Overview:

Press (1) to add a word to the dictionary

- The dictionary is held in the file called "dict.txt"

Press (2) to use the spellcheck function

- If the string is spelled correctly, the program will notify you; if not, a new word will be recommended

Press (3) to quit the program

- All words added to the dictionary will be saved

Details

Data structure: Trie

- Used to store all of the correct words from the dictionary file
- Chose over BK Trees, Hash Tables, and Binary Trees because Tries offered the best time complexity
- Time complexity: O(NM), for creating a Trie, where N represents the number of nodes and M is the length of the string being inserted.
- Space complexity: relatively high

How it works

Adding a word to the dictionary (1)

- Simply appends the inputed string to the end of the text file

Spellcheck function (2)

- First checks to see if the word entered exists in the trie. If it does, the program will inform the user that they have entered a correctly spelled word. If not, the program will execute below
- Utilizes the Levenshtein Distance algorithm
  - Works by first comparing each string in the trie to the word entered, then computes the similarity score for each word pair, and finally stores that value in a matrix
    - Insertions/deletions of a character is counted as +1
    - Substitutions of a character is counted as +2
      - Ex: similarity score of “mat” and “max” is 1
  - Once every string has been computed, the program returns the word associated with the similarity score stored in the bottom right side of the matrix. This will be the most similar word