Project Description

In this miniproject, you are expected to implement a brute force password cracker based on the Vigenere Cipher you implemented in mini-project 1.

Your password cracker is expected to take three parameters: (1) a string of ciphertext; (2) an integer keyLength that denotes the length of the key; and (3) an integer firstWordLength that denotes the length of the first word of the plaintext.

Your password cracker will test every possible key that has the length of keyLength: from all "A"s to all "Z"s. You cannot exploit the dictionary to guess the key, since the key may not be a valid word.

For each key candidate, you will generate a "plaintext", and compare it with the dictionary. In particular, you only need to check if the first word (number of letters of the word is given in firstWordLength) is a valid word in the dictionary. To do this, you need to load the dictionary into memory before processing any key, and search if the first word of the "plaintext" is in the dictionary. If Yes, display the plaintext and the key. However, do not stop, as the "plaintext" might be wrong.

Efficiency is very important in evaluating each "plaintext" candidate.

In some cases, a wrong key may generate a valid first word. Hence, you may get several "plaintexts" after all possible keys are tested. This is acceptable. You can look at the outputs and determine which key is correct.

You are encouraged to reused the code in mini-project 1.

Next, use the brute force password cracker for the Vigenere Cipher you implemented in Project 2 to decrypt the following messages:

1. "MSOKKJCOSXOEEKDTOSLGFWCMCHSUSGX";

key length = 2; firstWordLength = 6

2. "OOPCULNWFRCFQAQJGPNARMEYUODYOUNRGWORQEPVARCEPBBSCEQYEARAJUYGWWYACYWBPRNEJBMDTEAEYCCFJNENSGWAQRTSJTGXNRQRMDGFEEPHSJRGFCFMACCB"

keyLength=3; firstWordLength = 7

3. "MTZHZEOQKASVBDOWMWMKMNYIIHVWPEXJA"

keyLength=4; firstWordLength = 10

4. "HUETNMIXVTMQWZTQMMZUNZXNSSBLNSJVSJQDLKR"

keyLength=5; firstWordLength = 11

5. "LDWMEKPOPSWNOAVBIDHIPCEWAETYRVOAUPSINOVDIEDHCDSELHCCPVHRPOHZUSERSFS"

keyLength=6; firstWordLength = 9

6. "VVVLZWWPBWHZDKBTXLDCGOTGTGRWAQWZSDHEMXLBELUMO"

keyLength=7; firstWordLength = 13

When key length increases, cracking becomes slow. Please at least finish the first 4. Please record the time needed to decrypt each message.