CSCE 340 DESIGN AND ANALYIS OF ALGORITHMS
LONGEST PATHS IN A DAG

DR. GODFREY MUGANDA

1. The Project

The goal of this project is to devise and implement an algorithm for finding the longest path through a DAG that starts at a given vertex.

Similar to the first homework project, you will have recursive and non-recursive versions of DFS that solve this problem. Starter code that reads an input file, builds an adjacency list for the graph, and calls the appropriate methods the code that you write will be provided to you. The code in `main.java` must not be modified.

2. Sample Run

```
Is there another graph to process? (Y/N)y

Vertex name to integer map:
{a=0, b=1, c=3, d=2, e=4, f=6, g=5}
Vertex integer to string names map:
[a, b, d, c, e, g, f]

Adjacency list for the graph in string names form:
a : [b, d]
b : [c]
d : [g]
c : [d, e]
e : [f]
g : []
f : [d, g]

Non-recursive Longest paths from each vertex:
a b c e f d g
b c e f d g
d g
c e f d g
e f d g
g
f d g

Recursive Longest paths from each vertex:
a b c e f d g
b c e f d g
d g
```

```
c e f d g
e f d g
g
f d g

Is there another graph to process? (Y/N)
```

3. Strategy

The key is to use two arrays

```
static int [ ] dist;
static int [ ] nextHop;
```

where for each vertex v, the array entry `dist[v]` is the number of vertices on the longest path that starts at v, and `nextHop[v]` is the vertex immediately after v on that longest path.

Initialize each `dist[v]` to 1, and each `nextHop[v]` to -1 to indicate "no vertex."

Once this is done, modify the `search(v)` method of your DFS and RecursiveDFS class to begin a DFS search at v and store relevant information in the `dist[ ]` and `nextHop[ ]` arrays. Finally, add a three methods to your DFS classes:

```
public static void search()
{
  // Fill in:
  // Loop through all vertices v starting a search
  // at each v that has not yet been visited.
}

// Prints the longest path starting at each
// vertex in the graph g.
public static void printLongestPaths(Graph g)
{
    for (int v = 0; v < adjList.length; v++)
    {
        printLongestPath(g, v);
        System.out.println();
    }
    System.out.println();
}

public static void printLongestPath(Graph g, int v)
{
    // Fill in:
    // print the longest path that starts at v
    // using the information stored in nextHop[ ]
}
```

Once you get these three methods `search(v), search()`, and `printLongestPath(g, v)` working correctly, you will be done.

4. Due Date

February 20, 2020.