A4 Analysis – CS3505

Elijah Grubb && Tanner Barlow

AddWord_IsWord:

Note: since asWord and isWord are so intertwined in the API in ways we can test them, we combined their tests together to test both

OneWord

- This test adds a single word to our trie using addWord and tests that before it is added, isWord is false and that after it's added, isWord is true.

EmptyString

- This test makes sure that using isWord on an empty trie with an empty string returns false since there is nothing in the string

MultipleWords

- This test adds several words to our trie and makes sure that isWord returns true on all of them as a basic test that our trie works with many words.

AlmostWords

- This test checks edge cases against a word we've added to our trie, "basketball" and makes sure that they don't accidently return true. Adding or deleting a character from the end of the word should return false.

DifferentEnd

- This test adds a bunch of words to a trie instance and makes sure that simply changing the last character of the word won't trick our trie into thinking that word exists. So if "basketball" is a word, isWord("basketbalk") should return false but isWord("basketball") should return true.

RefCount

Note: this also tests addWord and isWord at a deeper level by ensuring that the nodes are in fact being created.

StressTest

- This test makes sure that the reference count correctly ramps up as you add words to a trie and then goes back to zero once you've deleted it. So the word "basketball" has a refcount of 11, one for each node containing each letter and an extra one for the root of our trie. Then once we've deleted it we make sure that the refCount has gone back to 0.

Prefix

PrefixCount

- This test makes sure the right number of wordWithPrefix are being returned. Since 3 words in our trie have the prefix “b” then the size of the returned vector should be 3.

PrefixWords

- This test individually checks the returned vector of a allWordsWithPrefix call and makes sure that all the expected words are contained in that vector.

SameWord

- This test ensures that when calling allWordsWithPrefix, if the input string is a word in our trie then that word should be returned with inside of the output vector.

AssignmentOperator

DISABLED_Stack

- This test checks that our assignment operator works on tries built on the stack, not with the new keyword.

DISABLED_Heap

- This test checks that our assignment operator works on tries located on the heap, created with the new keyword.

CopyConstructor

DISABLED_Parentheses

- This test checks that the copy constructor works when called with the new keyword with the old trie passed in. Ex: Trie b = new Trie(a);

DISABLED_Equals

- This test checks that our copy constructor works when called by creating a new trie that equals an old one. Ex: Trie b = a;