

# **CS 340 Fall 2023 Project 1**

**Due 11:30 PM Friday September 15**

# Project 1

Implement the methods in the TopologicalSort class shown on on the following slides. The class implements an adjacency list representation of a directed graph and the topological sort algorithm. The vertices in the adjacency list representation are stored in a singly linked list **with a** sentinel node. The edges for each vertex are stored in singly linked lists **without a** sentinel node. Part of the purpose of this assignment is to give you practice with lists and that is why I am requiring the two different list implementations.

The list of vertices is stored in **ascending order by name**.

The lists of edges are stored in **ascending order using the name of the second vertex** in the edge.

You can add private methods and instance variables. For each private method include a comment explaining what the method does and for each private instance variable include a comment explaining the value stored in the variable.

At the top of the java file put a comment with your name in it.

In this project you must not use any Java list implementations to implement the vertex list or the edge lists. You must implement the vertex list and the edge lists using the vertex node and edge node classes shown on the following slides. You can use a Java class for the FIFO Queue needed in the topoSort.

# Project 1

```
import java.io.*;
import java.util.*;

public class TopologicalSort {

    private class VertexNode {
        private String name;
        private VertexNode nextV; //reference to the next vertex
        private EdgeNode edges; //head of the edge list for the vertex
                                //the edge lists are sorted in ascending order
                                //based of the name of the second vertex in the edge

        private int indegree;

        private VertexNode(String n, VertexNode v) {

            name = n;
            nextV = v;
            edges = null;
            indegree = 0;
        }
    }
}
```

# Project 1

```
private class EdgeNode {
    private VertexNode vertex1; //the edge (vertex1, vertex2)
    private VertexNode vertex2;
    private EdgeNode nextE; //reference to the next edge in the edge list

    private EdgeNode(VertexNode v1, VertexNode v2, EdgeNode e) {
        vertex1 = v1;
        vertex2 = v2;
        nextE = e;
    }
}

private VertexNode vertices; //head of the list of vertices
                                //the list of vertices is sorted
                                // in ascending order based on the name

private int numVertices;

public TopologicalSort() {
    vertices = new VertexNode(null, null); //sentinel node
    numVertices = 0;
}
```

# Project 1

```
public void addVertex(String s) {  
    //insert a new vertex so the vertex list remains sorted in ascending order  
  
}  
  
public void addEdge(String n1, String n2) {  
    //PRE: the vertices n1 and n2 have already be added  
    //insert the new edge so the edge list remains sorted in ascending order  
  
}  
  
public void printGraph() {  
    //print the graph in the format shown in class  
  
}  
  
public String topoSort() {  
    //return a string with the vertex names in a topological order  
    //if no topological order exists return the empty string  
  
}
```

# Project 1

```
public static void main(String args[]) throws IOException{
    //Simple Test Driver
    //This driver assumes the input file is formatted correctly
    BufferedReader b = new BufferedReader(new FileReader(args[0]));
    TopologicalSort g = new TopologicalSort();
    String line = b.readLine();
    Scanner scan = new Scanner(line);
    while (scan.hasNext()) {
        g.addVertex(scan.next());
    }
    line = b.readLine();
    while (line != null) {
        scan = new Scanner(line);
        g.addEdge(scan.next(), scan.next());
        line = b.readLine();
    }
    g.printGraph();
    System.out.println("\nTopological Order\n"+g.topoSort());
}
```

# Project 1

## Example 1

### Input File

```
G F E D C B A
A F
B F
B G
C E
C D
C B
C A
D A
E B
E D
F G
```

### Program Output

Vertex	Adjacent Vertices
A	F
B	F G
C	A B D E
D	A
E	B D
F	G
G	

Topological Order  
C E B D A F G

# Project 1

## Example 2

### Input File

```
cs120 cs225 cs220 cs270 cs340 cs202 cs364 cs341
cs340 cs341
cs220 cs364
cs120 cs202
cs220 cs340
cs120 cs225
cs225 cs340
cs225 cs270
cs120 cs220
```

### Program Output

Vertex	Adjacent Vertices
cs120	cs202 cs220 cs225
cs202	
cs220	cs340 cs364
cs225	cs270 cs340
cs270	
cs340	cs341
cs341	
cs364	

### Topological Order

```
cs120 cs202 cs220 cs225 cs364 cs270 cs340 cs341
```



# Programming Project 1 Submission

- Upload one zip file to Canvas. The zip file must contain **only one file called TopologicalSort.java**. Do not upload your whole Eclipse project!
- I have included a simple test driver to help you test your code. I could use a different test driver.
- **This project must be submitted on time.**