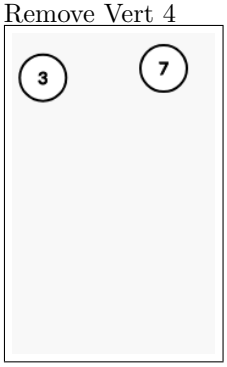
Remove Vert 5



Remove Vert 6



Remove Vert 4



Remove Vert 3