КРИПТОГРАФІЯ КОМП’ЮТЕРНИЙ ПРАКТИКУМ №2
Криптоаналіз шифру Віженера

ФБ-33 Стогнійчук Інна
ФБ-33 Грабченко Олександр
Варіант 4

Мета роботи: Засвоєння методів частотного криптоаналізу. Здобуття навичок роботи та аналізу потокових шифрів гамування адитивного типу на прикладі шифру Віженера.

Порядок виконання роботи:

0. Уважно прочитати методичні вказівки до виконання комп’ютерного практикуму.
1. Самостійно підібрати текст для шифрування (2-3 кб) та ключі довжини r = 2, 3, 4, 5, а також довжини 10-20 знаків. Зашифрувати обраний відкритий текст шифром Віженера з цими ключами.
2. Підрахувати індекси відповідності для відкритого тексту та всіх одержаних шифртекстів і порівняти їх значення.
3. Використовуючи наведені теоретичні відомості, розшифрувати наданий шифртекст (згідно свого номеру варіанта)

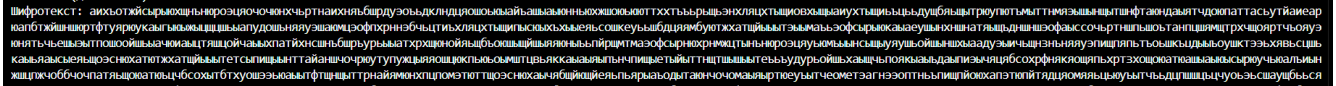
Хід роботи:

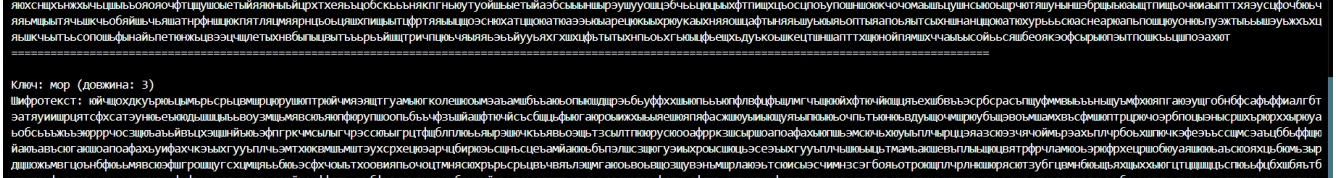
1. Самостійно підібрати текст для шифрування (2-3 кб) та ключі довжини r = 2, 3, 4, 5, а також довжини 10-20 знаків. Зашифрувати обраний відкритий текст шифром Віженера з цими ключами.

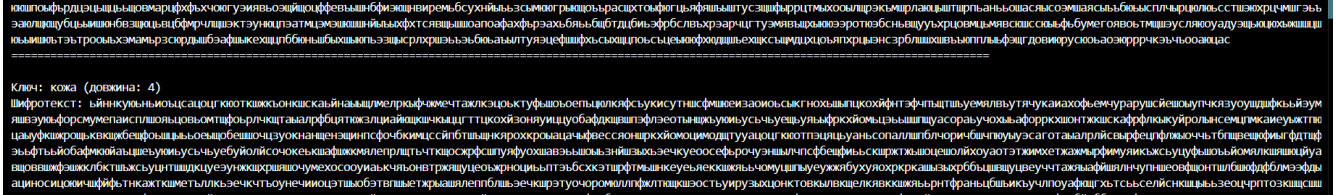
Текст який ми обрали — це відривок з твору “Мастер та Маргарита”. В коді є функція , яка відповідає за його фільтрацію.

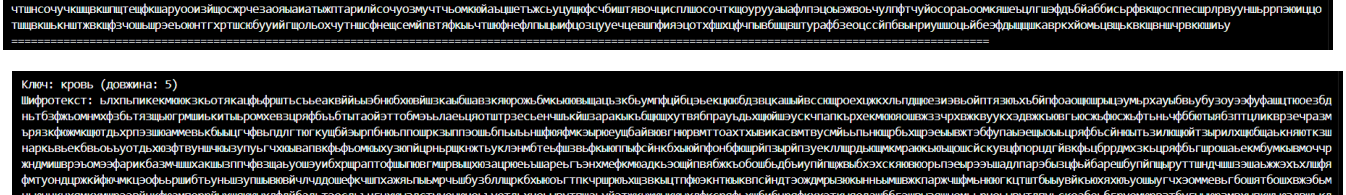
```
    — Ты  знаешь, — говорила Маргарита, —  как раз когда ты заснул вчера ночью,  я  читала  про тьму,
которая пришла со  средиземного моря...  И  эти идолы,  ах, золотые идолы.  Они  почему-то мне все
время не дают  покоя. Мне кажется, что сейчас будет дождь. Ты чувствуешь, как свежеет?

    — Все это хорошо и мило, — отвечал мастер, куря и разбивая рукой дым, —  и  эти  идолы,  бог  с
ними,  но  что  дальше получится,  уж  решительно непонятно!

    Разговор  этот  шел  на закате солнца,  как  раз тогда, когда к Воланду явился  Левий Матвей  на
террасе.  Окошко  подвала было открыто, и  если  бы кто-нибудь заглянул в него, он удивился бы тому,
насколько странно выглядят разговаривающие. На Маргарите прямо на голое тело был накинут черный плащ,
а мастер был в  своем больничном белье. Происходило это  оттого, что Маргарите решительно нечего было
надеть,  так  как  все ее вещи остались в особняке, и хоть этот особняк был очень  недалеко, конечно,
нечего было  и  толковать  о том,  чтобы  пойти туда и  взять там свои вещи.  А  мастер,  у  которого
все костюмы нашли в шкафу, как будто мастер никуда и не уезжал,  просто не желал одеваться, развивая
перед Маргаритой ту мысль, что вот-вот начнется какая-то совершеннейшая чепуха. Правда,  он был выбрит
впервые, считая  с той осенней ночи (в клинике бородку ему подстригали машинкой).

    Комната также имела очень странный вид, и что-нибудь понять  в хаосе ее было очень трудно. На
ковре лежали  рукописи,  они  же  были  и на  диване.  Валялась  какая-то книжка  горбом в кресле.  А
на круглом столе был  накрыт обед, и среди закусок стояло несколько бутылок. Откуда взялись все эти
яства и напитки, было  неизвестно  и Маргарите и мастеру. Проснувшись, они все это застали уже на
столе.

    Проспав до субботнего заката, и мастер, и  его подруга чувствовали себя совершенно окрепшими, и
только одно давало знать о вчерашних приключениях. У обоих  немного  ныл левый  висок. Со стороны  же
психики изменения  в обоих произошли  очень  большие, как убедился бы  всякий,  кто  мог бы
подслушать разговор в  подвальной  квартире.
```

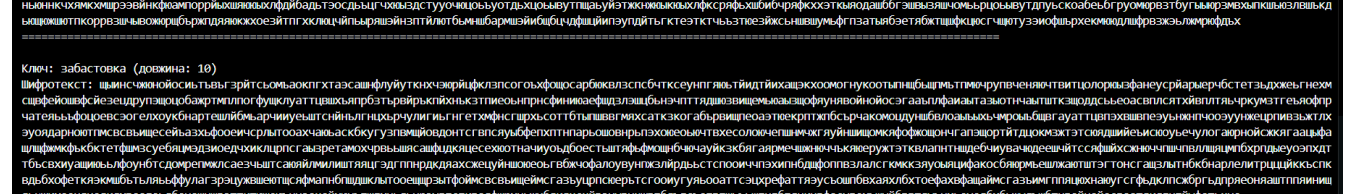
Результати шифрування :

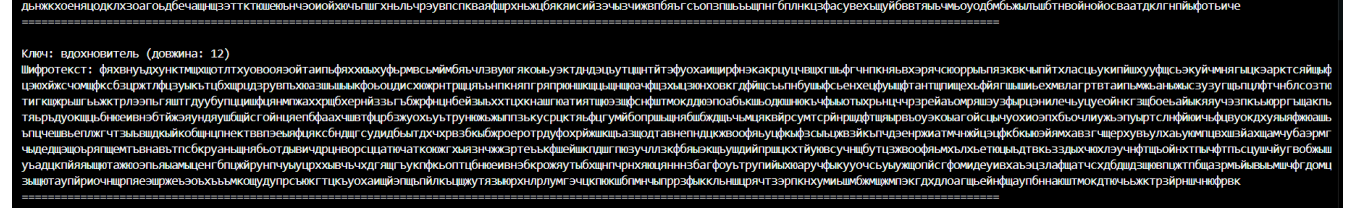


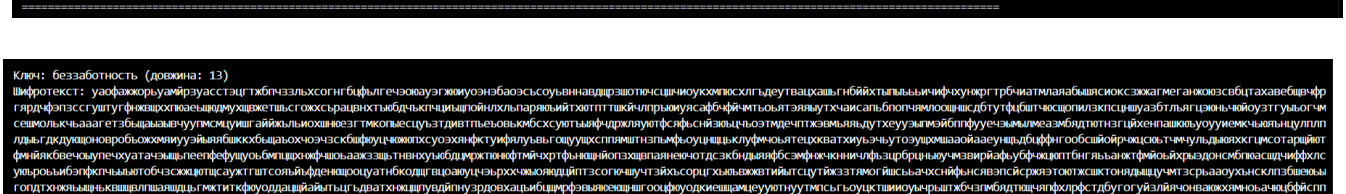


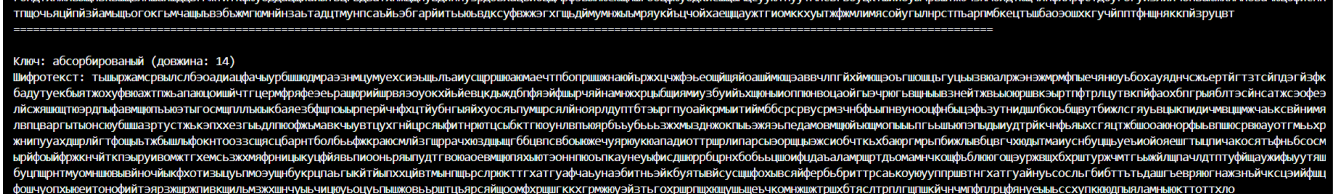


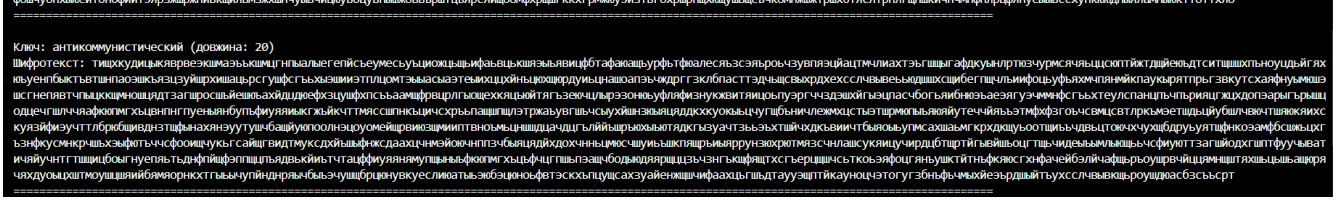








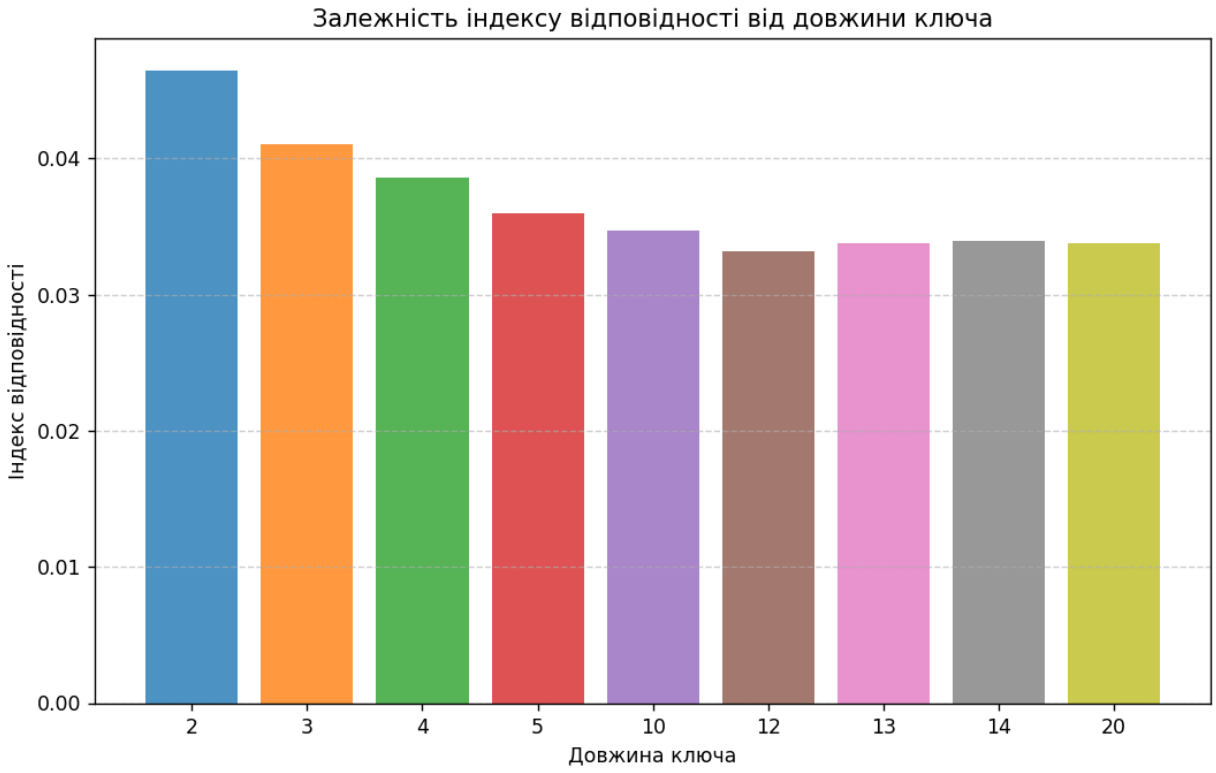




2. Підрахувати індекси відповідності для відкритого тексту та всіх одержаних шифртекстів і порівняти їх значення.

```
Індекс відповідності для відкритого тексту: 0.055973

Індекси відповідності для зашифрованих текстів:
-------------------------------------------------------------------------------------------------
Ключ                    Довжина         Індекс відповідності
-------------------------------------------------------------------------------------------------
он                      2               0.046507
мор                     3               0.041095
кожа                    4               0.038622
кровь                   5               0.035958
забастовка              10              0.034712
беззаботность           13              0.033769
абсорбированый          14              0.033923
антикоммунистический    20              0.033760
-------------------------------------------------------------------------------------------------
```

Діаграма залежностей :



Код :

```python
import re
import matplotlib.pyplot as plt

alphabet = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'

def clean_text(input_text):
    """Очищення тексту від небуквених символів та пробілів"""
    text = input_text.lower()
    text = text.replace('ё', 'е')
    text = re.sub(r'[^е-яА-ЯёЁ ]', '', text)
    text = re.sub(r'\s+', '', text)
    return text

def vigenere_cipher(message, key_word):
    """Шифрування методом Віженера"""
    global alphabet
    result = []
    key_len = len(key_word)

    for i, symbol in enumerate(message):
        if symbol in alphabet:
            m_index = alphabet.index(symbol)
            k_index = alphabet.index(key_word[i % key_len])
            new_char = alphabet[(m_index + k_index) % len(alphabet)]
            result.append(new_char)
        else:
            result.append(symbol)

    return ''.join(result)
```

```python
def coincidence_index(text):
    """Розрахунок індексу відповідності"""
    n = len(text)
    counts = {ch: text.count(ch) for ch in set(text)}
    return sum(c * (c - 1) for c in counts.values()) / (n * (n - 1))

def write_results(filename, clean_text, encrypted_variants, used_keys):
    """Збереження результатів у файл"""
    with open(filename, 'w', encoding='utf-8') as f:
        f.write("Очищений текст:\n")
        f.write(clean_text + "\n\n")
        f.write("Результати шифрування методом Віженера:\n")
        f.write("=" * 150 + "\n")

        for i, key in enumerate(used_keys):
            f.write(f"\nКлюч: {key} (довжина: {len(key)})\n")
            f.write(f"Шифротекст: {encrypted_variants[i]}\n")
            f.write("=" * 150 + "\n")

def visualize_indices(keys, indices):
    """Побудова графіка залежності індексу відповідності від довжини ключа"""
    lengths = [len(k) for k in keys]
    plt.figure(figsize=(10, 6))
    plt.bar(range(len(lengths)), indices, color=plt.cm.tab10.colors[:len(lengths)], alpha=0.8)
    plt.xlabel('Довжина ключа')
    plt.ylabel('Індекс відповідності')
    plt.xticks(range(len(lengths)), lengths)
    plt.title('Залежність індексу відповідності від довжини ключа')
    plt.grid(axis='y', linestyle='--', alpha=0.6)
    plt.show()
```

```
def execute(path):
    with open(path, 'r', encoding='utf-8') as f:
        source = f.read()

    prepared = clean_text(source)

    key_list = [
        "он", "мор", "кожа", "кровь", "забастовка",
        "вдохновитель", "беззаботность", "абсорбированый", "антикоммунистический"
    ]

    encrypted_versions = []
    indices = []

    for key in key_list:
        ciphered = vigenere_cipher(prepared, key)
        encrypted_versions.append(ciphered)
        indices.append(coincidence_index(ciphered))
        print(f"\nКлюч: {key} (довжина: {len(key)})")
        print(f"Шифротекст: {ciphered}")
        print("=" * 150)

    original_ic = coincidence_index(prepared)
    print(f"\nІндекс відповідності для відкритого тексту: {original_ic:.6f}\n")

    print("Індекси відповідності для зашифрованих текстів:")
    print("-" * 100)
    print(f"{'Ключ':<25} {'Довжина':<15} {'Індекс відповідності':<20}")
    print("-" * 100)
    for key, ic in zip(key_list, indices):
        print(f"{key:<25} {len(key):<15} {ic:.6f}")
    print("-" * 100)

    write_results("results_lab2.txt", prepared, encrypted_versions, key_list)
    visualize_indices(key_list, indices)
```

```
file_path = "c:\\Users\\u1208\\OneDrive\\Робочий стіл\\5 сем\\lab2crypto\\lab2\\mm.txt"
execute(file_path)
```

3. Використовуючи наведені теоретичні відомості, розшифрувати наданий шифртекст (згідно свого номеру варіанта).

Обрали варіант 1:

```
from collections import Counter
import os

alphabet = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'
ALPHABET_SIZE = len(alphabet)

RUSSIAN_FREQUENCIES_10000 = [
    820, 159, 450, 170, 318, 791, 94, 165, 745, 121, 349, 440, 322, 670, 1097, 280,
    476, 547, 621, 262, 20, 86, 51, 144, 49, 30, 3, 189, 174, 37, 17, 201
]

def index_of_coincidence(text):
    n = len(text)
    if n <= 1:
        return 0
    counts = Counter(text)
    return sum(c * (c - 1) for c in counts.values()) / (n * (n - 1))

def find_key_length(ciphertext, max_r=30):
    russian_ic = 0.0553
    best_r, best_ic_diff = 0, float('inf')

    print(f"{'Довжина ключа':<15} {'Середній IC':<15} {'Відхилення':<30}")
    print("-" * 60)

    for r in range(2, max_r + 1):
        blocks = [''.join(ciphertext[i::r]) for i in range(r)]
        avg_ic = sum(index_of_coincidence(block) for block in blocks) / r
        ic_diff = abs(avg_ic - russian_ic)
        print(f"{r:<15} {avg_ic:<15.4f} {ic_diff:<30.4f}")
        if ic_diff < best_ic_diff:
            best_ic_diff, best_r = ic_diff, r
    return best_r
```

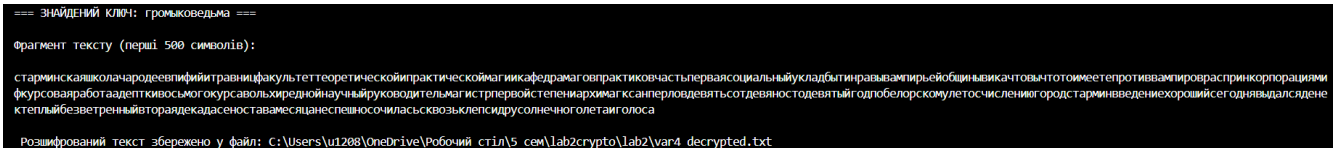
```python
def caesar_decrypt(char, shift, alphabet):
    if char not in alphabet:
        return char
    return alphabet[(alphabet.index(char) - shift) % ALPHABET_SIZE]

def chi_squared_test(block, shift):
    decrypted = [caesar_decrypt(c, shift, alphabet) for c in block]
    counts = Counter(decrypted)
    chi_sq = 0
    for i in range(ALPHABET_SIZE):
        expected = len(block) * (RUSSIAN_FREQUENCIES_10000[i] / 10000)
        observed = counts.get(alphabet[i], 0)
        if expected > 0:
            chi_sq += (observed - expected) ** 2 / expected
    return chi_sq

def break_vigenere_by_frequency(ciphertext, key_length):
    blocks = [''.join(ciphertext[i::key_length]) for i in range(key_length)]
    key_shifts = []

    print("\n--- Визначення ключа ---")
    for i, block in enumerate(blocks):
        best_shift, best_chi = 0, float('inf')
        for shift in range(ALPHABET_SIZE):
            chi = chi_squared_test(block, shift)
            if chi < best_chi:
                best_chi, best_shift = chi, shift
        key_shifts.append(best_shift)
        print(f"Блок {i+1}/{key_length}: shift = {best_shift} ({alphabet[best_shift]}), χ² = {best_chi:.2f}")

    key = ''.join(alphabet[s] for s in key_shifts)
    plaintext = [caesar_decrypt(c, key_shifts[i % key_length], alphabet) for i, c in enumerate(ciphertext)]
    return ''.join(plaintext), key

input_path = r"C:\Users\u1208\OneDrive\Робочий стіл\5 сем\lab2crypto\lab2\var4.txt"
output_path = os.path.join(os.path.dirname(input_path), "var4_decrypted.txt")
```

```python
try:
    with open(input_path, 'r', encoding='utf-8') as f:
        ciphertext_raw = f.read()
except FileNotFoundError:
    print(f"Файл не знайдено: {input_path}")

ciphertext = ''.join(c for c in ciphertext_raw.lower() if c in alphabet)

if ciphertext:
    key_length = find_key_length(ciphertext)
    print(f"\n=== Знайдена довжина ключа: {key_length} ===")
    plaintext, key = break_vigenere_by_frequency(ciphertext, key_length)
    print(f"\n=== ЗНАЙДЕНИЙ КЛЮЧ: {key} ===")
    print("\nФрагмент тексту (перші 500 символів):\n")
    print(plaintext[:500])
    with open(output_path, 'w', encoding='utf-8') as f:
        f.write(plaintext)
    print(f"\n Розшифрований текст збережено у файл: {output_path}")
else:
    print("Помилка: Шифротекст порожній або без символів алфавіту.")
```

Аутпут :

```
Довжина ключа   Середній IC     Відхилення
-----------------------------------------------
2               0.0327          0.0226
3               0.0326          0.0227
4               0.0327          0.0226
5               0.0325          0.0228
6               0.0326          0.0227
7               0.0327          0.0226
8               0.0328          0.0225
9               0.0325          0.0228
10              0.0325          0.0228
11              0.0328          0.0225
12              0.0327          0.0226
13              0.0543          0.0010
14              0.0328          0.0225
15              0.0324          0.0229
16              0.0329          0.0224
17              0.0325          0.0228
18              0.0326          0.0227
19              0.0325          0.0228
20              0.0324          0.0229
21              0.0328          0.0225
22              0.0326          0.0227
23              0.0322          0.0231
24              0.0326          0.0227
25              0.0323          0.0230
26              0.0538          0.0015
27              0.0323          0.0230
28              0.0327          0.0226
29              0.0325          0.0228
30              0.0324          0.0229

=== Знайдена довжина ключа: 13 ===
```

```
--- Визначення ключа ---
Блок 1/13: shift = 3 (г), χ² = 44.74
Блок 2/13: shift = 16 (р), χ² = 41.87
Блок 3/13: shift = 14 (о), χ² = 48.64
Блок 4/13: shift = 12 (м), χ² = 42.32
Блок 5/13: shift = 27 (ы), χ² = 28.02
Блок 6/13: shift = 10 (к), χ² = 86.55
Блок 7/13: shift = 14 (о), χ² = 39.24
Блок 8/13: shift = 2 (в), χ² = 41.29
Блок 9/13: shift = 5 (е), χ² = 41.39
Блок 10/13: shift = 4 (д), χ² = 47.06
Блок 11/13: shift = 28 (ь), χ² = 44.70
Блок 12/13: shift = 12 (м), χ² = 39.42
Блок 13/13: shift = 0 (а), χ² = 42.36
```



Розшифрований текст — це відрізок з книги Ольги Громико “Капкан для некроманта”.

Висновок:

У ході лабораторної роботи було досліджено роботу шифру Віженера та методи його криптоаналізу. Основною метою було визначити індекси відповідності для зашифрованого тексту та встановити можливість розшифрування за допомогою частотного аналізу.

За допомогою обчислення індексу відповідності вдалося визначити правильну довжину ключа. Для цього шифротекст було розбито на блоки відповідно до різних значень періоду, і для кожного блоку розраховано індекс відповідності. Порівняння отриманих значень з теоретичними частотними характеристиками російської мови дозволило визначити оптимальну довжину ключа, що надалі дало змогу розшифрувати текст методом частотного криптоаналізу.

З отриманих результатів видно, що для менших значень довжини ключа середній індекс відповідності має найбільше значення. Це пояснюється тим, що короткі ключі частіше створюють повтори символів у шифротексті, тоді як довші ключі зменшують ймовірність таких збігів, що відповідає теоретичним очікуванням.