Segurança Computacional
Trabalho 1 - Cifra de Vigenère

Mateus Freitas Cavalcanti
16/0137519
mat.fcavalcanti@gmail.com

Mariana Mendanha da Cruz
16/0136784
mariana.mendanha.mm@gmail.com

Abstract

Este trabalho explora a cifra de Vigenère, tendo duas partes: o cifrador/decifrador e o ataque de recuperação de senha por análise de frequência. Para a parte de cifração e decifração, foi mostrado a lógica por trás da codificação de Vigenère, tal como seu processo reverso. Por fim, para a quebra de cifra, foi mostrado o modelo implementado, baseado na busca pelo melhor candidato de tamanho de chave e cálculo de X^2 para cada caracter da possível chave.

1. Introdução

Cifra, também chamada de algoritmo de criptografia são sistemas para criptografar e descriptografar dados. As cifras convertem mensagens, chamadas de plaintext, em textos cifrados, isto é, textos que se pretendem ilegíveis para quem não possui uma informação mantida secreta que chamamos de chave.

Cifras geralmente são categorizadas de acordo com o seu funcionamento e com a utilização de suas chaves. Estas podem ser simples ou complexas, a criptografia moderna usa técnicas cada vez mais sofisticadas e desempenha papel crucial em várias atividades do nosso dia a dia a começar por proteger dados confidenciais e pagamentos assegurando principalmente a integridade dos dados.

A cifra de Vigenère é uma técnica de criptografia por substituição polialfabética que usa uma série de diferentes cifras de César baseadas nas letras de uma chave.

A cifra de César consiste em um deslocamento de letras por uma posição fixa, por exemplo, com o deslocamento de 2 a letra ”A” se torna ”C”. Sabendo disso a cifra de Vigenère faz o mesmo em sequência, com valores diferentes de deslocamento que dependem de uma chave dada.

Para cifrar, usamos a tabela que pode ser vista na Figura 1 onde o alfabeto é deslocado em cada linha até completar 26 deslocamentos, aonde se encerra o ciclo. Desta forma, a chave indica cada linha que iremos utilizar para cifrar enquanto cada letra da mensagem indica a coluna.

Portanto, se temos uma chave ”AB” e mensagem ”VIDA” a mensagem criptografada será ”VJDB” visto que repetimos a chave para caber na mensagem, ou seja, ”ABAB” observamos a linha ”A” e coluna ”V” e obtemos ”V” e seguimos com o mesmo processo até termos a mensagem cifrada.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 1. A grade de Vigenère, conhecido também por tabula recta, usado para criptografia e descriptografia

2. Materiais e Métodos

- Linguagem Python
- VS Code
- Cifra de Vigenère
- Ataque por Análise de frequência

2.1. Parte I

2.1.1 Modelo

O cifrador recebe uma senha e uma mensagem que é cifrada segundo a cifra de Vigenère, gerando um criptograma, enquanto o decifrador recebe uma senha e um criptograma que é decifrado segundo a cifra de Vigenère, recuperando uma mensagem.

2.1.2 Cifrador

A função Cifrador() recebe um plaintext e uma chave, com isso, para cada letra na mensagem faz-se a operação correspondente a chave atribuída com o código unicode e um offset para o alfabeto ascii, após a operação o código para a letra volta a ser um caractere e então é retornada a mensagem criptografada.

$$E_i = (P_i + K_i) \bmod 26$$

Figure 2. Codificação de Vigenère

2.1.3 Decifrador

A função Decifrador() faz a operação inversa, recebendo o criptograma e a chave, e retornando a mensagem decifrada.

$$D_i = (E_i - K_i + 26) \bmod 26$$

Figure 3. Decodificação de Vigenère

2.2. Parte II

2.2.1 Modelo

Foram fornecidas duas mensagens cifradas (uma em português e outra em inglês) com senhas diferentes. Cada uma das mensagens deve ser utilizada para recuperar a senha geradora do keystream usado na cifração e então decifradas. Sendo assim, o algoritmo de quebra de cifra foi separado em duas etapas: Determinação de melhor candidato para tamanho da chave desconhecida, e cálculo da melhor chave.

2.2.2 Tamanho da chave

Para determinar o tamanho da possível chave geradora do keystream, foi utilizado o método baseado no índice de coincidência. Essa abordagem consiste em efetuar o cálculo de IC para diferentes tamanhos de chaves e, considerando as frequências das letras do idioma em questão, é escolhido como melhor candidato aquele tamanho de chave que apresenta o índice de coincidência associado mais próximo do índice das frequências do idioma. Sendo assim, tendo em vista um intervalo fixo de tamanhos de chaves entre 2 e 20, é possível determinar o melhor tamanho para a chave desconhecida.

$$\mathbf{IC} = \frac{\sum_{i=1}^{c} n_i(n_i - 1)}{N(N-1)/c}$$

Figure 4. Equação do índice de coincidência de uma subamostra

Para o cálculo do IC de chaves com tamanhos maiores ou iguais a 2, tem-se que o índice de coincidência associado é a média dos índices das N subamostras da mensagem cifrada, com N igual ao tamanho da chave.

2.2.3 Melhor chave

Tendo como base o tamanho da chave encontrado (N), é utilizado o método X^2 para se determinar a possível chave do criptograma. Esse método consiste em separar o texto cifrado em N subamostras, onde todas as letras de uma determinada subamostra são cifradas pela mesma letra da chave. Sendo assim, é possível decifrar cada subamostra considerando uma cifra de césar, onde cada letra do texto em claro é cifrado pela mesma letra. Calculando o X^2 para cada possível valor de 'letra chave', tem-se que a letra com menor X^2 associado é adotada como melhor candidato. Por fim, para cada N subamostra, pode-se obter as N possíveis letras da chave.

$$\chi^2(C, E) = \sum_{i=A}^{i=Z} \frac{(C_i - E_i)^2}{E_i}$$

Figure 5. Cálculo do chi square (X^2) de uma subamostra

3. Resultados

3.1. Cifração

De maneira inicial, o algoritmo de cifração proposto foi testado para diversas mensagens. Considerando a cifração da mensagem referente à enunciação da **primeira lei de Newton**, é possível verificar a mensagem encriptada, considerando a chave **newton** (figura 6).

```
#####################################
# Seguração Computacional - Trabalho 1 #
#####################################

 (*) CODIFICADOR (*)

> Plain text: ifabodyisatrestormovingataconstantspeedinastraightline,itwill
remainatrestorkeepmovinginastraightlineatconstantspeedunlessitisactedupon by
aforce
[...] Encoding 'ifabodyisatrestormovingataconstantspeedinastraightline,itwi
llremainatrestorkeepmovinginastraightlineatconstantspeedunlessitisactedupon
byaforce' with key 'newton'

> Encrypted text: vjwucqlmotherwphfzbzegungeyhbfgejmgcrizbbnfxntwtuxhbbrmmp
pwyyvafovaepksfgsndsrcqkowatmjtggeeezvgymjxogpsjlhnaxoisrqyjesffmpbgnpxawic
brxrosbvyx
```

Figure 6. Teste de cifração

3.2. Decifração

Considerando o resultado da cifração da sessão anterior, é possível utilizar a mesma chave, '**newton**', para efetuar o processo reverso e encontrar a mensagem em claro. O resultado deste processo é visto na figura 7.

```
#####################################
# Seguração Computacional - Trabalho 1 #
#####################################

 (*) DECODIFICADOR (*)

> Encrypted text: vjwucqlmotherwphfzbzegungeyhbfgejmgcrizbbnfxntwtuxhbbrmmp
pwyyvafovaepksfgsndsrcqkowatmjtggeeezvgymjxogpsjlhnaxoisrqyjesffmpbgnpxawic
brxrosbvyx
[...] Decoding 'vjwucqlmotherwphfzbzegungeyhbfgejmgcrizbbnfxntwtuxhbbrmmppw
yyvafovaepksfgsndsrcqkowatmjtggeeezvgymjxogpsjlhnaxoisrqyjesffmpbgnpxawicbr
xrosbvyx' with key 'newton'

> Decrypted text: ifabodyisatrestormovingataconstantspeedinastraightlinezit
willremainatrestorkeepmovinginastraightlineatconstantspeedunlessitisactedup
onbyaforce
```

Figure 7. Teste de decifração

3.3. Quebra de cifra

Para o teste da quebra de cifra, foram testadas mensagens cifradas em português e inglês. Sendo assim, foram obtidos os seguintes resultados:

```
 (*) QUEBRA DE CIFRA (*)

> Encrypted text: jyqkiwsrulbilgkvivbhoynznpxfmuipeusiumajkstnmbqtzwom
nkbbjiifwbayyznzbhgfxnvtcyycpskoiuwrunskgknhtsdjmemxbicifitdxyvqinagfq
pehsadifsyfojweefoxuafmfakxmqmjikcunwyconcqsxabuw

> Language: br

# Calculando índices de coincidência ...
# Índice de coincidência 'br' 0.078
> Key lenght: 02 -> IC: 0.039
> Key lenght: 03 -> IC: 0.039
> Key lenght: 04 -> IC: 0.043
> Key lenght: 05 -> IC: 0.038
> Key lenght: 06 -> IC: 0.037
> Key lenght: 07 -> IC: 0.079
> Key lenght: 08 -> IC: 0.040
> Key lenght: 09 -> IC: 0.041
> Key lenght: 10 -> IC: 0.035
> Key lenght: 11 -> IC: 0.033
> Key lenght: 12 -> IC: 0.039
> Key lenght: 13 -> IC: 0.034
> Key lenght: 14 -> IC: 0.071
> Key lenght: 15 -> IC: 0.033
> Key lenght: 16 -> IC: 0.037
> Key lenght: 17 -> IC: 0.034
> Key lenght: 18 -> IC: 0.037
> Key lenght: 19 -> IC: 0.041
> Key lenght: 20 -> IC: 0.037
[!] Tamanho mais provável (Chave): 7
```

Figure 8. Melhor candidato para tamanho de chave (tamanho 7).

De acordo com a figura 8, o índice de coincidência calculado para o tamanho de chave igual a 7 retorna o valor mais próximo ao índice da frequência de letras da língua portuguesa. Sendo assim, o tamanho de chave 7 é tido como melhor candidato. São calculados os valores de X² para todos os *shifts* das 7 subamostras da mensagem:

```
# Coset: jrkymitwjygyikdlxgaoxkkob
> 00 shift -> X² = 7318.12
> 01 shift -> X² = 1973.49
> 02 shift -> X² = 4255.20
> 03 shift -> X² = 540.53
> 04 shift -> X² = 1289.47
> 05 shift -> X² = 919.31
> 06 shift -> X² = 34.27
> 07 shift -> X² = 700.09
> 08 shift -> X² = 1673.96
> 09 shift -> X² = 426.11
> 10 shift -> X² = 5241.08
> 11 shift -> X² = 1920.33
> 12 shift -> X² = 10282.02
> 13 shift -> X² = 2759.75
> 14 shift -> X² = 8644.09
> 15 shift -> X² = 180.34
> 16 shift -> X² = 2238.74
> 17 shift -> X² = 338.75
> 18 shift -> X² = 1686.14
> 19 shift -> X² = 667.51
> 20 shift -> X² = 96.25
> 21 shift -> X² = 841.90
> 22 shift -> X² = 850.86
> 23 shift -> X² = 497.32
> 24 shift -> X² = 2256.13
> 25 shift -> X² = 2609.88
[!] Min X² value for 'g': 34.269040
```

```
# Coset: yuvnuunoiyfyugjcyfdjuxcnu
> 00 shift -> X² = 6499.38
> 01 shift -> X² = 749.52
> 02 shift -> X² = 6434.59
> 03 shift -> X² = 1893.66
> 04 shift -> X² = 2041.29
> 05 shift -> X² = 549.77
> 06 shift -> X² = 1685.19
> 07 shift -> X² = 2073.90
> 08 shift -> X² = 597.17
> 09 shift -> X² = 1693.40
> 10 shift -> X² = 8037.32
> 11 shift -> X² = 2186.46
> 12 shift -> X² = 553.58
> 13 shift -> X² = 1932.53
> 14 shift -> X² = 3392.25
> 15 shift -> X² = 3910.28
> 16 shift -> X² = 593.34
> 17 shift -> X² = 3688.64
> 18 shift -> X² = 1298.39
> 19 shift -> X² = 146.82
> 20 shift -> X² = 29.65
> 21 shift -> X² = 1117.92
> 22 shift -> X² = 14669.32
> 23 shift -> X² = 1189.31
> 24 shift -> X² = 14648.65
> 25 shift -> X² = 1831.30
[!] Min X² value for 'u': 29.653663
```

Figure 9. Cálculo X² para 1º e 2º caracteres da chave.

```
# Coset: qlizimmmizxcwkmivqiwamucw
> 00 shift -> X² = 3887.80
> 01 shift -> X² = 2352.69
> 02 shift -> X² = 5552.95
> 03 shift -> X² = 1993.36
> 04 shift -> X² = 2027.77
> 05 shift -> X² = 178.91
> 06 shift -> X² = 2519.06
> 07 shift -> X² = 241.65
> 08 shift -> X² = 7.21
> 09 shift -> X² = 252.60
> 10 shift -> X² = 10238.76
> 11 shift -> X² = 801.14
> 12 shift -> X² = 12220.58
> 13 shift -> X² = 974.19
> 14 shift -> X² = 10449.89
> 15 shift -> X² = 1732.39
> 16 shift -> X² = 10270.59
> 17 shift -> X² = 143.86
> 18 shift -> X² = 2503.13
> 19 shift -> X² = 182.88
> 20 shift -> X² = 1636.62
> 21 shift -> X² = 86.74
> 22 shift -> X² = 515.50
> 23 shift -> X² = 572.04
> 24 shift -> X² = 9033.25
> 25 shift -> X² = 1231.51
[!] Min X² value for 'i': 7.211206
```

```
# Coset: kbvnpabnfnnprnefqpfefqnq
> 00 shift -> X² = 346.30
> 01 shift -> X² = 59.46
> 02 shift -> X² = 513.95
> 03 shift -> X² = 9215.23
> 04 shift -> X² = 938.34
> 05 shift -> X² = 3606.38
> 06 shift -> X² = 3884.81
> 07 shift -> X² = 7151.35
> 08 shift -> X² = 2159.24
> 09 shift -> X² = 6731.06
> 10 shift -> X² = 126.26
> 11 shift -> X² = 308.60
> 12 shift -> X² = 580.40
> 13 shift -> X² = 37.75
> 14 shift -> X² = 789.91
> 15 shift -> X² = 15088.58
> 16 shift -> X² = 1023.97
> 17 shift -> X² = 19660.63
> 18 shift -> X² = 4063.69
> 19 shift -> X² = 4381.36
> 20 shift -> X² = 4683.88
> 21 shift -> X² = 3836.27
> 22 shift -> X² = 213.50
> 23 shift -> X² = 558.81
> 24 shift -> X² = 137.07
> 25 shift -> X² = 521.60
[!] Min X² value for 'n': 37.748996
```

Figure 10. Cálculo X² para 3º e 4º caracteres da chave.

```
# Coset: libpejqkwzvsuhmilesemmws
> 00 shift -> X² = 1912.55
> 01 shift -> X² = 499.96
> 02 shift -> X² = 1986.23
> 03 shift -> X² = 1032.89
> 04 shift -> X² = 32.52
> 05 shift -> X² = 746.44
> 06 shift -> X² = 4023.25
> 07 shift -> X² = 291.48
> 08 shift -> X² = 5631.27
> 09 shift -> X² = 678.67
> 10 shift -> X² = 6918.42
> 11 shift -> X² = 1442.68
> 12 shift -> X² = 7965.95
> 13 shift -> X² = 624.40
> 14 shift -> X² = 4211.32
> 15 shift -> X² = 430.07
> 16 shift -> X² = 3788.50
> 17 shift -> X² = 709.16
> 18 shift -> X² = 506.37
> 19 shift -> X² = 557.68
> 20 shift -> X² = 6053.06
> 21 shift -> X² = 302.30
> 22 shift -> X² = 4213.55
> 23 shift -> X² = 751.52
> 24 shift -> X² = 5471.76
> 25 shift -> X² = 910.00
[!] Min X² value for 'e': 32.515637
```

```
# Coset: wlhxuktbbbtkntxtnhyffjyx
> 00 shift -> X² = 3168.14
> 01 shift -> X² = 4088.26
> 02 shift -> X² = 1798.60
> 03 shift -> X² = 4657.09
> 04 shift -> X² = 278.47
> 05 shift -> X² = 3771.16
> 06 shift -> X² = 94.15
> 07 shift -> X² = 1719.82
> 08 shift -> X² = 178.96
> 09 shift -> X² = 6678.61
> 10 shift -> X² = 473.30
> 11 shift -> X² = 2168.14
> 12 shift -> X² = 1994.20
> 13 shift -> X² = 2845.02
> 14 shift -> X² = 2710.61
> 15 shift -> X² = 2154.53
> 16 shift -> X² = 133.97
> 17 shift -> X² = 3602.35
> 18 shift -> X² = 289.47
> 19 shift -> X² = 26.02
> 20 shift -> X² = 180.66
> 21 shift -> X² = 7546.85
> 22 shift -> X² = 841.78
> 23 shift -> X² = 7547.43
> 24 shift -> X² = 1004.84
> 25 shift -> X² = 4040.14
[!] Min X² value for 't': 26.018997
```

Figure 11. Cálculo X² para 5º e 6º caracteres da chave.

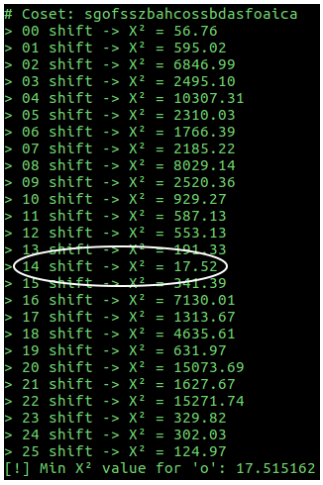

Figure 12. Cálculo X² para 7º caracter da chave.

Sendo assim, considerando os valores dos menores X² para cada caracter da possível chave e seu respectivo *shift*, é possível definir o melhor candidato para a chave:

- 1ª subamostra → X² mínimo para **06 shifts** → ***g***
- 2ª subamostra → X² mínimo para **20 shifts** → ***u***
- 3ª subamostra → X² mínimo para **08 shifts** → ***i***
- 4ª subamostra → X² mínimo para **13 shifts** → ***n***
- 5ª subamostra → X² mínimo para **04 shifts** → ***e***
- 6ª subamostra → X² mínimo para **19 shifts** → ***t***
- 7ª subamostra → X² mínimo para **14 shifts** → ***o***

Por fim, a chave é obtida: **guineto**. A decodificação da mensagem encriptada é verificada na figura 13.

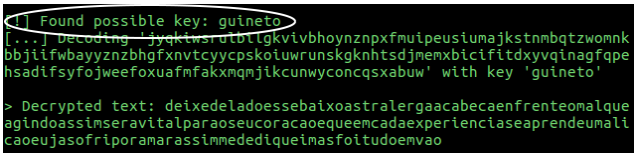

Figure 13. Resultado da decodificação da mensagem com a chave 'guineto'.

Aplicando os devidos espaçamentos, a mensagem em claro é dada por:

Conselho

Deixe de lado esse baixo astral
Erga a cabeça enfrente o mal
Que agindo assim será vital
Para o seu coração
É que em cada experiência
Se aprende uma lição
Eu já sofri por amar assim
Me dediquei, mas foi tudo em vão

Almir Guineto

De maneira análoga para a língua inglesa, considerando uma mensagem cifrada **'fhvzpiehmmnpdlzvdtuoeazncdseaxmdwsoodedqaahpahaztljyhbgtyulrbuifyyigmfxfreimoiywedakqzvvqsamvnzzzeyunamnbrghgmfccweuazdfutargwaruahpvfhpoebwwbneheeawqtktxysmwqryanlvqqgbxadexwtvtwogsviaifvasefnwwehmupjcxgbxnqeubzjrvudekppzrlsztrvjbulnsedjlzedseieosqqgeziebsmzldplvwqbratmlcbsmyrrqmzxdczjezeiiewevoztymtvghrzekbpvqwodegmlbcuepewqymqfhgnbalaahcqsjicgzdkunxbsqfwhqfzzdbguuqgvvpznwodoebsmvqtqremeqgxsqsrltkglozaigznbyedlrbtvjrrpstwxjvqepwzbsiudnpfltznzrdqljmybrqcqskzfkgxrqskwrmahrmtvtzzrpibsluhpvfhxofsdzrdsanrjwmgkeseemcighdxoimxqcvuyijbsmehfarviwenbsrrvmqzbprqpvbtbvrnunamnbrghgmfccweqozcyicipwedijbtkjrrpsvbn'** em **inglês**, a quebra da cifra pode ser vista na figura 14.

Figure 14. Teste de quebra de cifra

É possível verificar que, de maneira inicial, o algoritmo determinou o melhor candidato para tamanho da chave como sendo 9. Sendo assim, foi calculada a possível chave de tamanho 9 e foi obtido **'marilinda'**. Por fim, aplicando a decodificação da mensagem encriptada com a chave obtida, tem-se a seguinte mensagem:

'therearemanyvariationsofpassagesofloremipsumavailablebuthmajorityhavesufferedalterationinsomeformbyinjectedhumourorrandomisedwordswhichdontlookevenslightlybelievableifyouaregoingtouseapassageoflore

mipsumyouneedtobesurethereisntanythingembarrassinghiddeninthemiddleoftextalltheloremipsumgeneratorsontheinternettendtorepeatpredefinedchunksasnecessarymakingthisthefirsttruegeneratorontheinternetitusesadictionaryofoverlatinwordscombinedwithahandfulofmodelsentencestructurestogenerateloremipsumwhichlooksreasonablethegeneratedloremipsumisthereforealwaysfreefromrepetitioninjectedhumourornoncharacteristicwordsetc'

Aplicando os devidos espaçamentos na mensagem decodificada, é possível recuperar a mensagem original.

4. Discussão e Conclusões

A implementação do algoritmo foi bem sucedida, foi possível cifrar e decifrar mensagens em inglês e português com e sem o uso da palavra-chave a partir dos métodos de frequência e índice de coincidência apresentados anteriormente.

A abordagem escolhida se mostrou muito eficaz para os casos em que as mensagens possuem frequência parecida com seu respectivo idioma, o que geralmente acontece em textos longos e sem muitas repetições de palavras, mas este resultado pode variar caso a mensagem escolhida seja uma tentativa forçada de burlar essa frequência com muitas repetições de palavras ou a escolha de idiomas diferentes do que está sendo testado, por exemplo.

Comparando os resultados previstos pela teoria com os obtidos, chegamos à conclusão de que todos os testes feitos alcançaram resultados positivos, isto pode ser confirmado anteriormente em ”resultados”, onde todos os criptogramas testados foram decifrados com sucesso, seja com ou sem o uso da chave.

Com a conclusão deste projeto os conceitos passados em sala de aula como técnicas de segurança computacional foram reforçados, abordando a cifra de Vigenère enquanto que novos conceitos foram introduzidos e causaram questionamento, acarretando em muita pesquisa e aprendizado.

5. Referências

1. Slides de Aula
2. YOUTUBE: Frequency analysis to crack Vigenere 1
3. YOUTUBE: Breaking the Vigenere using frequency analisis
4. YOUTUBE: Brute Force Attack - Vigenère Cipher
5. YOUTUBE: Vigenere Cipher - Decryption (Known Key)
6. YOUTUBE: Vigenere Cipher - Decryption (Unknown Key)
7. YOUTUBE: Using the Index of Coincidence to Determine Vigenère Keyword Length
8. YOUTUBE: Index of Coincidence
9. YOUTUBE: Cryptanalysis of Vigenere cipher: not just how, but why it works