1. Notable Obstacles

My most notable obstacle was correctly finding all the permutations of the word in recBlends and putting all of the matches of its permutations to the strings in dict and putting each match into the results array with the correct number of matches inserted into results.

1. Test Cases

loadWords:

// really really big files (thousands to millions of lines of words)

// really really small files (very few lines of words)

// empty files (no words at all)

// files with only 1 word repeated multiple times

recBlends:

// “Book” // words with more than one of the same letter

// words that have repeated permutations

// really big words i.e. intermediate, immediately, etc.

// empty words

// strings that use only one letter as characters

// strings that use only two letter as characters

// strings that use only three letter as characters

// strings that use only four letter as characters

…

showResults:

// empty results array

// non-empty but not full results array

// full results array

Permute("", "book");

*// cout << endl;*

string dict[25143];

ifstream dictfile;

dictfile.open("words.txt");

loadWords(dictfile, dict);

string results[MAXRESULTS];

string iron[24];

ifstream testfile1;

testfile1.open("testfile1.txt");

loadWords(testfile1, iron);

cout << endl;

recBlends("book", iron, 24, results);

showResults(results, MAXRESULTS);

dictfile.close();

testfile1.close();

**return** 0;

}

**int** loadWords(istream &dictfile, string dict[])

{

string line;

**if** (getline(dictfile, line)) *// read in each line until we are out of lines*

{

dict[0] = line;

**return** 1 + loadWords(dictfile, dict + 1);

}

**return** 0;