Test Plan BFS

1. No duplicates in input, no duplicates in word ladder, length of ladder (shortest), not contained in dictionary, length of word, and no ladder exists between two words in dictionary

2.

1. In a dictionary set we create, we will choose two duplicate words.
2. We will create a duplicate test HashSet for each ladder we build to check for duplicates in ladder returned.
3. In a dictionary set we create, we will choose two words that are one length apart
4. In a dictionary set we create, we will choose two words that are not contained in dictionary
5. In a dictionary set we create, we will choose two words that are less than or greater than 5 letters long
6. In a dictionary set we create, we will choose one word that doesn’t convert into another word in the dictionary by swapping one letter.

3. After setting up the situation and calling getWordLadderBFS, we will call an assertEquals method to confirm each test case mentioned above. (Too specific to describe each test case in full.)

1. assertEquals(null,getWordLadderBFS(“aahed”,”aahed”);
2. assertEquals(false,duplicateTestSet.add(wordInLadder));
3. assert(null != getWordLadderBFS(“yogis”,”yogin”);
4. assertEquals(null,getWordLadderBFS(“aaaaa”,”bbbbb”);
5. assertEquals(null,getWordLadderBFS(“abadfagag”,”adg;”);
6. assertEquals(null,getWordLadderBFS(“aahed”,”zuzim”);

4. Expected output is referenced in assertions above.

5. Keyboard input handling prevents exceptions by identifying bad input. No exceptions are expected to occur unless there is some fatal programming error that we have missed in the BFS or DFS implementations.

Test Plan DFS

1. No duplicates in input, no duplicates in word ladder, length of ladder (no shorter than BFS), not contained in dictionary, length of word, agrees with BFS (whether ladder exists or not)

2.

1. In a dictionary set we create, we will choose two duplicate words.
2. We will create a duplicate test HashSet for each ladder we build to check for duplicates in ladder returned.
3. In a dictionary set we create, we will choose two words with a known ladder from BFS.
4. In a dictionary set we create, we will choose two words that are not contained in dictionary
5. In a dictionary set we create, we will choose two words that are less than or greater than 5 letters long
6. In a dictionary set we create, we will choose one word that doesn’t convert into another word in the dictionary by swapping one letter according to BFS results.

3. After setting up the situation and calling getWordLadderDFS, we will call an assertEquals method to confirm each test case mentioned above. (Too specific to describe each test case in full.)

1. assertEquals(null,getWordLadderDFS(“aahed”,”aahed”);
2. assertEquals(false,duplicateTestSet.add(wordInLadder));
3. assert(null != getWordLadderDFS(“yogis”,”yogin”);
4. assertEquals(null,getWordLadderDFS(“aaaaa”,”bbbbb”);
5. assertEquals(null,getWordLadderDFS(“abadfagag”,”adg;”);
6. assertEquals(null,getWordLadderDFS(“aahed”,”zuzim”);

4. Expected output is referenced in assertions above.

5. Keyboard input handling prevents exceptions by identifying bad input. No exceptions are expected to occur unless there is some fatal programming error that we have missed in the BFS or DFS implementations.

Team Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **7/6/16** | Spencer | Joseph | Both |
| Done | getWordLadderBFS, keyboard input handling |  | Test Plan, Team Plan |
| Plan to Do | Junit for BFS portion | getWordLadderDFS, Junit for DFS portion |  |