# **Assignment 5 - Depth-First Search**

### Depth-First Search

Section 14.4 of the course text, "How to Think About Algorithms", describes an iterative algorithm that executes a depth-first search on a graph. Section 14.5 presents a recursive implementation that is equivalent to the iterative solution. For this assignment, implement either one of the two algorithms.

### Implementation Details

Given an undirected graph G, and a starting node s, for each path encountered follows the path as deeply as possible into the graph before backtracking. The nodes in that collection form a tree of nodes in the graph G reachable from s.

Define a Java class named GraphSearch. The method that implements the depth-first search algorithm should look like this:

public static Set<Vertex> depthFirstSearch( Graph g, Vertex s ) { ... }

Do not include a public static void main method in class GraphSearch.

The algorithm returns a reference to the set foundHandled. Recall that Java  provides a class, LinkedHashSet that keeps an internal linked list that references the elements of the set in the order that they were inserted (Gaddis, chapter 19.3). This this particular type of set allows elements to be accessed in the order they were inserted. Use the Set interface and the LinkedHashSet class to implement and access the set foundHandled. That will enable test programs to access the nodes in the order they were inserted.

I will be testing your programs with a set of my own graphs.

#### **Time-Stamping and Classification of Edges**

The algorithm described in the text includes time-stamping. Feel free to omit time-stamping and edge classification from your implementation.

### What to Turn In

When you are finished, upload the Java file (named GraphSearch.java) to Canvas.