Project 3 report

Writer: Xinyi Li

Professor: E. Ambrosio

Date: August 1, 2020

Obstacles:

In this project, I have overcome a few obstacles.

1. The function characterMixer function takes time to think. My algorithm is to compare each word in the dictionary with the target word. I used a helper function to check if a word is a permutation of the input word. The helper function word compare return true if this word is a permutation of the target word. Using this helper function in the characterMixer function, if the word is a permutation, then add it to the result array.
2. The helper function is the main obstacle to overcome. It checks if two words are permutation to each other. Therefore, each iteration, if the first character of the target word is found in the dictionary word, throw this character away in both of the words. Feed these two substrings into the next helper function until two words that were input are equal ( return true) or some characters are not found in the dictionary word (return false).

Testing:

All my testing cases are satisfied. The testing main function is shown below:

**int** main()

{

string results[MAXRESULTS];

string dict[MAXDICTWORDS];

ifstream dictfile; // file containing the list of words

**int** nwords; // number of words read from dictionary

string word;

dictfile.open("words.txt");

**if** (!dictfile) {

cout << "File not found!" << endl;

**return** (1);

}

nwords = createDict(dictfile, dict);

cout << "count check " << nwords << endl;

cout << dict[nwords-1] << endl;

string str = "abcdefg";

**int** i = 0;

string sub = str.substr(0,i) + str.substr(i+1, str.size()-1);// get rid of a;

cout << sub << endl;

cout << i+1 << endl;

//viewAnswers(dict, 8);

cout << "Please enter a string for an anagram: ";

cin >> word;

**int** numMatches = characterMixer(word, dict, nwords, results);

**if** (!numMatches)

cout << "No matches found" << endl;

**else**

viewAnswers(results, numMatches);

**return** 0;

}

The output of inputting rat is:

**count check 25144**

**zygote**

**bcdefg**

**1**

**Please enter a string for an anagram:** rat

**Matching word art**

**Matching word rat**

**Matching word tar**

**Program ended with exit code: 0**

The output of inputting regardless is:

**count check 25144**

**zygote**

**bcdefg**

**1**

**Please enter a string for an anagram:** regardless

**No matches found**

**Program ended with exit code: 0**

The output of inputting adventitious is:

**count check 25144**

**zygote**

**bcdefg**

**1**

**Please enter a string for an anagram:** adventitious

**Matching word adventitious**

**Program ended with exit code: 0**

The output of inputting eat is:

**count check 25144**

**zygote**

**bcdefg**

**1**

**Please enter a string for an anagram:** eat

**Matching word ate**

**Matching word eat**

**Matching word eta**

**Matching word tea**

**Program ended with exit code: 0**

The output of inputting ant is:

**count check 25144**

**zygote**

**bcdefg**

**1**

**Please enter a string for an anagram:** ant

**Matching word ant**

**Matching word tan**

**Program ended with exit code: 0**