

Создаем файл caesar.py и реализовываем функции шифрования и расшифровывания числа и последовательности чисел:

```
1  def encrypt(m, key):
2      return (m + key) % 256
3
4  def decrypt(m, key):
5      return (m - key) % 256
6
7  def encrypt_data(data, key):
8      cypher_data = []
9      for m in data:
10         cypher_data.append(encrypt(m, key))
11     return cypher_data
12
13 def decrypt_data(data_c, key):
14     data = []
15     for c in data_c:
16         data.append(decrypt(c, key))
17     return data
```

Рис. 1. Файл caesar.py

Пример работы шифра Цезаря:

```
m = 24      # сообщение
key = 37    # ключ
c = caesar.encrypt(m, key)
print('c =', c)
m1 = caesar.decrypt(c, key)
print('m1 =', m1)
data = [34, 67, 123, 79, 201]
encrypt_data = caesar.encrypt_data(data, key)
print('encrypt_data =', encrypt_data)
decrypt_data = caesar.decrypt_data(encrypt_data, key)
print('decrypt_data =', decrypt_data)
```

```
c = 61
m1 = 24
encrypt_data = [71, 104, 160, 116, 238]
decrypt_data = [34, 67, 123, 79, 201]
```

Рис. 2. Пример работы шифра Цезаря

Формируем файл 'f1.txt':

He was a burly man of an exceedingly dark complexion, with an exceedingly large head and a corresponding large hand. He took my chin in his large hand and turned up my face to have a look at me by the light of the candle. He was prematurely bald on the top of his head, and had bushy black eyebrows that wouldn't lie down but stood up bristling. His eyes were set very deep in his head, and were disagreeably sharp and suspicious. He had a large watchchain, and strong black dots where his beard and whiskers would have been if he had let them. He was nothing to me, and I could have had no foresight then, that he ever would be anything to me, but it happened that I had this opportunity of observing him well.

Теперь зашифруем весь файл 'f1.txt'. Результат шифрования запишем в файл 'f1\_encrypt.txt'.

```
data = read_write_file.read_data_1byte('data/f1.txt')
encrypt_data = caesar.encrypt_data(data, key=67)
txt = ''.join([chr(s) for s in encrypt_data])
read_write_file.write_data_1byte('encrypt_data/f1_encrypt.txt', encrypt_data)
```

Рис. 3. Результат шифрования шифром Цезаря с ключом 67

Расшифруем обратно файл 'f1\_encrypt.txt', используя известный ключ.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/f1_encrypt.txt')
decrypt_data = caesar.decrypt_data(encrypt_data, key=67)
txt = ''.join([chr(s) for s in decrypt_data])
read_write_file.write_data_1byte('decrypt_data/f1_decrypt_know_key.txt', decrypt_data)
```

---

```
1 | He was a burly man of an exceedingly dark complexion, with an exceedingly large head and a
2 | corresponding large hand. He took my chin in his large hand and turned up my face to have a
3 | look at me by the light of the candle. He was prematurely bald on the top of his head, and had
4 | bushy black eyebrows that wouldn't lie down but stood up bristling. His eyes were set very deep
5 | in his head, and were disagreeably sharp and suspicious. He had a large watchchain, and strong
6 | black dots where his beard and whiskers would have been if he had let them. He was nothing to
7 | me, and I could have had no foresight then, that he ever would be anything to me, but it
8 | happened that I had this opportunity of observing him well.
```

Рис. 4. Результат расшифровки

Расшифруем файл 'f1\_encrypt', будем считать, что ключ нам неизвестен.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/fl_encrypt.txt')

for k in range(256):
    decrypt_data = caesar.decrypt_data(encrypt_data, key=k)
    txt = ''.join([chr(s) for s in decrypt_data])
    is_english = detectEnglish.isEnglish(txt)
    if is_english:
        print('key =', k)
        print('decrypt_data =', txt)
        break

read_write_file.write_data_1byte('decrypt_data/fl_decrypt_unknown_key.txt', decrypt_data)

key = 67
decrypt_data = He was a burly man of an exceedingly dark complexion, with an exceed
corresponding large hand. He took my chin in his large hand and turned up my face t
look at me by the light of the candle. He was prematurely bald on the top of his he
bushy black eyebrows that wouldn't lie down but stood up bristling. His eyes were s
in his head, and were disagreeably sharp and suspicious. He had a large watchchain,
black dots where his beard and whiskers would have been if he had let them. He was
me, and I could have had no foresight then, that he ever would be anything to me, b
happened that I had this opportunity of observing him well.
```

Рис. 5. Результат расшифровки, используя перебор ключа

Возьмем файл 'f2.png' (Рис. 6).



Рис. 6.

Зашифруем изображение.

```
data = read_write_file.read_data_1byte('data/f2.png')
encrypt_data = caesar.encrypt_data(data, key=143)
read_write_file.write_data_1byte('encrypt_data/f2_encrypt.png', encrypt_data)
```

Теперь считаем, что ключ неизвестен, расшифруем 'f2\_encrypt.png'.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/f2_encrypt.png')
for k in range(256):
    decrypt_data = caesar.decrypt_data(encrypt_data, key=k)
    if decrypt_data[0] == 0x89 and decrypt_data[1] == 0x50:
        print('key =', k)
        break
read_write_file.write_data_1byte('decrypt_data/f2_decrypt.png', decrypt_data)
```

```
key = 143
```

Рис. 7. Результат расшифровки 'f2\_encrypt.png'

Как видно, получили тот же ключ.

Дешифруем файл 't3\_caesar\_c\_all.txt'.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/t3_caesar_c_all.txt')
for k in range(256):
    decrypt_data = caesar.decrypt_data(encrypt_data, key=k)
    txt = ''.join([chr(s) for s in decrypt_data])
    is_english = detectEnglish.isEnglish(txt)
    if is_english:
        print('key =', k)
        print('decrypt_data =', txt)
        read_write_file.write_data_1byte('decrypt_data/t3_caesar_c_all_decrypt.txt', decrypt_data)
        break
```

Рис. 8.

Результат:

key = 135

```
1 "Poor soul!" Camilla presently went on (I knew they had all been looking at me in the mean time), "he is so very
2 strange! Would anyone believe that when Tom's wife died, he actually could not be induced to see the importance of the
3 children's having the deepest of trimmings to their mourning? 'Good Lord!' says he, 'Camilla, what can it signify so
4 long as the poor bereaved little things are in black?' So like Matthew! The idea!"
5 "Good points in him, good points in him," said Cousin Raymond; "Heaven forbid I should deny good points in him; but he
6 never had, and he never will have, any sense of the proprieties."
7 "You know I was obliged," said Camilla, "I was obliged to be firm. I said, 'It WILL NOT DO, for the credit of the
8 family.' I told him that, without deep trimmings, the family was disgraced. I cried about it from breakfast till
9 dinner. I injured my digestion. And at last he flung out in his violent way, and said, with a D, 'Then do as you like.'
10 Thank Goodness it will always be a consolation to me to know that I instantly went out in a pouring rain and bought the
11 things."
12 "He paid for them, did he not?" asked Estella.
13 "It's not the question, my dear child, who paid for them," returned Camilla. "I bought them. And I shall often think of
14 that with peace, when I wake up in the night."
15 The ringing of a distant bell, combined with the echoing of some cry or call along the passage by which I had come,
16 interrupted the conversation and caused Estella to say to me, "Now, boy!" On my turning round, they all looked at me
17 with the utmost contempt, and, as I went out, I heard Sarah Pocket say, "Well I am sure! What next!" and Camilla add,
18 with indignation, "Was there ever such a fancy! The i-de-a!"
19 As we were going with our candle along the dark passage, Estella stopped all of a sudden, and, facing round, said in
20 her taunting manner with her face quite close to mine:
21 "Well?"
```

Рис. 9.

Дешифруем файл 'c4\_caesar\_c\_all.bmp'.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/c4_caesar_c_all.bmp')
for k in range(256):
    decrypt_data = caesar.decrypt_data(encrypt_data, key=k)
    if decrypt_data[0] == 0x42 and decrypt_data[1] == 0x4d:
        print('key =', k)
        print('decrypt_data =', decrypt_data[:10])
        read_write_file.write_data_1byte('decrypt_data/c4_caesar_c_all_decrypt.bmp', decrypt_data)
        break
```

Результат:

key = 231



Рис. 10.

Зашифруем рис.10, оставив первые 50 байт без изменения.

```
data = read_write_file.read_data_1byte('decrypt_data/c4_caesar_c_all_decrypt.bmp')
encrypt_data = data[:50] + caesar.encrypt_data(data[50:], key=231)
read_write_file.write_data_1byte('encrypt_data/c4_caesar_c_all_encrypted_50.bmp', encrypt_data)
```



Рис. 11. Шифрование с ключом 231



Рис. 12. Шифрование с ключом 142



Реализуем модуль substitution.py.

```
k=[179, 109, 157, 182, 126, 141, 251, 220, 169,
 237, 188, 131, 207, 22, 32, 242, 208, 68, 216,
 170, 249, 199, 44, 198, 206, 8, 148, 197, 136,
 195, 159, 98, 175, 53, 123, 212, 233, 150, 6,
 243, 38, 79, 156, 153, 2, 134, 47, 215, 102,
 15, 57, 110, 236, 24, 184, 72, 137, 113, 171,
 70, 161, 64, 252, 247, 49, 103, 105, 138, 119,
 213, 87, 130, 203, 90, 167, 238, 231, 116, 78,
 86, 173, 250, 200, 239, 178, 97, 114, 94, 166,
 142, 104, 31, 75, 89, 106, 56, 128, 69, 164,
 67, 26, 228, 61, 181, 125, 227, 54, 96, 168,
 107, 17, 14, 37, 190, 219, 211, 121, 112, 35,
 18, 143, 158, 193, 129, 71, 23, 101, 191, 41,
 241, 82, 201, 223, 120, 59, 177, 58, 63, 151,
 42, 36, 183, 226, 127, 172, 202, 84, 132, 3,
 45, 73, 30, 235, 50, 189, 4, 1, 43, 221, 205,
 83, 232, 46, 147, 93, 192, 124, 244, 12, 21,
 80, 55, 160, 145, 245, 209, 88, 204, 176, 13,
 253, 11, 99, 165, 140, 19, 224, 111, 27, 185,
 65, 62, 16, 163, 210, 115, 217, 34, 92, 187,
 152, 155, 108, 5, 122, 229, 174, 118, 162, 95,
 100, 7, 66, 29, 230, 144, 149, 52, 9, 91, 117,
 214, 76, 48, 33, 194, 254, 10, 234, 218, 40,
 133, 196, 139, 135, 240, 60, 25, 225, 85, 255,
 246, 51, 28, 146, 74, 222, 186, 39, 77, 0, 20,
 180, 154, 81, 248]

def encrypt_data(data):
    cypher_data = []
    for d in data:
        cypher_data.append(k[d])
    return cypher_data

def decrypt_data(data):
    decypher_data = []
    for d in data:
        decypher_data.append(k.index(d))
    return decypher_data
```

И с помощью таблицы замен k расшифруем файл 'c3\_subst\_c\_all.png'.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/c3_subst_c_all.png')
for k in range(256):
    decrypt_data = substitution.decrypt_data(encrypt_data)
    if decrypt_data[0] == 0x89 and decrypt_data[1] == 0x50:
        print('k =', k)
        break
read_write_file.write_data_1byte('decrypt_data/c3_subst_c_all_decrypt.png', decrypt_data)
```

Результат:



Рис.13.

Теперь зашифруем рис. 13 оставив первые 350 байт без изменения.

```
data = read_write_file.read_data_1byte('decrypt_data/c3_subst_c_all_decrypt.png')
encrypt_data = data[:350] + substitution.encrypt_data(data[350:])
read_write_file.write_data_1byte('encrypt_data/c3_subst_c_all_encrypted_350.png', encrypt_data)
```

Результат:



Рис. 14.

Для аффинных шифров реализуем модуль affine.py. с алгоритмом Евклида, расширенным алгоритмом Евклида и функциями шифрования и дешифрования.

```
def gcd(a, b):
    # Return the GCD of a and b using Euclid's Algorithm
    while a != 0:
        a, b = b % a, a
    return b

def findModInverse(a, m):
    # Returns the modular inverse of a % m, which is
    # the number x such that a*x % m = 1

    if gcd(a, m) != 1:
        return None

    # Calculate using the Extended Euclidean Algorithm:
    u1, u2, u3 = 1, 0, a
    v1, v2, v3 = 0, 1, m
    while v3 != 0:
        q = u3 // v3 # // is the integer division operator
        v1, v2, v3, u1, u2, u3 = (u1 - q * v1), (u2 - q * v2), (u3 - q * v3), v1, v2, v3
    return u1 % m

def encrypt(m, a, b):
    return (a * m + b) % 256

def decrypt(m, a, b):
    return ((m - b) * findModInverse(a, 256)) % 256

def encrypt_data(data, a, b):
    cypher_data = []
    for m in data:
        cypher_data.append(encrypt(m, a, b))
    return cypher_data

def decrypt_data(data, a, b):
    decypher_data = []
    for m in data:
        decypher_data.append(decrypt(m, a, b))
    return decypher_data
```

Зашифруем файл 'f3.bmp' аффинным шифром  $a = 167$ ,  $b = 35$ .



Рис. 15. f3.bmp

```
data = read_write_file.read_data_1byte('data/f3.bmp')
a = 167
b = 35
encrypt_data = affine.encrypt_data(data, a, b)
read_write_file.write_data_1byte('encrypt_data/f3_affine_encrypt.bmp', encrypt_data)
```

Расшифруем 'f3\_affine\_encrypt.bmp'.

```
a = 167
b = 35
encrypt_data = read_write_file.read_data_1byte('encrypt_data/f3_affine_encrypt.bmp')
decrypt_data = affine.decrypt_data(encrypt_data, a, b)
read_write_file.write_data_1byte('decrypt_data/f3_affine_decrypt.bmp', decrypt_data)
```

Теперь зашифруем оставив первые 50 байт без изменения.

```
encrypt_data = decrypt_data[:50] + affine.encrypt_data(decrypt_data[50:], a, b)
read_write_file.write_data_1byte('encrypt_data/f3_affine_c_encrypted_50.png', encrypt_data)
```

Результат:



Рис. 16.



Дешифруем файл 'text10\_affine\_c\_all.txt'. Шифр аффинный, а и b неизвестны.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/text10_affine_c_all.txt')
count_keys = 0
is_english = False
for a in range(256):
    if affine.gcd(a, 256) == 1:
        for b in range(256):
            count_keys += 1
            decrypt_data = affine.decrypt_data(encrypt_data[:30], a, b)
            text = ''.join([chr(s) for s in decrypt_data[:30]])
            is_english = detectEnglish.isEnglish(text)
            if is_english:
                print('k =', count_keys)
                print('a =', a)
                print('b =', b)
                decrypt_data = affine.decrypt_data(encrypt_data, a, b)
                read_write_file.write_data_1byte('decrypt_data/text10_affine_c_all_decrypt.txt', decrypt_data)
                break
        if is_english:
            break
```

Результат:

3701 итерация, a = 29, b = 116.

```
k = 3701
a = 29
b = 116
```

```
1 For oft, when on my couch I lie
2 In vacant or in pensive mood,
3 They flash upon that inward eye
4 Which is the bliss of solitude;
5 And then my heart with pleasure fills,
6 And dances with the daffodils.
```

Дешифруем файл 'b4\_affine\_c\_all.png'. Шифр аффинный, а и b неизвестны.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/b4_affine_c_all.png')
count_keys = 0
is_PNG = False
for a in range(256):
    if affine.gcd(a, 256) == 1:
        for b in range(256):
            count_keys += 1
            decrypt_data = affine.decrypt_data(encrypt_data[:2], a, b)
            is_PNG = decrypt_data[0] == 0x89 and decrypt_data[1] == 0x50
            if is_PNG:
                print('k =', count_keys)
                print('a =', a)
                print('b =', b)
                decrypt_data = affine.decrypt_data(encrypt_data, a, b)
                read_write_file.write_data_1byte('decrypt_data/b4_affine_c_all_decrypt.png', decrypt_data)
                break
        if is_PNG:
            break
```

Результат:

7955 итераций, a = 63, b = 18.

```
k = 7955  
a = 63  
b = 18
```



Рис. 17.

Расшифруем файл 'im6\_vigener\_c\_all.bmp'. Шифр Виженера, ключ – magistr.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/im6_vigener_c_all.bmp')  
key = 'magistr'  
decrypt_data = []  
for i in range(len(encrypt_data)):  
    m = (encrypt_data[i] - ord(key[i % len(key)])) % 256  
    decrypt_data.append(m)  
read_write_file.write_data_1byte('decrypt_data/im6_vigener_decrypt.bmp', decrypt_data)
```

Результат:



Рис. 18

Зашифруем рис. 18, оставив первые 50 байт без изменения.

```
data = read_write_file.read_data_1byte('decrypt_data/im6_vigener_decrypt.bmp')
key = 'magistr'
encrypt_data = []

for i, d in enumerate(data[50:]):
    encrypt_data.append((d + ord(key[i % len(key)])) % 256)

read_write_file.write_data_1byte('encrypt_data/im6_vigener_encrypt_50.bmp', data[:50] + encrypt_data)
```

Результат:

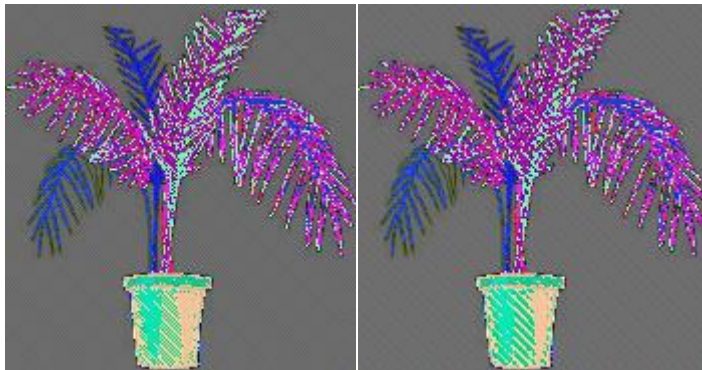


Рис. 19. Шифрование ключом magistr(слева), ключом crypto (справа)

Дешифруем файл 'text4\_vigener\_c\_all.txt'. Известно, что в файле есть слово housewives.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/text4_vigener_c_all.txt')
word = 'housewives'
for i in range(len(encrypt_data)):
    key = ''
    ww = encrypt_data[i:i+len(word)]
    for j, w in enumerate(ww):
        k = (w - ord(word[j % len(word)])) % 256
        key += chr(k)
    is_english = detectEnglish.isEnglish(key[:3])
    if is_english:
        print(key) # key = student

key = 'student'
decrypt_data = []
for i, e in enumerate(encrypt_data):
    decrypt_data.append((e - ord(key[i % len(key)])) % 256)
read_write_file.write_data_1byte('decrypt_data/text4_vigener_c_all_decrypt.txt', decrypt_data)
```

Результатом является ключ – student. И расшифрованный текст:

```
1 |
2 | Long walks at night--
3 | that's what good for the soul:
4 | peeking into windows
5 | watching tired housewives
6 | trying to fight off
7 | their beer-maddened husbands.
```

Расшифруем файл 'text1\_vigener\_c.txt'. Известно, что в файле присутствуют слова it therefore.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/text1_vigener_c.txt')
word = 'it therefore'
for i in range(len(encrypt_data)):
    key = ''
    ww = encrypt_data[i:i+len(word)]
    for j, w in enumerate(ww):
        k = (w - ord(word[j % len(word)])) % 256
        key += chr(k)
    # print(key)
    is_english = detectEnglish.isEnglish(key[:3])
    if is_english:
        print(key) # KEYBOARD

key = 'KEYBOARD'
decrypt_data = []
for i, e in enumerate(encrypt_data[48:]):
    decrypt_data.append((e - ord(key[i % len(key)])) % 256)
read_write_file.write_data_1byte('decrypt_data/text1_vigener_c_all_decrypt.txt', decrypt_data)
```

Результатом является слово – KEYBOARD. И расшифрованный текст:

```
1  Take this kiss upon the brow!
2  And, in parting from you now,
3  Thus much let me avow--
4  You are not wrong, who deem
5  That my days have been a dream;
6  Yet if hope has flown away
7  In a night, or in a day,
8  In a vision, or in none,
9  Is it therefore the less gone?
10 All that we see or seem
11 Is but a dream within a dream.
12
13 I stand amid the roar
14 Of a surf-tormented shore,
15 And I hold within my hand
16 Grains of the golden sand--
17 How few! yet how they creep
18 Through my fingers to the deep,
19 While I weep--while I weep!
20 O God! can I not grasp
21 Them with a tighter clasp?
22 O God! can I not save
23 One from the pitiless wave?
24 Is all that we see or seem
25 But a dream within a dream?
```

Расшифруем файл 'text2\_permutation\_c.txt'. Текст зашифрован шифром перестановки длины 6.

```
import itertools

encrypt_data = read_write_file.read_data_1byte('encrypt_data/text2_permutation_c.txt')
template = '123456'
ew = detectEnglish.ENGLISH_WORDS
for p in itertools.permutations(template, len(template)):
    perm = ''.join(p)
    decrypt_data = ''
    for i in range(0, 24, len(perm)):
        ww = encrypt_data[i:i+len(perm)]
        for j in range(len(perm)):
            decrypt_data += chr(ww[int(perm[j])-1])

    dd_split = decrypt_data.upper().split(' ')
    for d in dd_split:
        if d == '' or d == 'I': continue
        if d not in ew:
            break
    else:
        print('text =', decrypt_data)
        print('permutation =', perm)
        break
```

Результат:

```
text = Whose woods these are I
permutation = 231465
```

Расшифруем теперь весь файл.

```
encrypt_data = read_write_file.read_data_1byte('encrypt_data/text2_permutation_c.txt')
key = '231465'
decrypt_data = []
for i in range(0, len(encrypt_data), len(key)):
    ww = encrypt_data[i:i+len(key)]
    for j in range(len(key)):
        decrypt_data.append(ww[int(key[j]) - 1])
read_write_file.write_data_1byte('decrypt_data/text2_permutation_c_decrypt.txt', decrypt_data)
```

Результат:

```
1 |Whose woods these are I think I know.
2 |His house is in the village, though;
3 |He will not see me stopping here
4 |To watch his woods fill up with snow.
5 |My little horse must think it queer
6 |To stop without a farmhouse near
7 |Between the woods and frozen lake
8 |The darkest evening of the year.
9 |
10|He gives his harness bells a shake
11|To ask if there is some mistake.
12|The only other sound's the sweep
13|Of easy wind and downy flake.
14|The woods are lovely, dark and deep,
15|But I have promises to keep,
16|And miles to go before I sleep,
17|And miles to go before I sleep
```