Tony Diep
CS 2420
3-30-17
Design Document

Design Document: Graphs and Pac-Man

The purpose of this assignment is to familiarize ourselves with the graph data structure and its applications to one of the widely popular childhood games: Pac-Man. We are asked to build a program that finds the shortest path in which Pac-Man can travel through a maze at a given time. However, the assignment is a slight variation of the Pac-Man game in that there are no ghosts nor the berries and power-ups. Rather, we primarily focus on Pac-Man, the dots, and start and goal indicators.

This assignment is a free-form assignment, meaning that we are free to create as many classes, methods, and fields as we desire. However, the assignment requires two implementations: 1) implement a graph somewhere for using key operations such as performing breadth first search (BFS), and 2) we must implement a PathFinder class which contains the method “solveMaze” (which is the basis of taking an input maze text file and saving the solved version onto the user’s desktop). Because of the assignment being free-form, we have strategies to program a sufficient and efficient maze path solver.

Most of the assignment specifications are direct giveaways as to how to implement the code. For example, the pseudocode for BFS is really straightforward in that we can simply translate it into Java code (the pseudocode of which was in lecture). Additionally, in the section that says “Project Structure,” reading the paragraph under that section also gives excellent starting points. Another part of the assignment specifications advised to start by creating a Node class and a Graph class to simplify the process, which is more of a requirement for us when creating the program. Sequentially, we construct the Node class first, then the Graph class, then the PathFinder class, and finally, the PathFinderTester which contains the main method and a couple of tester methods that help ensure working code in our program.

The following is an outline that shows our plans for doing this assignment:

- Create a Graph that will take in a file name of the maze text file
- Create an adjacency matrix of chars that will hold each individual character in the maze text file
- Create an adjacency list of Nodes, where each Node corresponds to a non ‘X’ character in the maze text file
    - Have a node that saves where the “start” indicator is on the maze
    - Have another node that saves where the “goal” indicator is on the maze
- Perform breadth first search from our adjacency matrix and adjacency list to find the paths
    - Store the shortest path sequence in a Node array to use later when solving the maze
- Output the solved maze (from the original maze text file) into the user’s desktop