**EE422C Project3 (Word Ladder) Test Plan**

<Kiptoo Tonui> <kt23347>

Summer 2017

**Test Plan Summary**

Used JUNIT for multiple test cases and tested all at once, and focused more effort on fixing bugs that the test cases failed. Covered DFS and BFS with words that do and do not have a rung ladder between them. Also tested both methods for PrintLadder with words that do not have rung ladders between them.

1. basicTestBFS
   1. Checks for correct printing for a pair of words
   2. Set up for the test – initialization
   3. A <n> rung ladder exists between XXX and YYY.
   4. Ladder had no duplicates and no stack overflow
2. noWordLadderBFS
   1. Checks to make sure no word ladder exists between a pair of words
   2. Set up for test – initialization
   3. no word ladder exists between XXX and YYY
   4. Ladder does not exist and time to search is minimized.
3. timeTestBFS
   1. Checks for correct printing for a pair of words and make sure the time it takes to look for the word is minimized
   2. Set up for the test – initialization
   3. A <n> rung ladder exists between XXX and YYY, and the test took less than 3 seconds
   4. Ladder had no duplicates, no stack overflow, and time was efficient.
4. ResetTestBFS
   1. Made sure the variables that were used were reset properly
   2. Set up for test – initialization
   3. That the method still had the correct output and the variables were properly reset within a time efficient matter
   4. Ladder had no duplicates, no stack overflow, and reset was successful
   5. Encountered a problem where the tests would fail since the previous tests still had some variables. Found the issue within Nodes as I was not setting the parent back to null after a search.
5. printLadderBFS
   1. Ensured printLadder function worked properly.
   2. Set up for test – initialization
   3. Program printed the ladder with the correct number and made sure all words were within a one letter difference.
   4. Ladder had no duplicates, no stack overflow, and words were within one letter difference with each other.
6. basicTestDFS
   1. Checks for correct printing for a pair of words
   2. Set up for the test – initialization
   3. A <n> rung ladder exists between XXX and YYY.
   4. Ladder had no duplicates and no stack overflow
7. noWordLadderDFS
   1. Checks to make sure no word ladder exists between a pair of words
   2. Set up for test – initialization
   3. no word ladder exists between XXX and YYY
   4. Ladder does not exist and time to search is minimized.
8. timeTestDFS
   1. Checks for correct printing for a pair of words and make sure the time it takes to look for the word is minimized
   2. Set up for the test – initialization
   3. A <n> rung ladder exists between XXX and YYY, and the test took less than 3 seconds
   4. Ladder had no duplicates, no stack overflow, and time was efficient.
9. ResetTestDFS
   1. Made sure the variables that were used were reset properly
   2. Set up for test – initialization
   3. That the method still had the correct output and the variables were properly reset within a time efficient matter
   4. Ladder had no duplicates, no stack overflow, and reset was successful
   5. Encountered a problem where the tests would fail since the previous tests still had some variables. Found the issue within Nodes as I was not setting the parent back to null after a search.
10. stackOverFlowDFS
    1. Ensure that DFS did not cause a stack overflow.
    2. Set up for test – initialization
    3. DFS did not cause a stack overflow and printed the correct sequence of rung ladders without duplicates.
    4. Ladder did not cause a stack overflow and duplicates were not printed.
11. printLadderDFS
    1. Ensured printLadder function worked properly.
    2. Set up for test – initialization
    3. Program printed the ladder with the correct number and made sure all words were within a one letter difference.
    4. Ladder had no duplicates, no stack overflow, and words were within one letter difference with each other.