Project Report

Approach used in Project

- We created a Dictionary class containing an unordered_map of all the words from dictionary2.txt and the string from dictionary1.txt. We setup functions in the Dictionary class which test string takes as an input and returns number of characters it could match from the test string with any word from dictionary2.txt or partial string from dictionary1.txt.
- Our main class takes number of keys and a cipher text as an input.
- We created Dictionary class, so that it could be reused by any de-cipher algorithm to test if it found a plaintext or not.
- It has these member variables and methods-
    - hashMap to store words from dictionary2
    - initHashMap() to initialize the dictionary
    - isTestTextInDictionary() to check if testText is in Dictionary.
    - getMaxLenWords() returns max length of words in dictionary
    - getMinLenWords() returns min length of words in dictionary
- This main method checks the number of keys:
- if it is equal to 1, we know that the ciphertext is encrypted using simple shift cipher.
    - We created new Class called ShiftCipher. In general, we are planning to use new class for each type of new algorithm we would like to test our cipher text against.
    - ShiftCipher has one member function - getShiftCipherText(), which takes ciphertext and shiftKeyPosition as an input and returns a ciphertext string shifted by shiftKeyPositions
    - For each k = 0 to k = 25, we run getShiftCipherText() with ciphertext as an input. The returned string is provided as an input to Dictionary object's isTestTextInDictionary()
    - if isTestTextInDictionary() returns >0, it means, we have found our key and the string from getShiftCipherText() is our plain text.
- If numKeys == 26, we know that this could be mono alphabetic substitution cipher where each alphabet is shifted by a fixed unique key between 1 and 26.
    - To test our ciphertext for numKeys == 26, we created new class called MonoAlphaSubstituteCipher which contains the following member variables and methods -
        - getMonoAlphaSubstituteCipher() returns a string where each alphabet is shifted by corresponding key from key array.
        - permuteKeys() generates new set of keys in key array
    - Our main method runs permuteKeys() to generate new set of keys and then runs getMonoAlphaSubstituteCipher() to get testtext when applied mono alphabetic substituition with the keys generated previously by permuteKeys().

The returned string is tested against dictionary by calling Dictionary object's isTestTextInDictionary()
    - if isTestTextInDictionary() returns >0, it means, we have found our key and the string from getMonoAlphaSubstituteCipher() is our plain text.
    - if isTestTextInDictionary() returns 0, repeat the process with new set of keys generated by permuteKeys()
- If numKeys is not equal to 1 or 26, our job becomes more difficult.
    - We could try taking a keyphrase from combining words from english dictionary. For this program, we are considering using combination of words from Dictionary2.txt, so that the combined string's length is equal to numKeys. e.g. if the number of key entered as input is 10, we could try passkey phrase as legalfurry combining two english words - legal and fury from dictionary2.txt
    - For this, we created new class called PolyAlphaSubstituteCipher which contains the following member variables and methods -
        - getPolyAlphaSubstituteCipher() returns a string where each alphabet is shifted by repeating keyphrase
        - permuteKeyPhrase() generates new keyphrases
    - Our main method runs generateKeyPhrase() to generate a new pass keyphrase and then runs getPolyAlphaSubstituteCipher() to get testtext when applied poly substitution with the key passphrse generated previously by generateKeyPhrase(). The returned string is tested against dictionary by calling Dictionary object's isTestTextInDictionary().
    - If isTestTextInDictionary() returns >0, it means, we have found our pass keyphrase and the string from getPolyAlphaSubstituteCipher() is our plain text.
    - If isTestTextInDictionary() returns 0, generate new pass keyphrase and repeat the above steps.