

Assignment #1

You intercepted a single ciphertext. Decipher it as much as you can. To receive full or partial credit you must show all your work. Attach any code you have implemented (you can use any programming language) and you can use any libraries you find online that is publicly available. You may not use existing code available on online repositories such as GitHub. You must include citations of all sources you used in the report.

**HUKUUEUUYREUYYKGRRGNZZKXNGNOLKXTSAQNZYEGNZTRGTOYALSSEH
SYGRVSOOLEGIKVKNZOEJYXT**

1 Introduction

In order to approach this assignment a large amount of research had to be done in order to understand the complexity of ciphers and how they are deciphered. Hints were given off throughout the class sessions along with the documentation of the project so that was the beginning.

Due to the open nature of the this project, researching Caesar ciphers, Columnar transposition, and much more topics made the assignment intimidating from the start. Site such as learn cryptography^[1] and dcode^[3] plus many random blogs increased my understanding. After hours of reach I felt confident about the assignment and began cracking down on the cipher.

2 Solving the Cipher

At first, I tried solving by hand which lead me to a dead end. Initially I performed a Caesar Cipher by hand using all available keys which was too time consuming and redundant. Python became my new best friend during this assignment because of my previous experience using it and the many libraries that it came with. At first I wrote a script that would perform a Caesar Cipher just so I can have idea of how to manipulate the string(Cipher). After that I looked at each shift and saw that the sixth shift really stood out because it had a combination of letters that just made sense. I broke it down and pull words from the shifted key by hand and saw some basic words like be, happy, fun, your, ect. It was only until later that this way of analyzing letters is called "Frequency Analysis". I had trouble writing a script for it, instead I ran it through dcode which spat out some number that didn't help me much. I was stuck at a brick wall until PyCipher came in.

Along my research I stumbled upon PyCipher^[2] which made solving this Cipher extremely less stressful. Built in were tools such as Caesar Cipher and Columnar Transposition which were given in the hints. PyCipher's method of implementing a Caesar cipher became a drop in replacement from my original method. In order to use PyCipher Column Transposition tool I had to find a key, which created another roadblock because the key was not given. Up until the

professor mentioned the use of `itertools`^[7], a python library, to create permutations of generated keys. I created the Columnar Transposition using a key of 01234567 because the key had to be between 1-10 and there are 77 letters, so 7 fit perfectly. I used the six shifted cipher because I was able to pull some words from it by hand. I ran the python script and appended the results in `colTrans.txt`. I loaded up the text file in my editor and by pure luck used `ctrl+f` and typed happy, one of my guessed words which resulted in: **03247615:**

The screenshot shows a TextEditor application with a file explorer on the left and a code editor on the right. The file explorer lists several files: 'caesar.txt', 'colTrans.txt', 'dictAttack.py', 'noodle.py', and 'permutations.txt'. The code editor displays the contents of 'noodle.py', which is a Python script implementing a Caesar cipher. The script defines a 'caesar' class with methods for encryption and decryption. It takes a message and a shift value as input, encrypts the message by shifting each letter, and then decrypts it back to the original message. The output is printed to the console. The status bar at the bottom indicates the cursor is at line 1, column 1, in UTF-8 encoding.

```

1  # Caesar Cipher
2  def caesar.Salad(ciphertext):
3      caesar_shift = {}
4      for x in range(1, 26):
5          caesar_shift[x] = caesar(x).decipher(ciphertext)
6      return caesar_shift
7
8  # Run
9  ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
10 # Key shift & looks the most human readable
11 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
12 outputfile = "output.txt"
13
14 # PyPi Caesar Cipher
15 def caesar.Salad(ciphertext):
16     caesar_shift = {}
17     for x in range(1, 26):
18         caesar_shift[x] = caesar(x).decipher(ciphertext)
19     return caesar_shift
20
21 # Run
22 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
23 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
24 outputfile = "output.txt"
25
26 # PyPi Caesar Cipher
27 def caesar.Salad(ciphertext):
28     caesar_shift = {}
29     for x in range(1, 26):
30         caesar_shift[x] = caesar(x).decipher(ciphertext)
31     return caesar_shift
32
33 # Run
34 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
35 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
36 outputfile = "output.txt"
37
38 # PyPi Caesar Cipher
39 def caesar.Salad(ciphertext):
40     caesar_shift = {}
41     for x in range(1, 26):
42         caesar_shift[x] = caesar(x).decipher(ciphertext)
43     return caesar_shift
44
45 # Run
46 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
47 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
48 outputfile = "output.txt"
49
50 # PyPi Caesar Cipher
51 def caesar.Salad(ciphertext):
52     caesar_shift = {}
53     for x in range(1, 26):
54         caesar_shift[x] = caesar(x).decipher(ciphertext)
55     return caesar_shift
56
57 # Run
58 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
59 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
60 outputfile = "output.txt"
61
62 # PyPi Caesar Cipher
63 def caesar.Salad(ciphertext):
64     caesar_shift = {}
65     for x in range(1, 26):
66         caesar_shift[x] = caesar(x).decipher(ciphertext)
67     return caesar_shift
68
69 # Run
70 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
71 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
72 outputfile = "output.txt"
73
74 # PyPi Caesar Cipher
75 def caesar.Salad(ciphertext):
76     caesar_shift = {}
77     for x in range(1, 26):
78         caesar_shift[x] = caesar(x).decipher(ciphertext)
79     return caesar_shift
80
81 # Run
82 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
83 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
84 outputfile = "output.txt"
85
86 # PyPi Caesar Cipher
87 def caesar.Salad(ciphertext):
88     caesar_shift = {}
89     for x in range(1, 26):
90         caesar_shift[x] = caesar(x).decipher(ciphertext)
91     return caesar_shift
92
93 # Run
94 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
95 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
96 outputfile = "output.txt"
97
98 # PyPi Caesar Cipher
99 def caesar.Salad(ciphertext):
100    caesar_shift = {}
101    for x in range(1, 26):
102        caesar_shift[x] = caesar(x).decipher(ciphertext)
103    return caesar_shift
104
105 # Run
106 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
107 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
108 outputfile = "output.txt"
109
110 # PyPi Caesar Cipher
111 def caesar.Salad(ciphertext):
112    caesar_shift = {}
113    for x in range(1, 26):
114        caesar_shift[x] = caesar(x).decipher(ciphertext)
115    return caesar_shift
116
117 # Run
118 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
119 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
120 outputfile = "output.txt"
121
122 # PyPi Caesar Cipher
123 def caesar.Salad(ciphertext):
124    caesar_shift = {}
125    for x in range(1, 26):
126        caesar_shift[x] = caesar(x).decipher(ciphertext)
127    return caesar_shift
128
129 # Run
130 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
131 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
132 outputfile = "output.txt"
133
134 # PyPi Caesar Cipher
135 def caesar.Salad(ciphertext):
136    caesar_shift = {}
137    for x in range(1, 26):
138        caesar_shift[x] = caesar(x).decipher(ciphertext)
139    return caesar_shift
140
141 # Run
142 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
143 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
144 outputfile = "output.txt"
145
146 # PyPi Caesar Cipher
147 def caesar.Salad(ciphertext):
148    caesar_shift = {}
149    for x in range(1, 26):
150        caesar_shift[x] = caesar(x).decipher(ciphertext)
151    return caesar_shift
152
153 # Run
154 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
155 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
156 outputfile = "output.txt"
157
158 # PyPi Caesar Cipher
159 def caesar.Salad(ciphertext):
160    caesar_shift = {}
161    for x in range(1, 26):
162        caesar_shift[x] = caesar(x).decipher(ciphertext)
163    return caesar_shift
164
165 # Run
166 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
167 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
168 outputfile = "output.txt"
169
170 # PyPi Caesar Cipher
171 def caesar.Salad(ciphertext):
172    caesar_shift = {}
173    for x in range(1, 26):
174        caesar_shift[x] = caesar(x).decipher(ciphertext)
175    return caesar_shift
176
177 # Run
178 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
179 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
180 outputfile = "output.txt"
181
182 # PyPi Caesar Cipher
183 def caesar.Salad(ciphertext):
184    caesar_shift = {}
185    for x in range(1, 26):
186        caesar_shift[x] = caesar(x).decipher(ciphertext)
187    return caesar_shift
188
189 # Run
190 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
191 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
192 outputfile = "output.txt"
193
194 # PyPi Caesar Cipher
195 def caesar.Salad(ciphertext):
196    caesar_shift = {}
197    for x in range(1, 26):
198        caesar_shift[x] = caesar(x).decipher(ciphertext)
199    return caesar_shift
200
201 # Run
202 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
203 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
204 outputfile = "output.txt"
205
206 # PyPi Caesar Cipher
207 def caesar.Salad(ciphertext):
208    caesar_shift = {}
209    for x in range(1, 26):
210        caesar_shift[x] = caesar(x).decipher(ciphertext)
211    return caesar_shift
212
213 # Run
214 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
215 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
216 outputfile = "output.txt"
217
218 # PyPi Caesar Cipher
219 def caesar.Salad(ciphertext):
220    caesar_shift = {}
221    for x in range(1, 26):
222        caesar_shift[x] = caesar(x).decipher(ciphertext)
223    return caesar_shift
224
225 # Run
226 ciphertext = "HUKUEUYREYUQKGRONZKQXNOLKJTSQKZYEZNTGTOTYALSSEHSYGVSGOLGKXWZOEJYKT"
227 size = "HUCQYDQSLWSSALLANTBHWLFEWMLKJTSYNTNANLISUYPWMSALPHLTFACEPHTYDSN"
228 outputfile = "output.txt"
229
230 # PyPi Caesar Cipher
231 def caesar.Salad
```

Solving the Cipher was a stroke of luck because my last option was to brute for the colTrans.txt with a dictionary. If I did not luckily pick out the random word “happy” by hand my next python function would have to compare the dictionary values in colTrans to a random wordlist, such as rock4you or /usr/share/dict/words found in unix system. There was also the option to create a python script to automate the whole process to spit out every Caesar Cipher and permutation to use for columnar transposition, then brute forcing those results.

- [1] <https://learncryptography.com/classical-encryption/caesar-cipher>
- [2] <http://pycipher.readthedocs.io/en/master>
- [3] <http://www.dcode.fr/caesar-cipher>
- [4] <http://www.dcode.fr/frequency-analysis>
- [5] <http://practicalcryptography.com/ciphers/columnar-transposition-cipher/>
- [6] <https://inventwithpython.com/hacking/chapter20.html>
- [7] <https://docs.python.org/3/library/itertools.html#itertools.permutations>