# CS001X: Introduction to Computer Programming with Python

Department of Computer Science, University of Pittsburgh

Project 2: Cryptography - Lab 1

---

**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 encodes text based on a second string that is called **key**. 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 Enconding 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.

| 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 | 25, 5, 19, … |
| encrypted text | nmlrwtsvyf ts |

| TEXT | pittsburgh pa |
|---|---|
| key | b |
| adjusted key | bbbbbbbbbbbb (we ignore anything that's not a letter, so key is one character shorter for this example) |
| calculated shift | 1, … |
| encrypted text | qjuutcvshi qb |

| TEXT | pittsburgh pa |
|---|---|
| key | a |
| adjusted key | aaaaaaaaaaaa (we ignore anything that's not a letter, so key is one character shorter for this example) |
| calculated shift | 0, … |
| encrypted 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 |

**Part 1**

1) Write a function **adjusted_key(text, key)**, that takes two string arguments: text and key, and returns the key adjusted to fit the length of the string found in text.

Ex.
text = 'I love python programming'
key = 'abc'

adjustedkey = 'abcabcabcabcabcabcabca'

2) Write a function **encrypt_vigenere()**, that doesn't take any arguments. The function should ask the user for the text to be encrypted (use the input() function) and for an encoding key.

*IMPORTANT: To make the process of encrypting 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 encrypted string. Use a loop to visit every character in the text and to concatenate the resulting encrypted letter to a variable that holds the encrypted text. Remember that only letters should be encrypted. Numbers, punctuation, and other special characters should not be touched and concatenated to result as is.

Finally, print the encrypted text to the screen

**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**:

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('l') =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) add shift to code
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) add shift to code
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'

To make your program a bit simpler, you can convert an uppercase character into lowercase before applying the shift. To convert from uppercase to lowercase you can use the lower() method or you can add 32 from the characters code if:

$65 \leq \text{ord(character)} \leq 90$

Example:

- character = 'P'
- ord('P') = 80
- $65 \leq 80 \leq 90$ #True
- 80 +32 = 112
- chr(112) = 'p'