Testing Outline, Christopher Friesen cmf2536 Malvika Gupta mg42972

Functional Black-Box testing is a set of tests that observe the functionality of a program without knowledge of the inner workings of the program. For our program, the overall functionality will be if it can correctly verify that a word ladder exists between two words with our dictionary “A3words” which contains 5757 five-lettered words. Therefor our black box testing will consist of running two words into the solver that we know do or do not have a word ladder. Also the input class will not allow invalid words to be passed to the solver, so our test will consist of dictionary correct words utilizing the JTest with the method testComputeLadder(). Here is the list of words we will test to observe this functionality:

Words	Ladder Exists?
“dears“ and “fears ”	True
“stone” and “money”	True
“mumbo” and “ghost”	False
“hello” and “buddy”	True
“hello” and “world”	False
“heads” and “tails”	True
“atlas” and “zebra”	False

Complete branch coverage white box testing is a set of tests that analyze every path of data flow through a program. This is done using knowledge of the system to make sure every part of the code is run. We will accomplish this by using the JTest methods testgetLadderWords() and testDictionary() in conjunction with our previous computeLadder() test. getLadderWords will test the code that gets and reads the input file. The dictionary test will simply test the functionality of the search and valid line functions. The black box test already covers every branch in the solver algorithm. Our tests will be as follows:

Test LadderWords()	Expected result	Test Dictionary()	Expected result
“ “	Empty exception	123678	Will not be added to the dictionary, cannot be found within the dictionary
“stone money atone”	Too many words exception	“ “ (empty)	^^^
“ryan joeeeey”	Words too long/short exception	Blame	^^^
“stone stone”	Words same exception	!@#$@	^^^
“abcde 12345”	Words not in dictionary exception	blame	Will be added to the dictionary and can be added to it
“stone atone”	“stone”, “atone”		
<no next line>	Return null		