Network Security (CSE350)
Programming Assignment no. 1
(Project 1)

Programming language: C++
Platform: MS Windows

In this project, we have implemented a Poly-alphabetic substitution cipher system using a symmetric key of known length 4. We implemented functions for encryption, decryption, hashing of a string of lower case English letters, viz. {a, b, …,z}. We have also implemented a function recognizable() to test if a plaintext is recognizable or not. And finally, we implemented the bruteforce() function to find the key.

We read the value of the key from a .txt file. We also read 5 string/original text from a .txt file. We have limited the length of original text in order to limit the length of plain-text so that the computations can be carried out in good time. We have also checked if the key and the original texts are strings of lowercase English alphabet or not.
We then obtain the plaintext from the original text in this form:
p = (string, Hash(string))
We obtain this form by concatenating the original text and the hash value of the original text using the "+" operator of strings. The hash value is obtained using the Hash() function.

Then we encrypt the set of plain texts using the given symmetric key of length 4 using the encrypt() function to obtain the corresponding cipher texts.

Then we decrypt the set of cipher texts using the same key using the decrypt() function to obtain the corresponding plain texts to ensure the encryption and decryption algorithms are working fine. Then we launch a brute force attack by testing each key combination from "aaaa" to "zzzz" for each cipher text using the recognizable() function to find the key.

Functions implemented in the system:

1. check_string() : Takes the original text as parameter and returns false if any character is not a lower case English alphabet, else returns true.

2. encrypt() : Takes two strings str and key as parameters and encrypts the given str into the corresponding ciphertext using poly-alphabetic substitution algorithm using the given key and returns the ciphertext/ string.

3. decrypt() : Takes two strings str and key as parameters and decrypts the given str into the corresponding plaintext using poly-alphabetic substitution algorithm using the given key and returns the plaintext/ string.

4. Hash() : Takes a string/ original text as parameter and returns the hash value of the original text. The hash value or hash string has a length of 16. Hash value is calculated by placing all characters of original text row-wise in a matrix of 16 columns. Then for each column the integer ASCII values of every character in that column is added, then the resulting integer is converted into a character by doing a modulo operator with 26, then adding 97 and storing the final integer as a character . Hence the resulting 16 characters fall in the range of lower-case English alphabets. These 16 characters are returned as a string in order also called the hash value.

5. recognizable() : Takes a string plaintext and returns true if the given string satisfies the π property, else returns false.

6. bruteForce() : Iterates from “aaaa” to “zzzz” to choose a key. For each key, we decrypt all ciphertexts using this key and check the π property using the recognizable function() by passing the decrypt ciphertexts as parameter and if the chosen key holds true for all the ciphertexts, it returns the found key. Else we drop this key and repeat this process for a new key. If no key is found, print an error message.

π property for a given plaintext:

Last 16 characters of plaintext (Hash String) == Hash (Remaining characters of plaintext)

Sample Input/Outputs:

The sample Input/Outputs are show below :

```
PS C:\Users\vansh\Desktop\NSC-A1> cd "c:\Users\vansh\Desktop\NSC-A1\" ; if ($?) { g++ NSC-A1.cpp -o NSC-A1 } ; if ($?) { .\NSC-A1 }
Known key: high

First OrinialText   : wearediscoveredsaveyourselfover
First PlainText     : wearediscoveredsaveyourselfoverilqbejlwslmeyugd
First CipherText    : dmgyllozjwblymjzhdkfvcxzltlvcmxpsyhlqtczsukfboj
First decryptedText : wearediscoveredsaveyourselfoverilqbejlwslmeyugd

Second OrinialText   : yesthisisthefirstassignmentofnetworksecuritycourse
Second PlainText     : yesthisisthefirstassignmentofnetworksecuritycourseduysepelklqnjgmz
Second CipherText    : fmyaoqypzbnlmqxzaiyzpottlvzvmvkadwxrzmibyqzfjwayzmjbfakwltqsxvpnth
Second decryptedText : yesthisisthefirstassignmentofnetworksecuritycourseduysepelklqnjgmz

Third OrinialText   : cannotfindagoodexampleforanoriginaltextandfinallytheassignmentiscompletedontime
Third PlainText     : cannotfindagoodexampleforanoriginaltextandfinallytheassignmentiscompletedontimedyuffhfzryiqesxf
Third CipherText    : jituvblpulgnvwjleiswsmlvyitvyqmpuiralfzhullpuirsfbnlhaypnvslubozjwswsmzlkwtapukkfclmonfyfqwlzfl
Third decryptedText : cannotfindagoodexampleforanoriginaltextandfinallytheassignmentiscompletedontimedyuffhfzryiqesxf

Fourth OrinialText   : thisisthefourthsampleinput
Fourth PlainText     : thisisthefourthsampleinputfffjpymsikkzfcesd
Fourth CipherText    : apozpazolnubybnzhuvslqtwbblmqxetzqqrgnilzl
Fourth decryptedText : thisisthefourthsampleinputfffjpymsikkzfcesd

Fifth OrinialText   : thisisgoingtobethebiggestofallthesampleinputsokokokokokokokokokokokokokokokokwehaveusedpolyalphabeticsubstitutionalgorithm
Fifth PlainText     : thisisgoingtobethebiggestofallthesampleinputsokokokokokokokokokokokokokokokokwehaveusedpolyalphabeticsubstitutionalgorithmzkmayzibmuyueruk
Fifth CipherText    : apozpamvpvmavjkaomhpnokzawlhstzolagtwtkpuxaazwqvrwqvrwqvrwqvrwqvrwqvrwqdlpgclcylkxusfirwoihlaqizbjyapbaapwthsouypbntgsshfhoitceblzar
Fifth decryptedText : thisisgoingtobethebiggestofallthesampleinputsokokokokokokokokokokokokokokokokwehaveusedpolyalphabeticsubstitutionalgorithmzkmayzibmuyueruk

LAUNCHING BRUTE FORCE ATTACK...

Found Key: high
PS C:\Users\vansh\Desktop\NSC-A1>
```