INTRODUCTION TO CRYPTOGRAPHY – LAB 4

B.Tech. Computer Science and Engineering (Cybersecurity)

|  |  |
| --- | --- |
| Name: Anish Sudhan Nair | Roll No.: K041 |
| Batch: K2/A2 | Date of performance: 04/02/2022 |

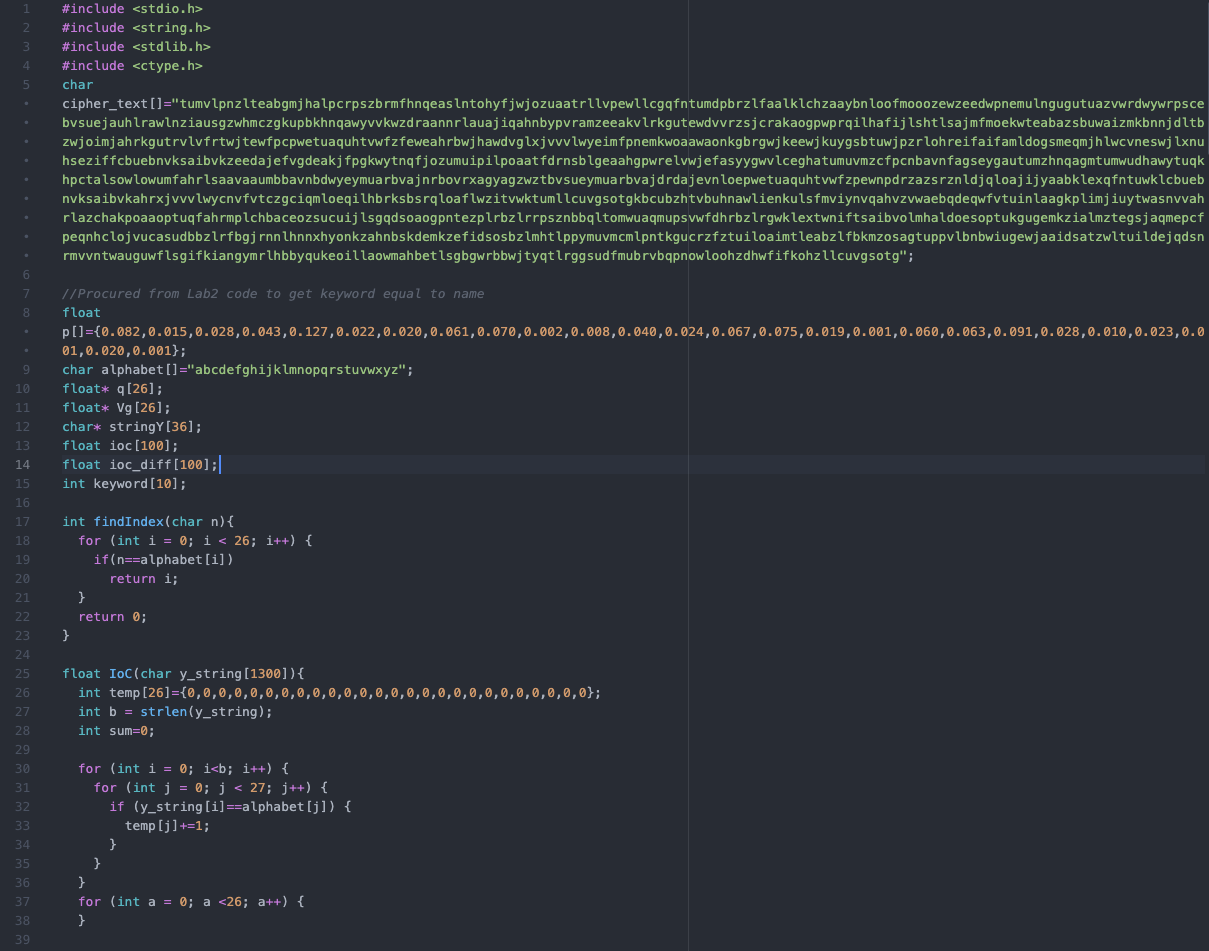
Aim: To code the Diffie-Hellman algorithm for public-private key encryption

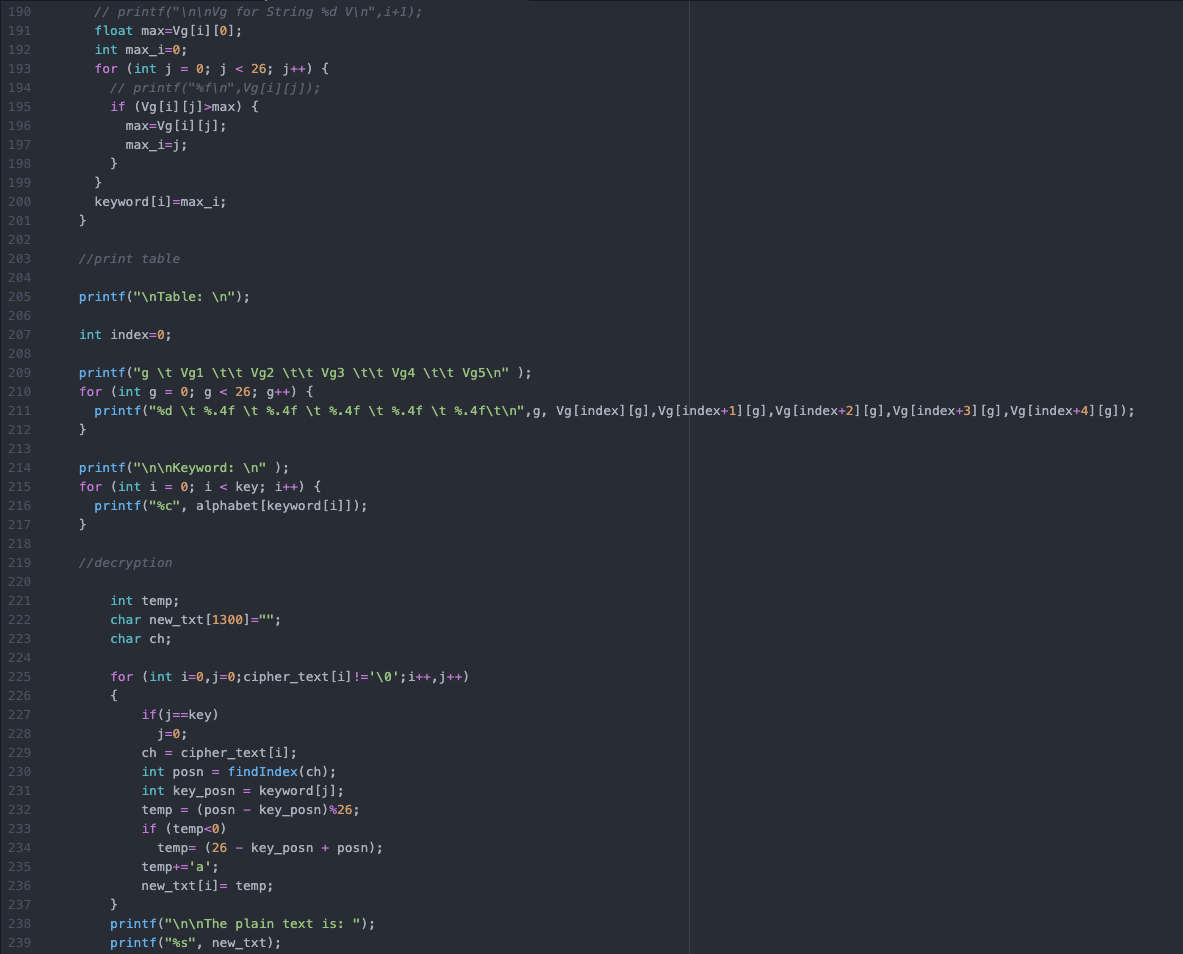
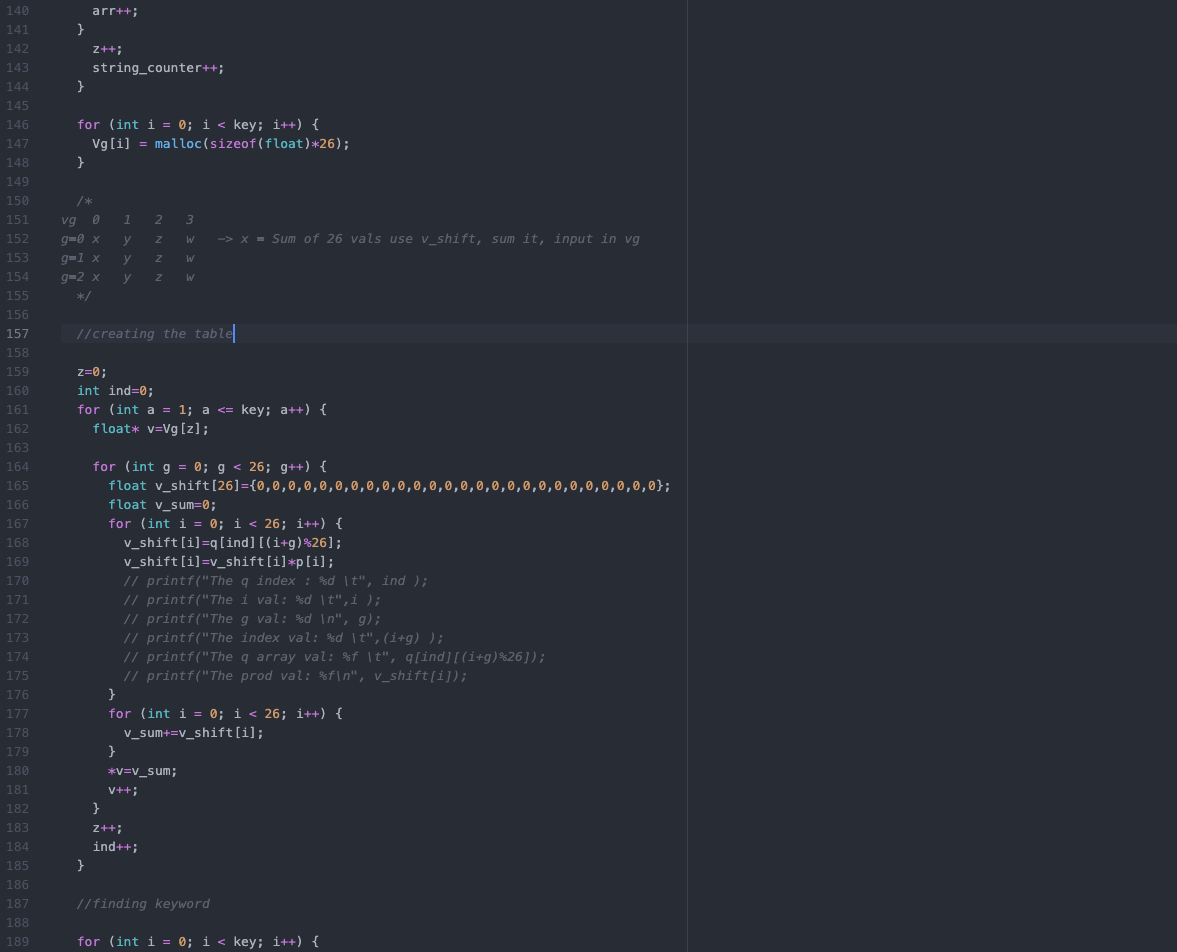
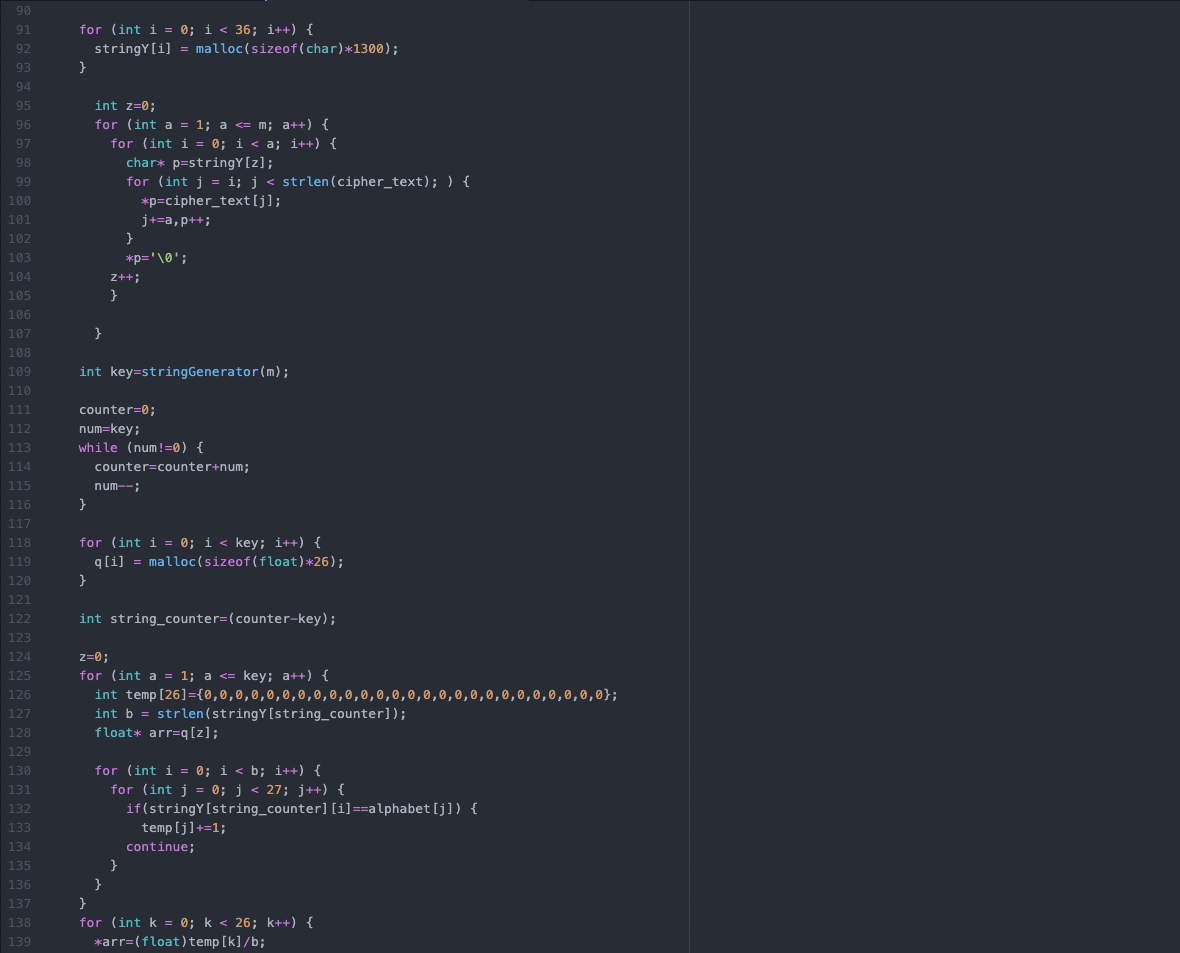
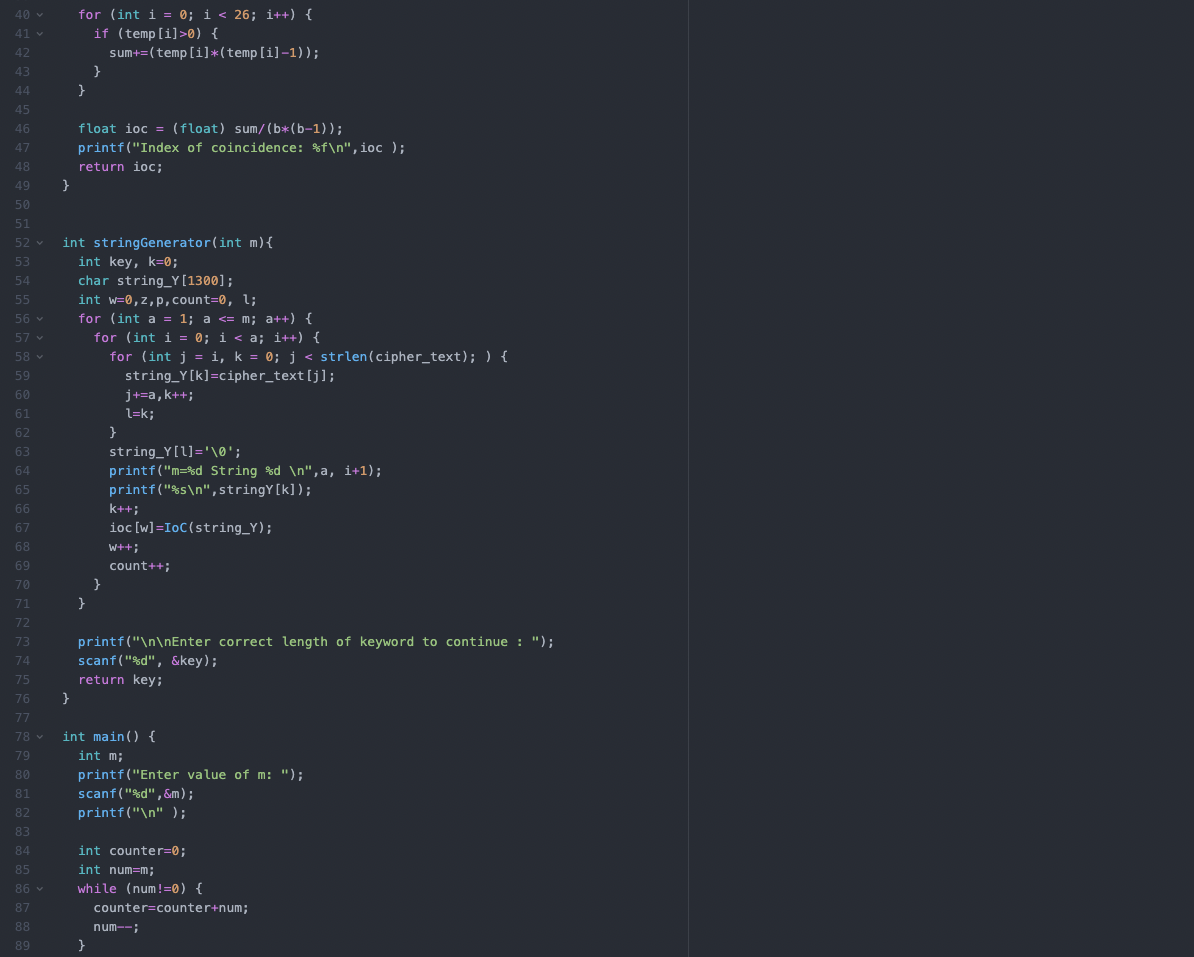
**Code:**

Language: C

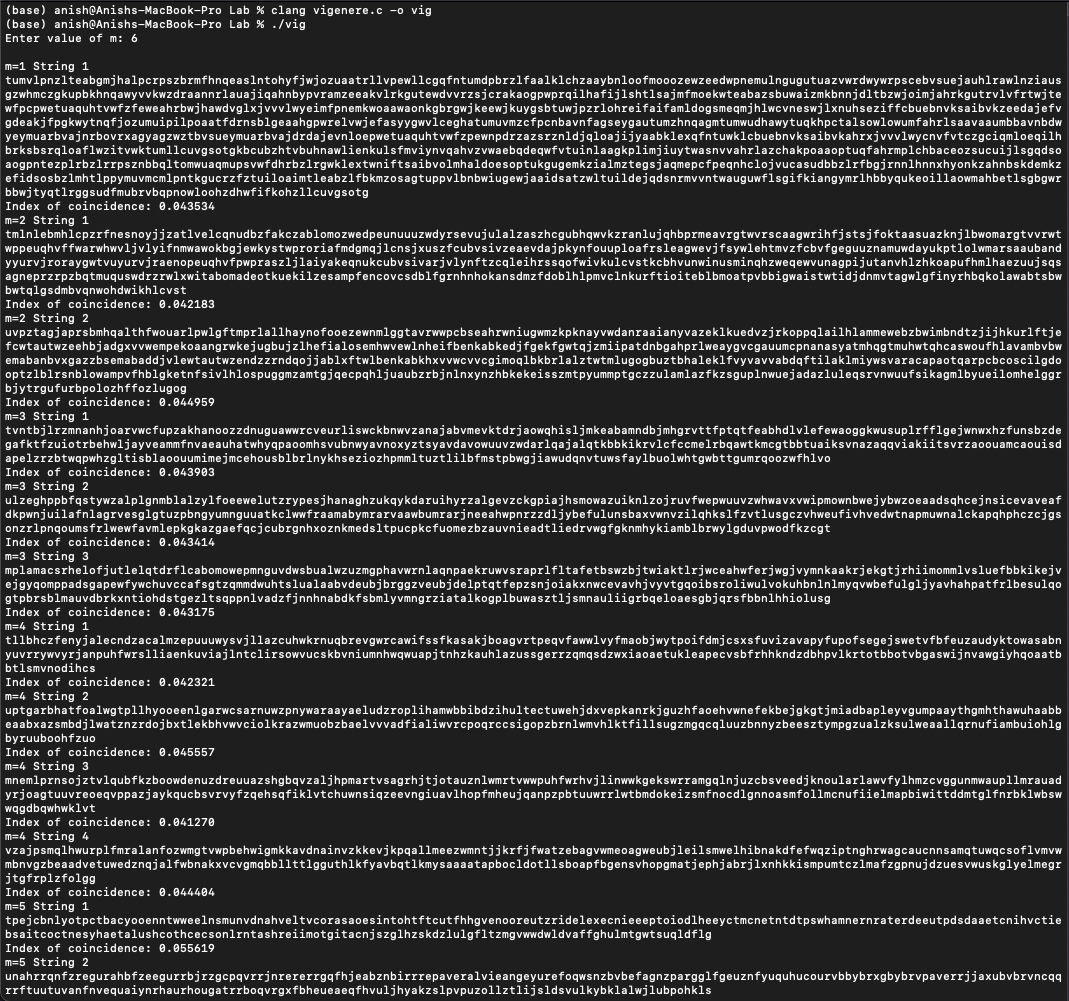
Editor: Atom

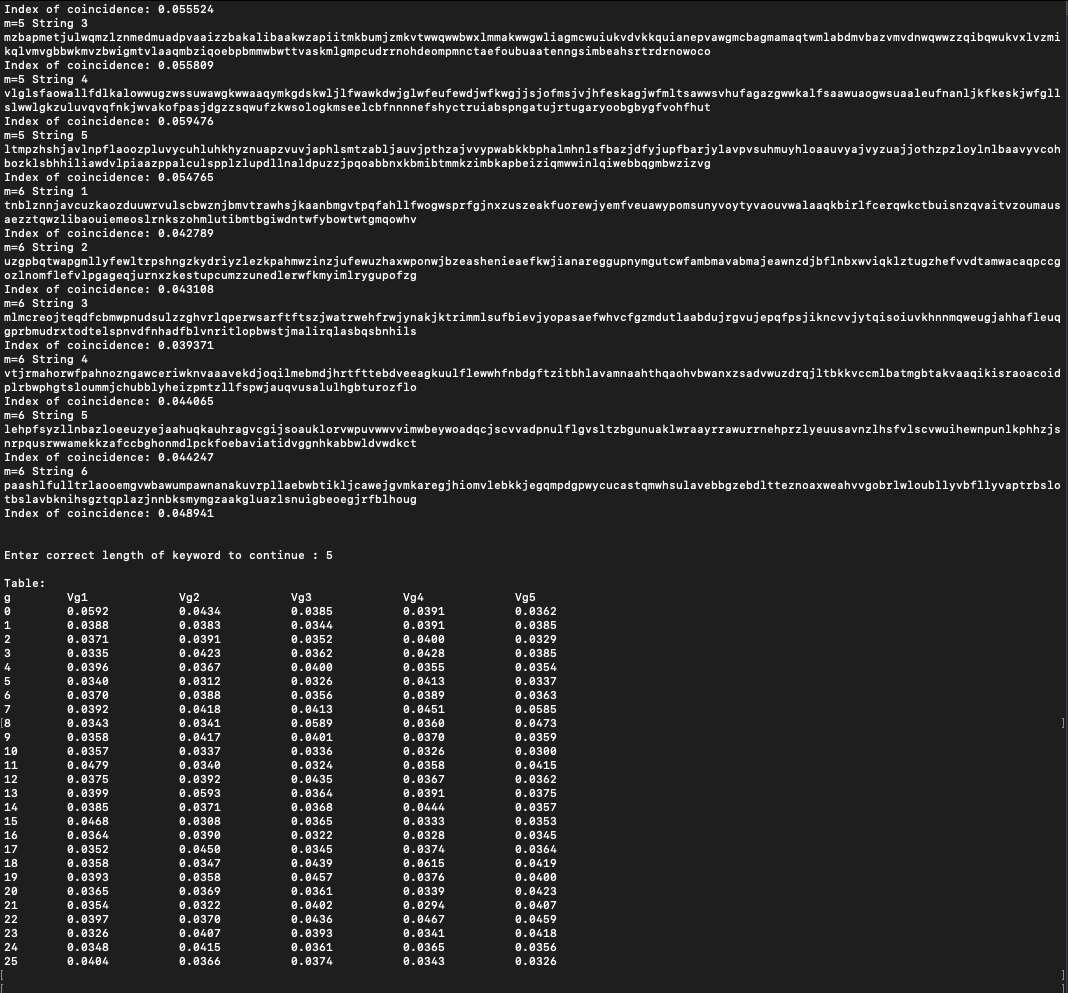
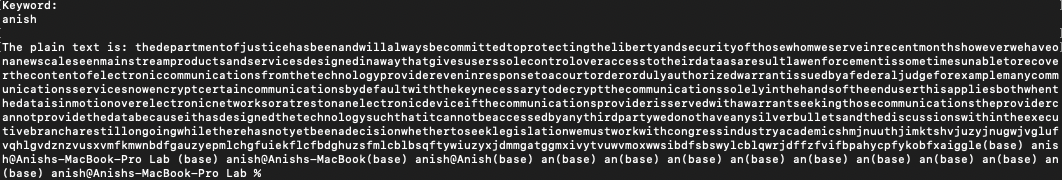
Compiler: clang/ZSH





**Complete Output:**





Questions:

1.What are various ways of cryptanalysis? Explain any two methods.

-> Cryptanalysis of the Vigenere Cipher is actually a two step process that involves finding the length of the keyword first and then figuring out the keyword itself.

Method 1: Displacement-Coincidences

This method involves comparing the cipher text against itself with shifted places. The simple nature of this method makes it easy to implement even on a piece of paper.

As aforementioned, you begin by fixing the position one cipher text’s letters and then compare the another string/paper with the san cipher text by shifting the places one by one until all places are exhausted.

Whilst shifting places, the cryptanalyst must take of the number of times a particular position has the same letter on either text.

Finally, the results are to be tabulated and the shift with the most number of coincidences is guessed/assumed to be the keyword’s length.

Once the keyword length has been figured out, it’s just a matter of performing a frequency analysis on the sets of letters of the cipher that each position of the keyword corresponds to.

Method 2: Friedman Test

This is the most mathematicaly complex method for the cryptanalysis of the Vigenere cipher.

The first step is to guess the length of the keyword, assuming m=1,2,3,4…

Depending on the value of m, m sub strings are to be created taking every mth letter of the string for the sub string, shifting the starting placer by a unit for each sub string.

After creating the substrings we find the Index of Coincidence (explained in question 3) of each sub string and if our assumed value of m has substrings whose Index of Coincidence has values near 0.065, we can be fairly certain that our assumption is valid.

Once the keyword length has been determined, we must move on to finding the actual keyword. For this, we firstly need to keep a frequency table ready that charts the frequencies of the letters of the particular language’s alphabet (Eg: The table from Cryptographical Mathematics by Robert Edward Lewand for the English alphabet) in usual text.

For each sub string for the particular value of m, we are to create a vector q that would consist of the frequency of appearance of each character of the alphabet divided by the length of the substring.

Following the creation of m such vectors, we are to create another vector vg that would consist of the elements of q shifted by g places, with g=0-25. As such, for every vector q, we would have 26 vg vectors.

After the creation of vg vectors, we proceed to compute the dot product of p and every vg. This would result in a new vector M for every q that would consist of 26 values. These values are to be arranged in respective columns and from each column, the highest value (which would be near 0.065) is to be chosen. Corresponding to the values, their respective shift (g) is to be noted and using the alphabet encoding table where every letter from A-Z is assigned a numeric value from 0-25, we find out the corresponding letters and compose them together to form the keyword.

After this point, we simply decrypt the text using the keyword’s alphabetically assigned number to subtract them from the cipher text’s numeric values and cycling the keyword until the ciphertext is exhausted.

2.Explain Kasiski’s method with an example.

Kasiski’s method is the third method of cryptanalysis of the Vigenere cipher which is concerned with the presence of digrams, trigrams and so on, in the cipher text.

Therefore, we create a frequency table of those digrams and trigrams appearing more than once and their position occurrence.

We then take two of those groups of letter that have highest frequency and compute the difference of the starting positions of their consecutive occurrences. We then take the gcd of the position differences of the two groups which would work out to be the keyword length.

Once the keyword length has been figured out, it’s just a matter of performing a frequency analysis on the sets of letters of the cipher that each position of the keyword corresponds to.

Eg:krgclgcgsbvcxxjimldlcwmbnpcyjqdybimlqjmblgcivkqqkrbrigclyfmlqpmdwmpjsxhmsresx

n-grams -> frequency -> starting posns. -> difference

iml: 2 ->16, 34 ->18

lgc: 2 ->5, 41 -> 36

gcl: 2 ->3, 54 -> 51

mlq: 2 ->35, 59 -> 24

gcd(18,36)=18 (unlikely since the number is quite high, reserve it for later test)

gcd(18,51)=3

Consider keyword length=3,

Using tool: <https://crypto.interactive-maths.com/kasiski-analysis-breaking-the-code.html#act>

We decipher the keyword to be KEY and the plain text to be:

anishiscurrentlyinthemiddleofstudyingforhisexamsandheishavinglotsoffundoingit

3.Explain Index of Coincidence.

The index of coincidence for a string or particular group of letters is the sum of the frequency of every letter’s appearance in the string (assume equal to n) multiplied by (n-1) and the whole sum divided by the product of the length of the string times minus 1 of the length.

Eg; Assume string x= anishstudiescryptography

It consists of the letter frequency

A 2

B 0

C 1

D 1

E 1

F 0

G 1

H 2

I 2

J 0

K 0

L 0

M 0

N 1

O 1

P 2

Q 0

R 2

S 3

T 2

U 1

V 0

W 0

X 0

Y 2

Z 0

And the length of x = 24

So, the Ic(x) = ((2)(1) + (2)(1) + (2)(1) + (2)(1) + (2)(1) + (3)(2) + (2)(1) + (2)(1))/((24)(23))

[We ignore the letters with frequency 1 since their product would turn out to be 0]

Therefore, the Ic(x)=0.036231