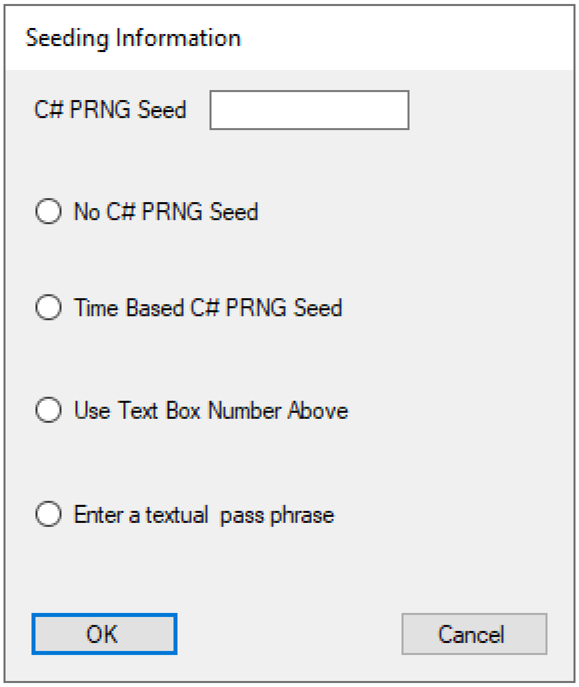
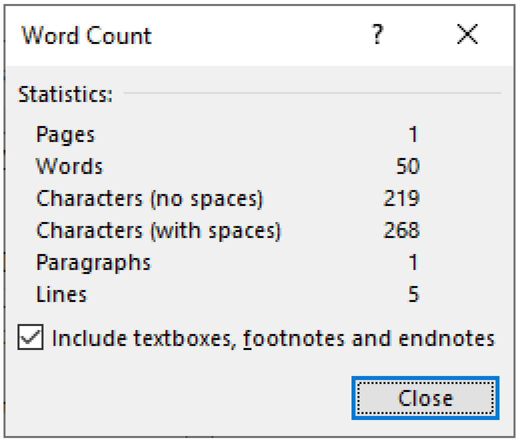
The Advanced Encryption Standard (AES) became the Golden Chalice or Gold Standard for a non-military grade encryption algorithm for the federal system at the confidential level of security in 2001. Before that time from the late 1970s to the early 2000s the Data Encryption Standard (DES) and Triple-DES were heavily used in the private and parts of the public sector. All the lower level of security encryption algorithms are certified by the National Institutes of Standards and Technology (NIST) Department of the United States government. Typically, these algorithms are also certified by the National Security Agency (NSA) which is known as the puzzle palace. The NSA is responsible for military grade encryption algorithms.

I have encoded DES and triple-DES algorithms in three languages beginning in 1996. These languages are C, C++, and C#. I was late in developing AES. I developed the algorithm in the C# computer language in 2009. I later revisited the algorithm with a triple-AES version of the algorithm in around 2018. This version used an encryption-decryption-encryption (EDE) variant of a multiple pass encryption algorithm. Triple passes allow key-lengths of 3 * 128 = 384 or 3 * 192 = 576 or 3 * 256 = 768 bits. Compare triple-AES with at most 768-bits versus triple-DES with a typical 168 bits. I created a triple-DES stream cipher encryption algorithm to offer nearly one-time pad level of security.

The rest of this text consists of screenshots of my C# Triple-AES test library application with a couple of Microsoft Word application pictures.


Seeding Information
C# PRNG Seed
○ No C# PRNG Seed
○ Time Based C# PRNG Seed
○ Use Text Box Number Above
○ Enter a textual pass phrase
OK
Cancel

“This is a textual pass phrase of at least one hundred and forty-seven characters. That is a pretty long pass phrase as far as pass phrases are concerned. Glad to have Word to perform a character count. Word did not like my usage of pretty long in my previous sentence.” Word hates flowery language.

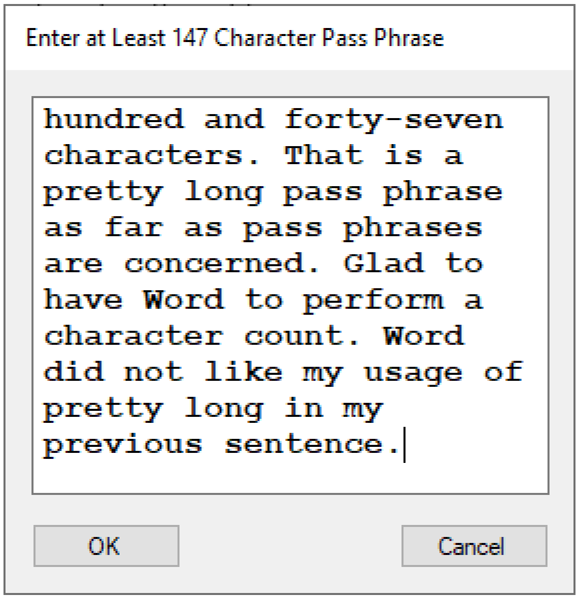


Word Count

Statistics:

Pages	1
Words	50
Characters (no spaces)	219
Characters (with spaces)	268
Paragraphs	1
Lines	5

☑ Include textboxes, footnotes and endnotes



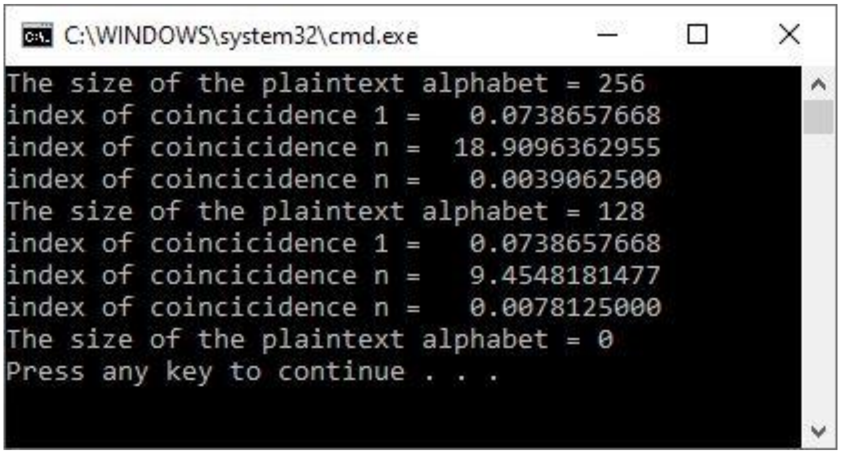
Enter at Least 147 Character Pass Phrase

```
hundred and forty-seven
characters. That is a
pretty long pass phrase
as far as pass phrases
are concerned. Glad to
have Word to perform a
character count. Word
did not like my usage of
pretty long in my
previous sentence.
```

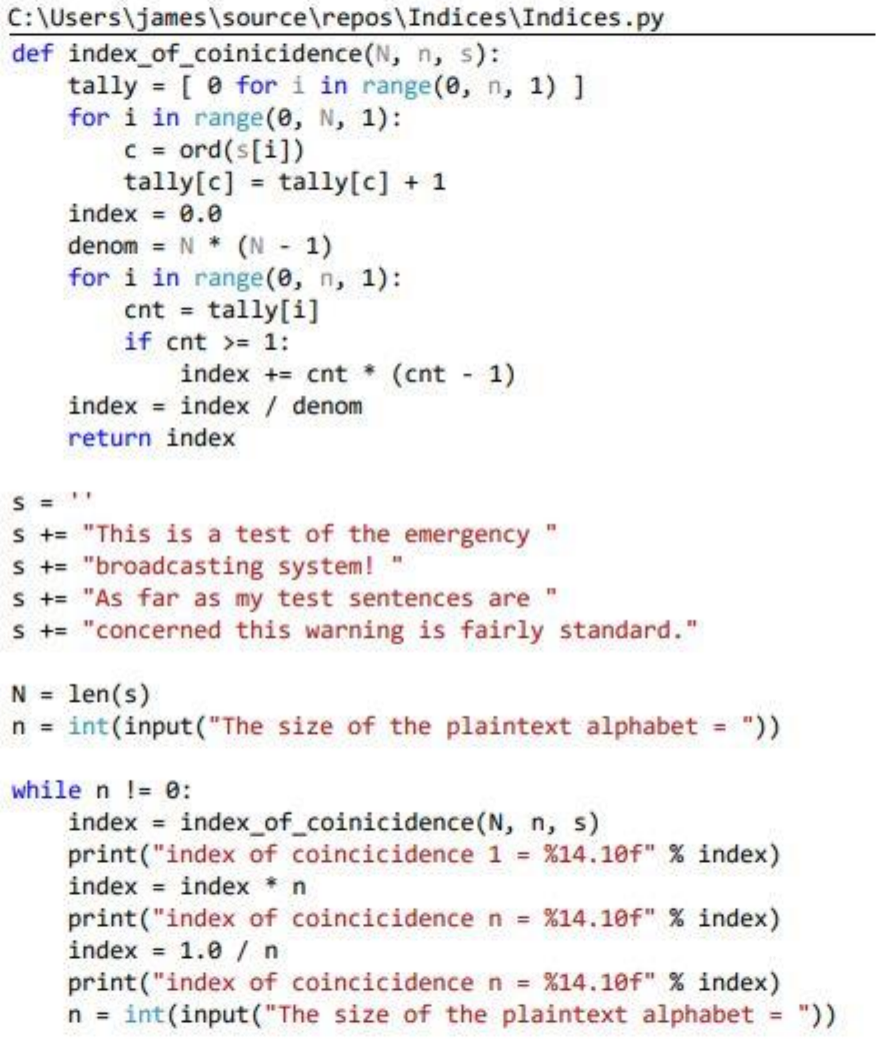
“This is a test of the emergency broadcasting system! As far as my test sentences are concerned this warning is fairly standard.”

Online calculator: Index of Coincidence (planetcalc.com)

From the calculator website the index of coincidence is 0.0749. I compute 0.07387. My ciphertext has an index of coincidence of 0.0040. The normalized indices of coincidence are 18.9096362955 for an alphabet size of 256 characters or 9.4548181477 for a length of 128 ASCII encoded characters.



```
The size of the plaintext alphabet = 256
index of coincicidence 1 =    0.0738657668
index of coincicidence n =   18.9096362955
index of coincicidence n =    0.0039062500
The size of the plaintext alphabet = 128
index of coincicidence 1 =    0.0738657668
index of coincicidence n =    9.4548181477
index of coincicidence n =    0.0078125000
The size of the plaintext alphabet = 0
Press any key to continue . . .
```

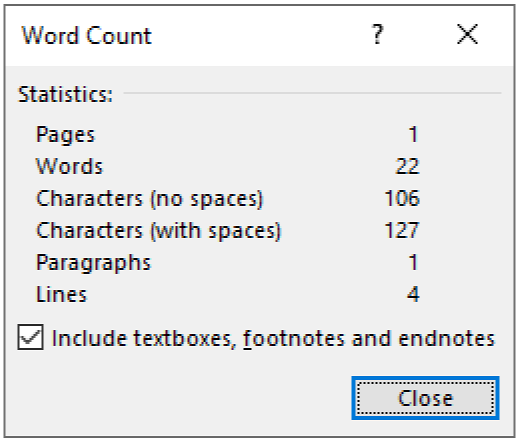

C:\Users\james\source\repos\Indices\Indices.py

```python
def index_of_coinicidence(N, n, s):
    tally = [ 0 for i in range(0, n, 1) ]
    for i in range(0, N, 1):
        c = ord(s[i])
        tally[c] = tally[c] + 1
    index = 0.0
    denom = N * (N - 1)
    for i in range(0, n, 1):
        cnt = tally[i]
        if cnt >= 1:
            index += cnt * (cnt - 1)
    index = index / denom
    return index

s = ''
s += "This is a test of the emergency "
s += "broadcasting system! "
s += "As far as my test sentences are "
s += "concerned this warning is fairly standard."

N = len(s)
n = int(input("The size of the plaintext alphabet = "))

while n != 0:
    index = index_of_coinicidence(N, n, s)
    print("index of coincicidence 1 = %14.10f" % index)
    index = index * n
    print("index of coincicidence n = %14.10f" % index)
    index = 1.0 / n
    print("index of coincicidence n = %14.10f" % index)
    n = int(input("The size of the plaintext alphabet = "))
```

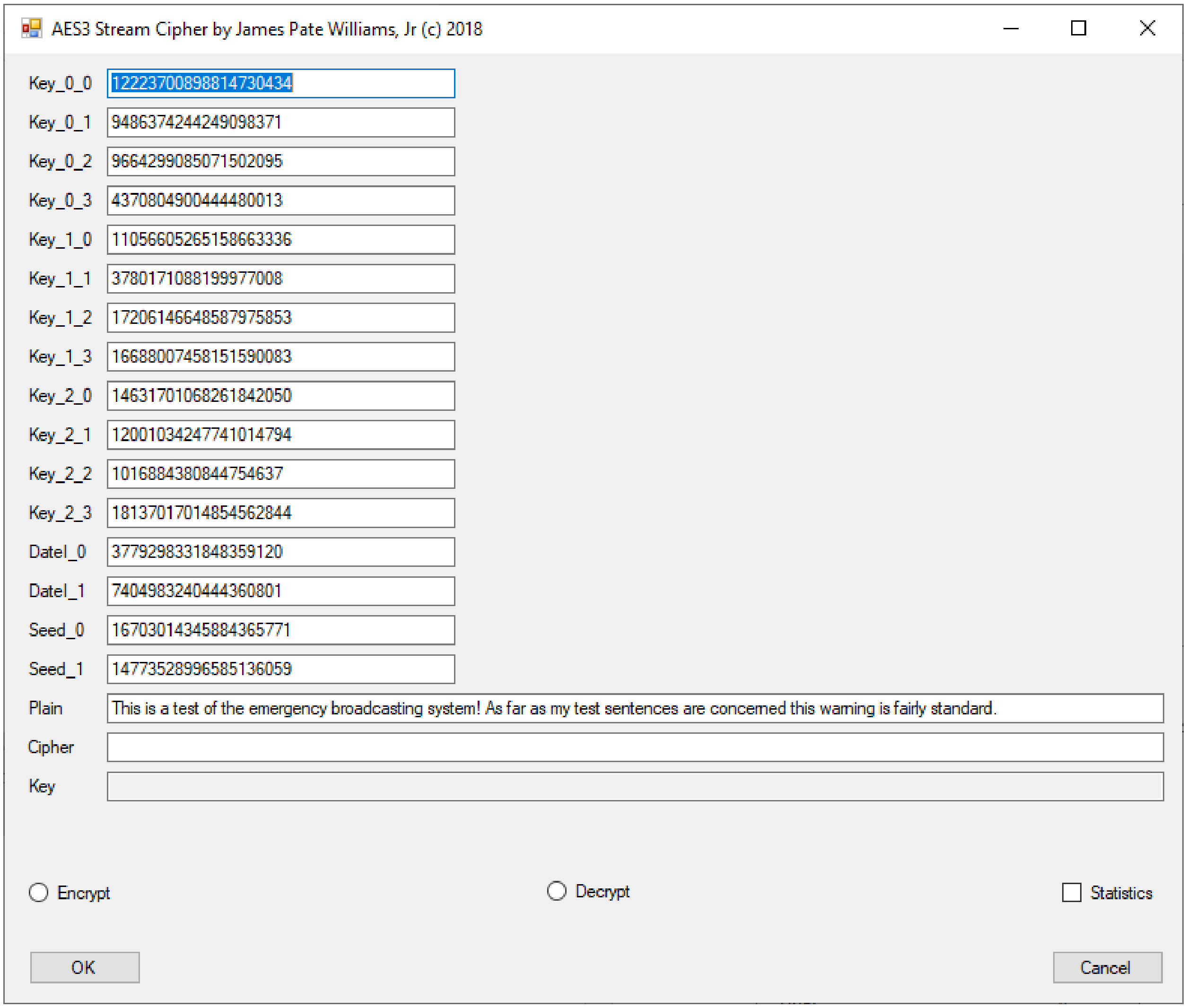


Word Count

Statistics:

Pages	1
Words	22
Characters (no spaces)	106
Characters (with spaces)	127
Paragraphs	1
Lines	4

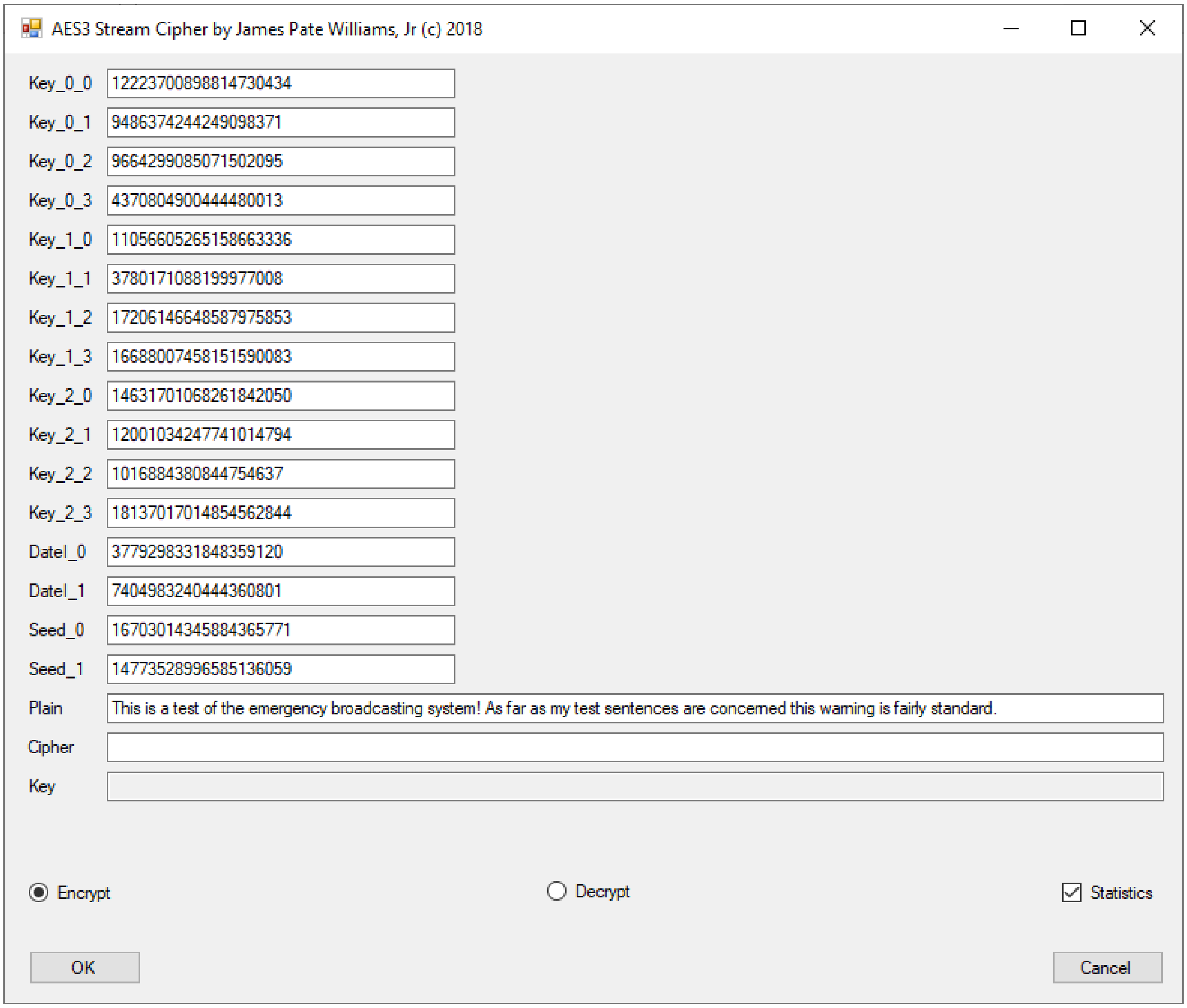
☑ Include textboxes, footnotes and endnotes



AES3 Stream Cipher by James Pate Williams, Jr (c) 2018

Key_0_0	12223700898814730434
Key_0_1	9486374244249098371
Key_0_2	9664299085071502095
Key_0_3	4370804900444480013
Key_1_0	11056605265158663336
Key_1_1	3780171088199977008
Key_1_2	17206146648587975853
Key_1_3	16688007458151590083
Key_2_0	14631701068261842050
Key_2_1	12001034247741014794
Key_2_2	1016884380844754637
Key_2_3	18137017014854562844
DateI_0	3779298331848359120
DateI_1	7404983240444360801
Seed_0	16703014345884365771
Seed_1	14773528996585136059
Plain	This is a test of the emergency broadcasting system! As far as my test sentences are concerned this warning is fairly standard.
Cipher	
Key	

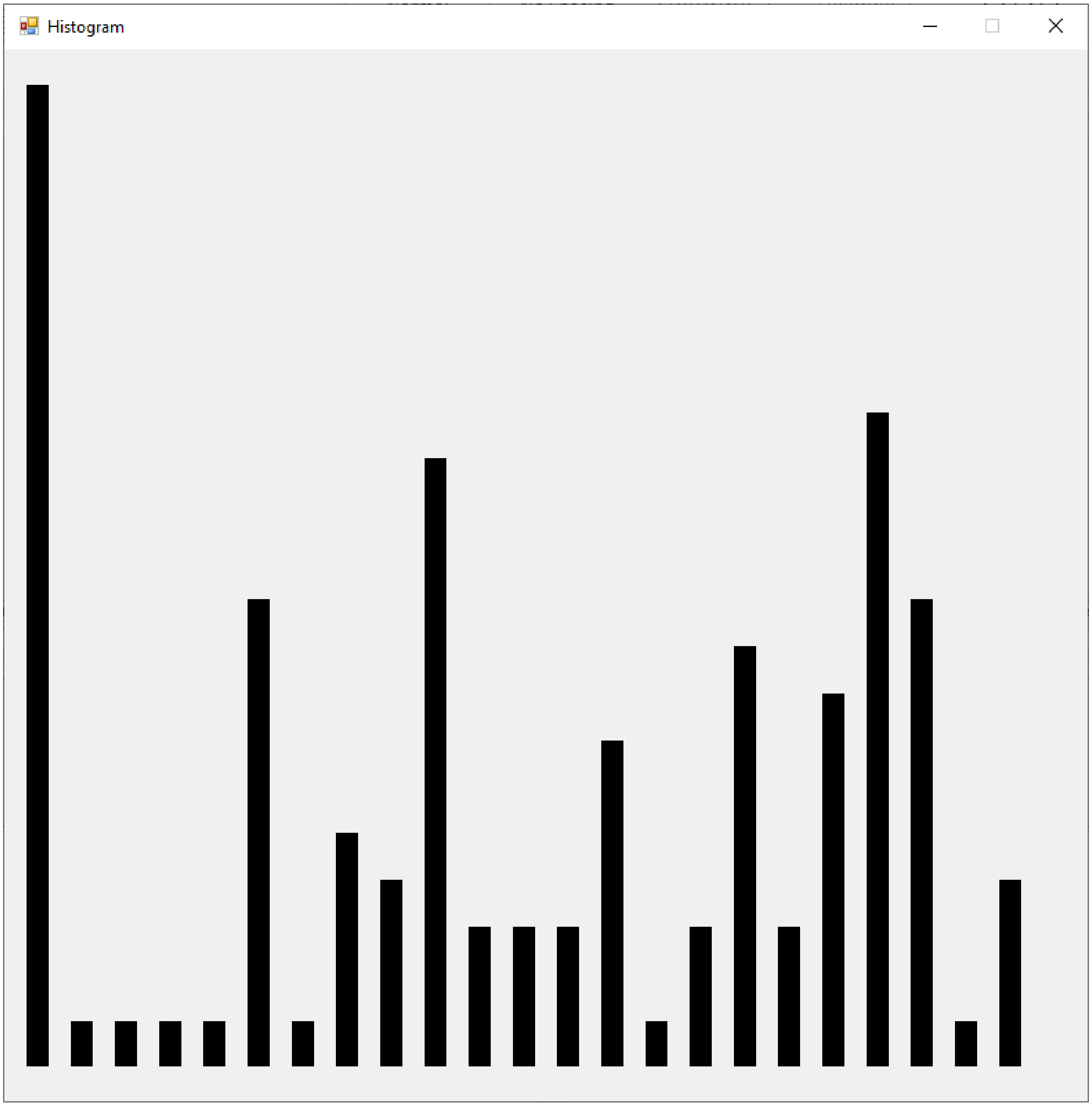
○ Encrypt ○ Decrypt □ Statistics

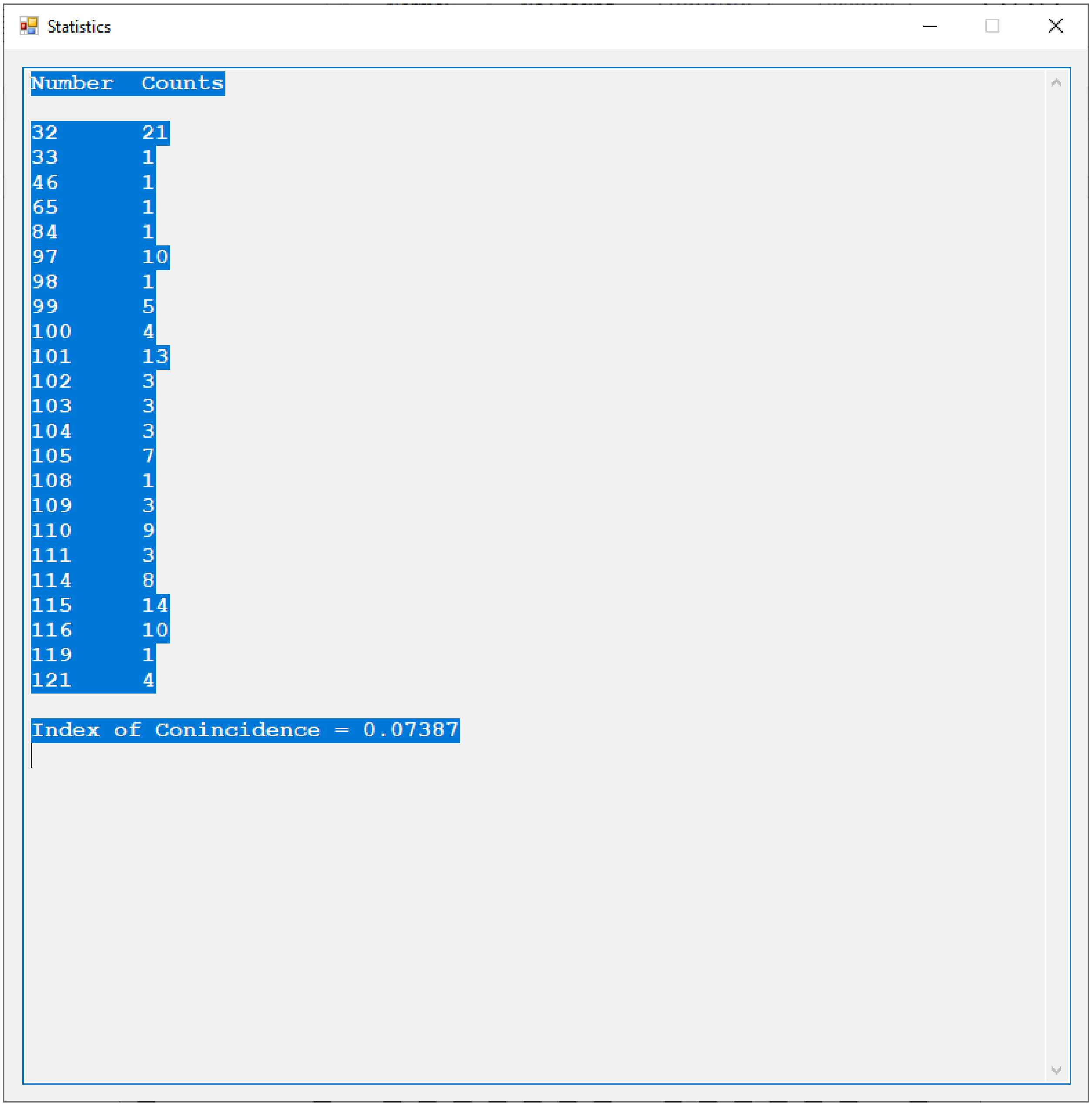


AES3 Stream Cipher by James Pate Williams, Jr (c) 2018

Key_0_0	12223700898814730434
Key_0_1	9486374244249098371
Key_0_2	9664299085071502095
Key_0_3	4370804900444480013
Key_1_0	11056605265158663336
Key_1_1	3780171088199977008
Key_1_2	17206146648587975853
Key_1_3	16688007458151590083
Key_2_0	14631701068261842050
Key_2_1	12001034247741014794
Key_2_2	1016884380844754637
Key_2_3	18137017014854562844
DateI_0	3779298331848359120
DateI_1	7404983240444360801
Seed_0	16703014345884365771
Seed_1	14773528996585136059
Plain	This is a test of the emergency broadcasting system! As far as my test sentences are concerned this warning is fairly standard.
Cipher	
Key	

● Encrypt ○ Decrypt ☑ Statistics

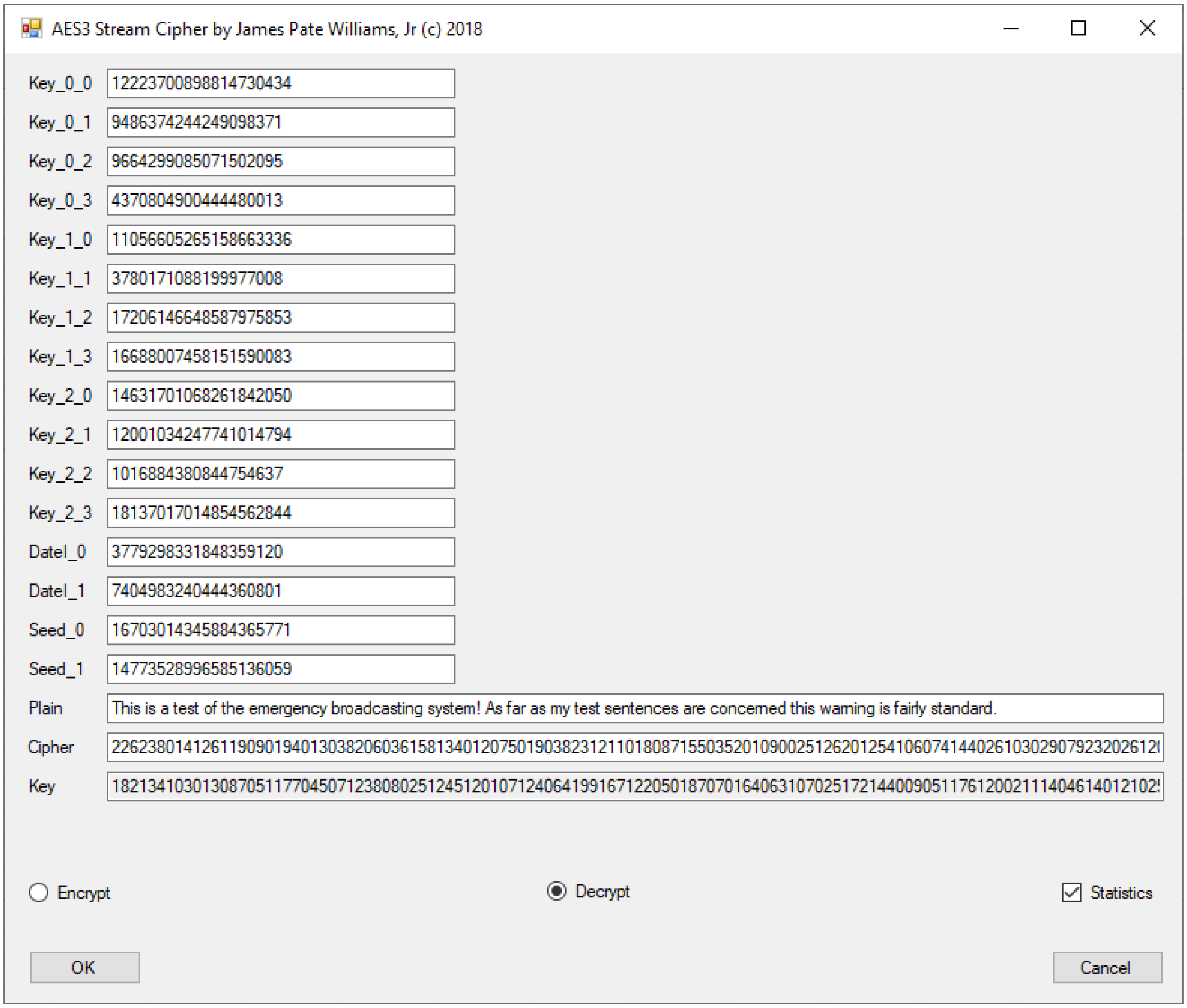

Histogram



```
Number  Counts

32      21
33      1
46      1
65      1
84      1
97      10
98      1
99      5
100     4
101     13
102     3
103     3
104     3
105     7
108     1
109     3
110     9
111     3
114     8
115     14
116     10
119     1
121     4

Index of Conincidence = 0.07387
```



AES3 Stream Cipher by James Pate Williams, Jr (c) 2018

Key_0_0 12223700898814730434

Key_0_1 9486374244249098371

Key_0_2 9664299085071502095

Key_0_3 4370804900444480013

Key_1_0 11056605265158663336

Key_1_1 3780171088199977008

Key_1_2 17206146648587975853

Key_1_3 16688007458151590083

Key_2_0 14631701068261842050

Key_2_1 12001034247741014794

Key_2_2 1016884380844754637

Key_2_3 18137017014854562844

Datel_0 3779298331848359120

Datel_1 7404983240444360801

Seed_0 16703014345884365771

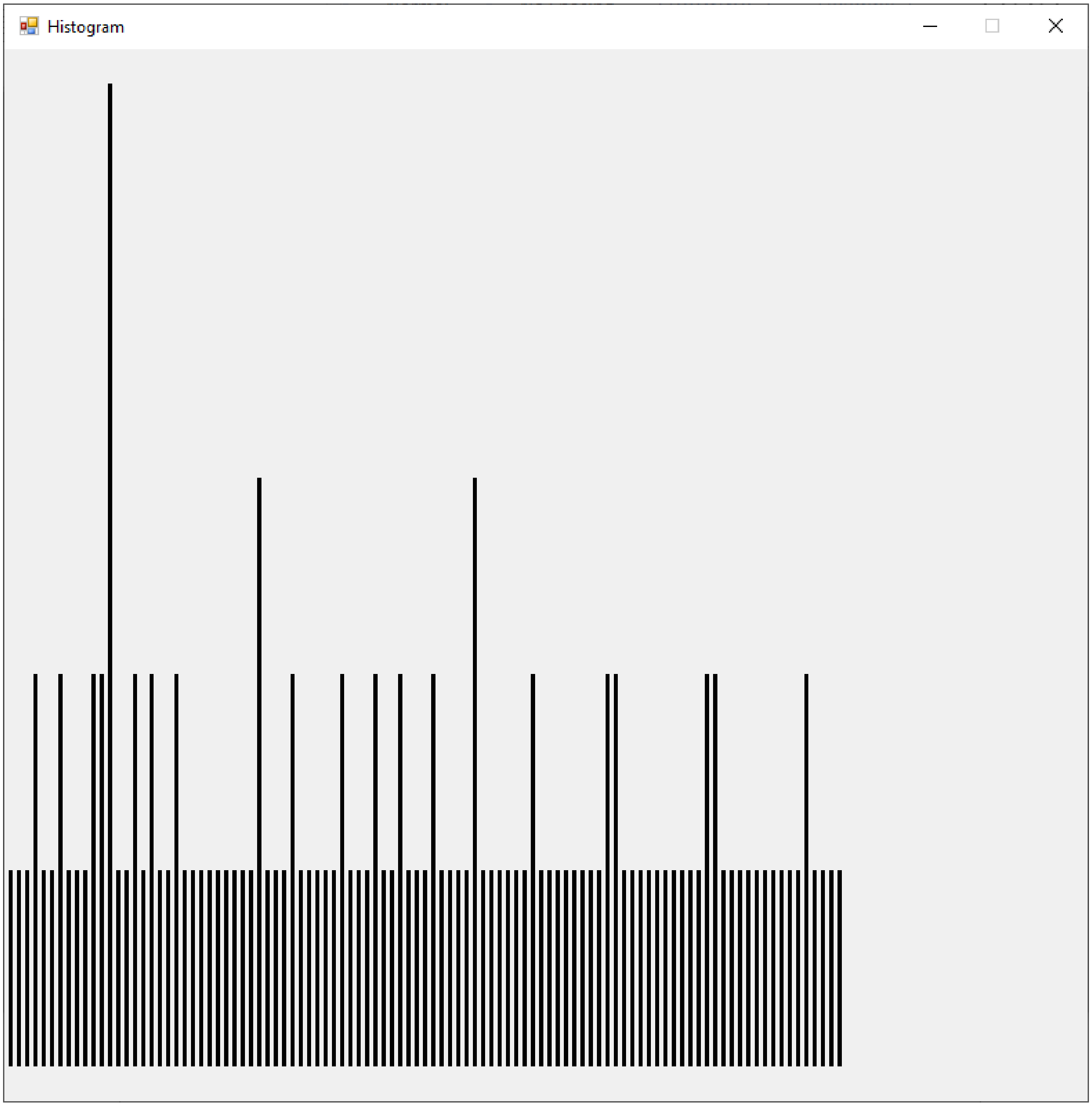
Seed_1 14773528996585136059

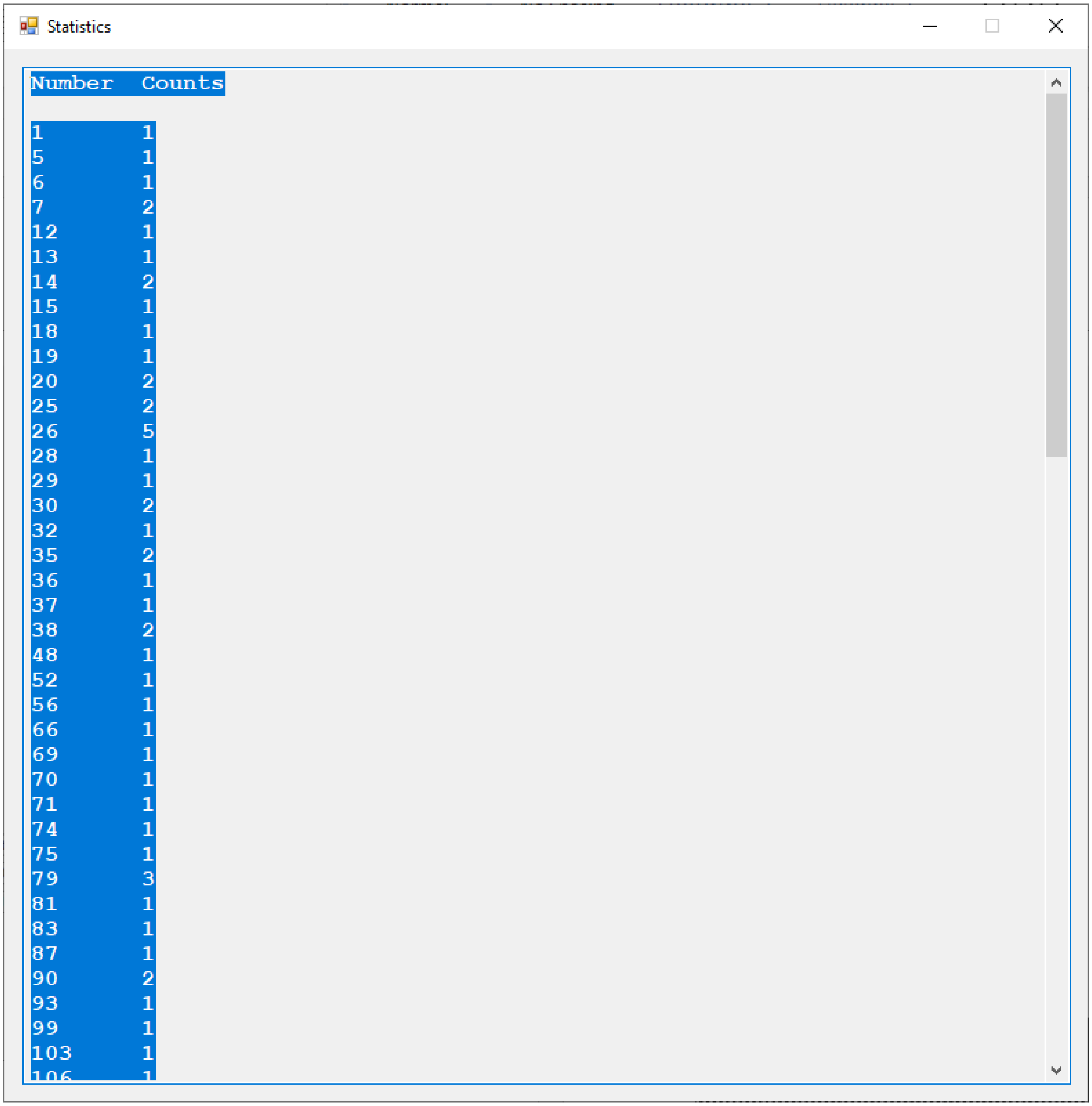
Plain This is a test of the emergency broadcasting system! As far as my test sentences are concerned this warning is fairly standard.

Cipher 22623801412611909019401303820603615813401207501903823121101808715503520109002512620125410607414402610302907923202612

Key 18213410301308705117704507123808025124512010712406419916712205018707016406310702517214400905117612002111404614012102

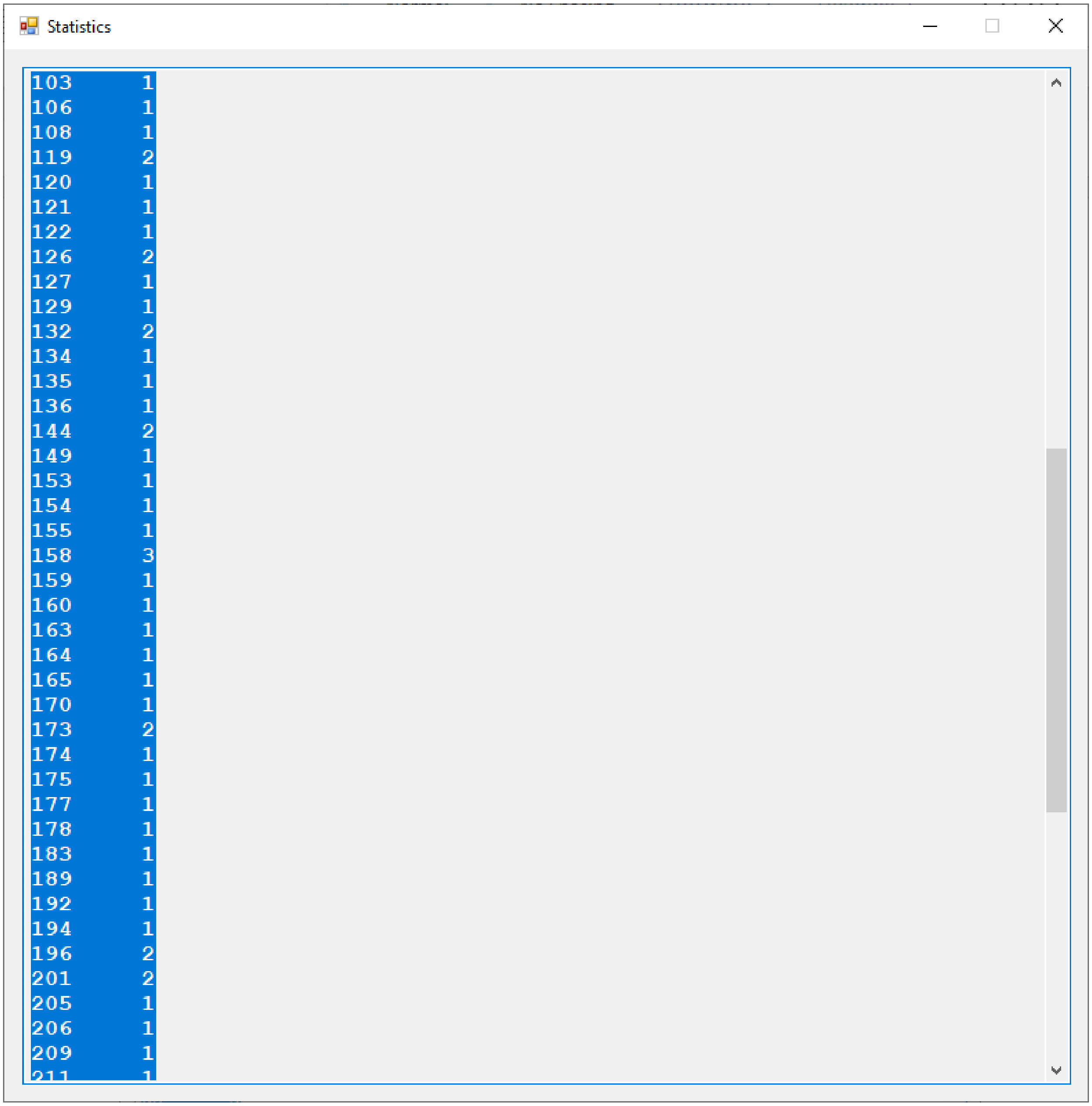
○ Encrypt ● Decrypt ☑ Statistics


Histogram

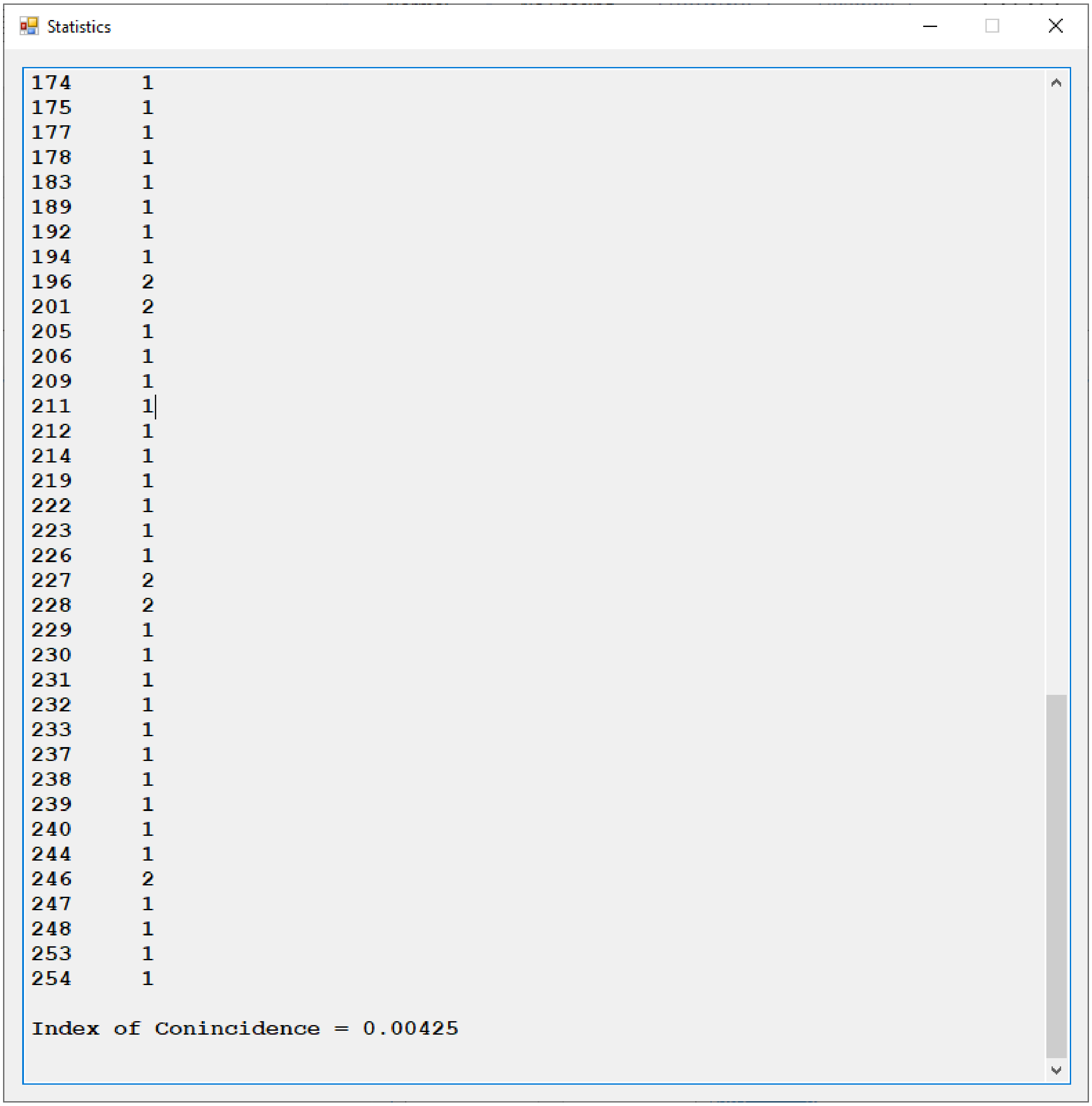


```
Number  Counts

1           1
5           1
6           1
7           2
12          1
13          1
14          2
15          1
18          1
19          1
20          2
25          2
26          5
28          1
29          1
30          2
32          1
35          2
36          1
37          1
38          2
48          1
52          1
56          1
66          1
69          1
70          1
71          1
74          1
75          1
79          3
81          1
83          1
87          1
90          2
93          1
99          1
103         1
106         1
```



```
103        1
106        1
108        1
119        2
120        1
121        1
122        1
126        2
127        1
129        1
132        2
134        1
135        1
136        1
144        2
149        1
153        1
154        1
155        1
158        3
159        1
160        1
163        1
164        1
165        1
170        1
173        2
174        1
175        1
177        1
178        1
183        1
189        1
192        1
194        1
196        2
201        2
205        1
206        1
209        1
211        1
```



```
174      1
175      1
177      1
178      1
183      1
189      1
192      1
194      1
196      2
201      2
205      1
206      1
209      1
211      1
212      1
214      1
219      1
222      1
223      1
226      1
227      2
228      2
229      1
230      1
231      1
232      1
233      1
237      1
238      1
239      1
240      1
244      1
246      2
247      1
248      1
253      1
254      1

Index of Conincidence = 0.00425
```