CS001X: Introduction to Computer Programming with Python

Department of Computer Science, University of Pittsburgh

Project 2: Cryptography - Lab 2

Introduction

For this assignment you will need to create a basic cryptography program based the Vigenère cipher.

To Note:

- You can assume that the length of the text to be encrypted/decrypted is always greater than zero.
- **If you find a character that's *not a letter*,** i.e. a number, a space, or a punctuation character (, ; . ! @ # $ & % ,etc.), **do not change it, leave as is, and copy it over.**
- **Although the examples given in these instructions are a good place to start. Your program must work with any encrypted/decrypted messages given in the inputs.**

The Vigenère cipher decodes text based on a second string that is called **key**. The **key** must be the same **key** used to encrypt the text if you want to decrypt the encrypted text to discover the original text. The alphabetical position of each of the characters in the key determines how many positions the character in the original text must be shifted. The letter "A" in the key means a shift of zero, "B" produces a shift of 1, "C" a shift of 2 and so on. A key can contain more than one character, in those cases you need to 'extend' the key by repeating it as many times as necessary until it reaches the same length of the whole text. **Your program must ask the user for the Decoding Key**. To learn more about the cipher visit: https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher.

For example, suppose that the text to be encrypted is "pittsburgh pa" and the key is "yes". The first step is to concatenate "yes" over and over until it's same length as "pittsburgh pa", consider white spaces as a special character. We did this task in lab 1.

Encrypting Text:

TEXT	pittsburgh pa
key	yes
adjusted key	yesyesyesyes (we ignore anything that's not a letter, so key is one character shorter for this example)
calculated shift	24, 4, 18, …
encrypted text	nmlrwtsvyf ts

Now that we have encrypted the text, no one else will know that 'nmlrwtsvyf ts' represents 'pittsburg pa' unless they have our **decryption key**. To be able to decrypt 'nmlrwtsvyf ts' back to 'pittsburgh pa', we need to decrypt the text. Remember, the **key must be the same!**

Decrypting Text:

TEXT	nmlrwtsvyf ts
key	yes
adjusted key	yesyesyesyes (we ignore anything that's not a letter, so key is one character shorter for this example)
calculated shift	24,4,18, …
decrypted text	pittsburgh pa

TEXT	computer programming
key	python
adjusted key	pythonpythonpythonp (we ignore anything that's not a letter, so key is one character shorter for this example)
calculated shift	15, 24, 19, 7, 14, 13
encrypted text	rmfwigtp iyctgyftwav

TEXT	rmfwigtp iyctgyftwav
key	python
adjusted key	pythonpythonpythonp (we ignore anything that's not a letter, so key is one character shorter for this example)
calculated shift	15, 24, 19, 7, 14, 13
decrypted text	computer programming

Part 2

1) Write a function **decrypt_vigenere()**, that doesn't take any arguments. The function should ask the user for the text to be decrypted (use the input() function) and for an decoding key.

*IMPORTANT: To make the process of decrypting easier, convert both text and key to lower case using the .lower() method. (You will learn more about lower when we cover chapter 8).

text = 'Hi, my name is Robocop!'
new_text = text.lower() new_text ->>>
'hi, my name is robocop!'

After that, your function should call adjusted_key(), passing text and key so an adjusted key can be created for the encryption process.

The next step is to use text and the adjusted key to create the decrypted string. Now that you know how to encrypt text, decrypting should be very simple. Use a loop to visit every character in the text and to concatenate the resulting decrypted letter to a variable that holds the decrypted text. Remember that only letters should be decrypted. Numbers, punctuation, and other special characters should not be touched and concatenated to result as is. Finally, print the decrypted text to the screen

2) Reading and writing to files

Modify encrypt_vigenere() to prompt the user to enter the name of a file in .txt format containing the text to be encrypted. Use the text in that file as the source to be encrypted text instead of obtaining it from the keyboard input. Also, prompt the user for a name for an output file. The function now should write the encrypted message to that file instead of printing it to the screen.

Modify decrypt_vigenere() to prompt the user to enter the name of a file in .txt format containing the text to be decrypted. Use the text in that file as the source to be decrypted text instead of obtaining it from the keyboard input. Also, prompt the user for a name for an output file. The function now should write the decrypted message to that final instead of printing it to the screen.

When testing your program, you original text should match the decrypted text when passing in the same key for encoding and decoding. View the example below:

Encoding:

textfile1.txt contains 'pittsburgh pa' [**input file for encryption**]

encodedtextfile1.txt contains 'nmlrwtsvyf ts' [**output file for encryption**]

Decoding:

encodedtextfile1.txt contains 'nmlrwtsvyf ts' [**input file for decryption**]

decodedtextfile1.txt contains 'pittsburgh pa' [new **output file for decryption**]

Notes:

How to determine if a character is a letter:

Recall from the lectures that every character in the ASCII table has a numeric code. 'a'
= 97, 'b' = 98, … , 'z' = 122
'A'= 65, 'B' = 66, … , 'Z' = 90

You can obtain the numeric code of a character by using the function ord()

ord('a') = 97, ord('t')= 116…

So, if ord(character) is a number between 97 and 122 it is a letter in lower-case. If ord(character) is a number between 65 and 90 it is a letter in upper-case.

To convert a numeric code into the corresponding character, we can use the chr() function.

chr(72) = 'H', chr(114) = 'r', chr(65) = 'A'

Applying a shift for decoding:

Consider a shift of 3 and the character 'c'

ord('c') = 99

(original character from file) 99 + 3 = 102

(code for character + shift) chr(102) = 'f'

(final character encrypted)

What if shift + character code exceeds 122?

To keep the result encoded using only letters, if shift is greater than the numeric code for z (122) we must begin again at a. i.e.

For ‘y’ and shift of 10 -> ord(‘y’) = 121 y-> z ->

a-> b-> c-> d-> e-> f-> g-> h-> i ord(‘I’) =105

How to do that?

for lower-case letters we must keep the numeric codes within [97, 122]. That can be done as follows:

1) obtain code using ord()
2) subtract 97 from it
3) subtract shift to code
4) add 26 to that
4) obtain the remainder of the division of the previous number by 26 (there are 26 letters in the alphabet)
5) add 97 to that
6) covert code back to string using chr().

ord(‘y’) = 121 -> 121– 97 = 24 -> 24+ 10 = 34 -> 34%26 = 8 -> 9+97 = 105 -> chr(105) = ‘i’

for upper-case letters we must keep the numeric codes within [65, 90]. That can be done as follows:

7) obtain code using ord()
8) subtract 65 from it
9) subtract shift to code
10) add 26 to that
10) obtain the remainder of the division of the previous number by 26 (there are 26 letters in the alphabet)
11) add 65 to that
12) covert code back to string using chr().

ord(‘Y’) = 89 -> 89 – 65 = 24 -> 24+ 10 = 34 -> 34%26 = 8 -> 8+65 = 73 -> chr(73) = ‘I’