4/16/08 **Lab 12 Graphs** Name:_____

Objectives: To understand how a graph can be represented and traversed.

Activity 1: The text's version of an adjacency list implementation of a graph is available by copying the folder P:\810-063-CSIII\LABS\LAB_12. Open and run the word ladder program in the word_ladder.py file in IDLE. This program tranforms one word into another by changing one letter at a time, e.g., transform FOOL into SAGE by FOOL → FOIL → FAIL → FALL → PALL → PALE → SALE → SAGE by using a breadth-first search (bfs).

Study the output and the code to answer the following questions:

a) In the `buildGraph()` function contained in word_ladder.py, how do the vertices get added to the graph?

b) How do we get the reverse trace of the word ladder transformation of FOOL into SAGE printed?

c) The Vertex class has a number of data attributes (color, dist, pred, disc, fin, and cost) that may or may not get used by any particular graph algorithm, and what if you needed additional data attributes? Suggest a more general way of handling this problem.

After you have answered the above questions, raise your hand and explain your answers.

Activity 2: An alternative representation of a graph is an *adjacency matrix* (section 6.3.1 of the text) which uses a two-dimensional matrix/list to store information about each edge (e.g., its weight). Implement an alternative Graph class using an adjacency matrix representation. Then, use your new graph class in the word ladder application.

After you have tested your code, raise your hand and demonstrate that it works correctly.

NOTE: If you complete all of the activities within the lab period, you do not need to hand anything it. However, if you need to finish activities outside of the lab period, then hand in printouts of code and output for activities not completed during the lab period.