

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ “КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. ІГОРЯ СІКОРСЬКОГО”

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №2

Криптоаналіз шифру Віженера

Виконали:

Студенти групи ФБ-22

Орлов Антон, Ялбуган Федір

(бригада 7)

КИЇВ 2024

**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**

Ми взяли шматок тексту з попередньої лабораторної роботи розміром 3КБ.

****

Далі підібрали ключи відповідних довжин:

keys = ["ад", "рай", "вера", "иисус", "судныйдень", "триединство", "апостолиоанн", "игорьандрущак", "архангелмихаил"]

Далі ми зашифрували текст з цими ключами і створили окремий файл для кожного ШТ:

def text\_to\_shifts(key, alphabet):

    return [alphabet.index(char) for char in key if char in alphabet]

def vigenere\_encrypt(text, key\_shifts, alphabet):

    encrypted\_text = []

    key\_length = len(key\_shifts)

    for i, char in enumerate(text):

        if char in alphabet:

            shift = key\_shifts[i % key\_length]

            char\_index = alphabet.index(char)

            new\_index = (char\_index + shift) % len(alphabet)

            encrypted\_text.append(alphabet[new\_index])

        else:

            encrypted\_text.append(char)

    return ''.join(encrypted\_text)

def encrypt\_text\_with\_keys(input\_file, keys, alphabet):

    with open(input\_file, 'r', encoding='utf-8') as file:

        text = file.read().strip()

    for key in keys:

        key\_shifts = text\_to\_shifts(key, alphabet)

        encrypted\_text = vigenere\_encrypt(text, key\_shifts, alphabet)

        output\_filename = f'encrypted{len(key)}.txt'

        with open(output\_filename, 'w', encoding='utf-8') as output\_file:

            output\_file.write(encrypted\_text)

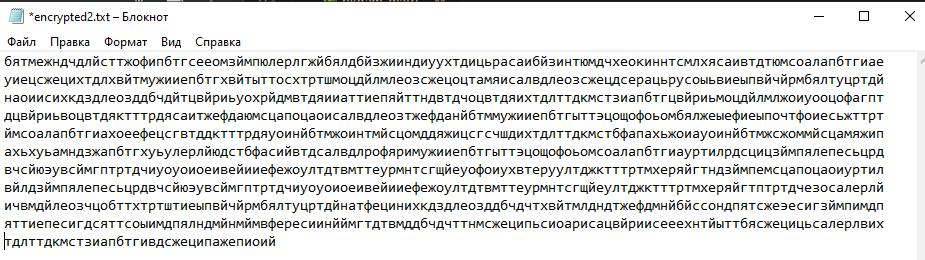
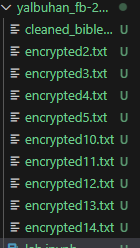
alphabet = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'

keys = ["ад", "рай", "вера", "иисус", "судныйдень", "триединство", "апостолиоанн", "игорьандрущак", "архангелмихаил"]

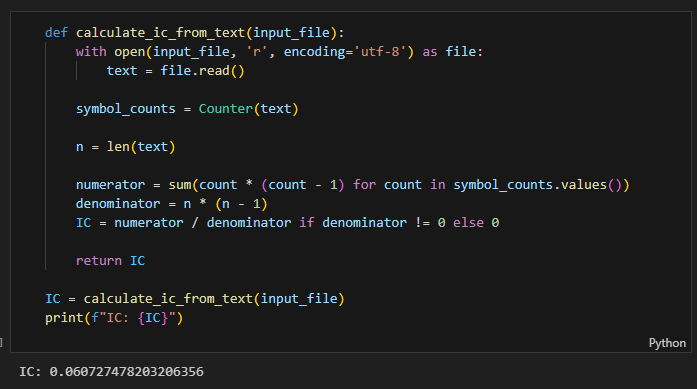
input\_file = 'cleaned\_bible\_no\_spaces.txt'

encrypt\_text\_with\_keys(input\_file, keys, alphabet)

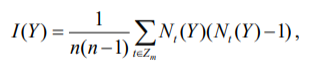
Результат:



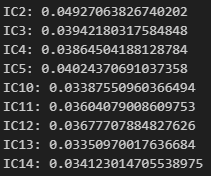
**2**

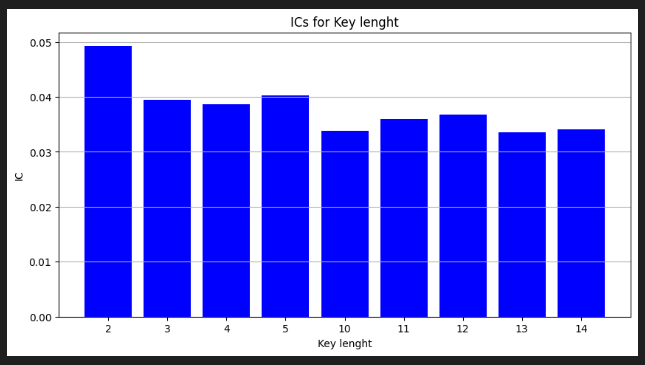
Підрахований індекс ВТ (**IC: 0.060727478203206356**):  


Обраховували по формулі:

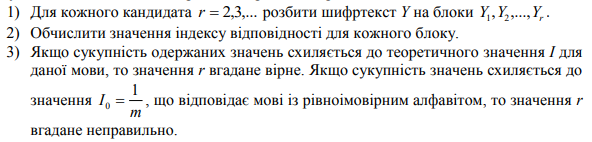
****

Після застосування цієї функці до шифротекстів, отримали такі Індекси Відповідності:





**3**

Беремо ШТ свого варіанту. Використовуємо цей алгоритм для знаходження довжини ключа:  


В коді це реалізовано так:

def calculate\_single\_IC(text\_block):

    freaks = Counter(text\_block)

    n = len(text\_block)

    if n == 0:

        return 0

    index = sum(f \* (f - 1) for f in freaks.values()) / (n \* (n - 1))

    return index

def splitting\_func(text, key\_length):

    blocks = ['' for \_ in range(key\_length)]

    for i, char in enumerate(text):

        blocks[i % key\_length] += char

    return blocks

def calculate\_ICs(file\_path, min\_key\_length=2, max\_key\_length=30):

    with open(file\_path, 'r', encoding='utf-8') as file:

        text = file.read().replace('\n', '').replace(' ', '')

    for key\_length in range(min\_key\_length, max\_key\_length + 1):

        blocks = splitting\_func(text, key\_length)

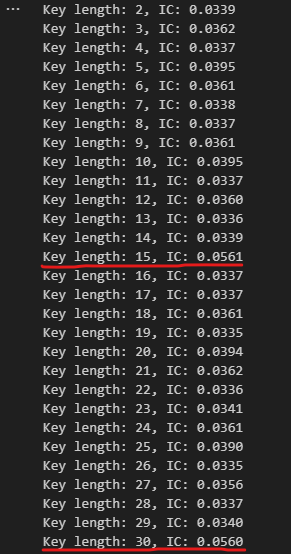
        indices = [calculate\_single\_IC(block) for block in blocks]

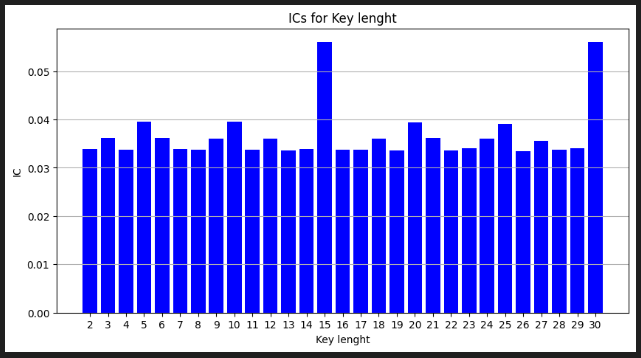
        average\_index = sum(indices) / len(indices)

        print(f"Key length: {key\_length}, IC: {average\_index:.4f}")

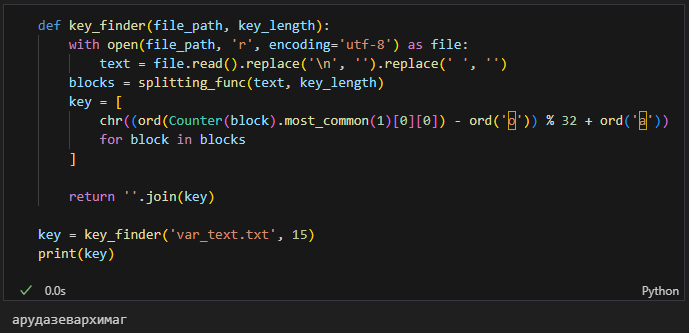
calculate\_ICs('var\_text.txt')

Отримуємо такі значення IC, вибираємо ті, які найближчі до індексу відповідності ВТ:





Звідси випливає, що наш ключ скоріш за все буде довжиною 15 або 30. Перевіримо це за допомогою наступного коду. Ділимо на блоки для довжини ключа 15, обраховуємо зсув між найпоширенішою буквою кожного блоку та буквою «о» (найпоширенішою буквою відкритого тексту російською мовою). Ці числа, що виходять, будуть індексами букв нашого ключа.



Отримали такий ключ: **арудазевархимаг**

Одразу бачимо слово «архимаг», яке стане основою для пошуку справжнього ключа. Ми вирішили здогадатися яким буде повний ключ, і знайшли книгу Олександра Рудазова «Архімаг», тому повним ключем нашого шифротексту буде **арудазовархимаг** (майже те саме, що видав код))).

Перевіримо це інвертованою до *vigenere\_encrypt()* фукції з першого пункту завдання:

def vigenere\_decrypt(ciphertext, key):

    alphabet = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'

    decrypted\_text = []

    key = key.lower()

    key\_length = len(key)

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

    for i, char in enumerate(ciphertext):

        if char in alphabet:

            text\_index = alphabet.index(char)

            key\_index = alphabet.index(key[i % key\_length])

            decrypted\_index = (text\_index - key\_index) % len(alphabet)

            decrypted\_text.append(alphabet[decrypted\_index])

        else:

            decrypted\_text.append(char)

    return ''.join(decrypted\_text)

with open('var\_text.txt', 'r', encoding='utf-8') as file:

    ciphertext = file.read().replace('\n', '').replace(' ', '')

key = "арудазовархимаг"

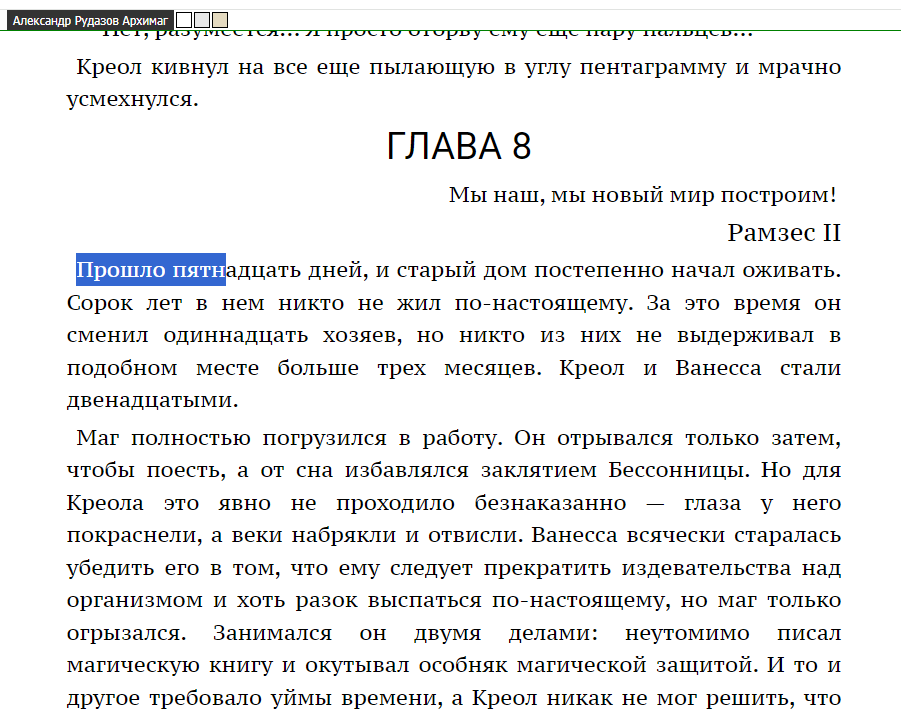
decrypted\_message = vigenere\_decrypt(ciphertext, key)

print("Decrypted text:", decrypted\_message)

У виводі бачимо знайдений ВТ:



Можна навіть знайти, що цей текст було взято з 8 глави з цієї книги:



**Висновки:** Підрахунок індексів відповідності для відкритого тексту та всіх шифртекстів дав змогу зробити висновки щодо того, як збільшення довжини ключа впливає на ентропію та структуру шифртексту. Зазвичай, чим довший ключ, тим більше шифртекст наближається до випадковості, що ускладнює його розшифрування.

Використовуючи теоретичні методи криптоаналізу, вдалося успішно розшифрувати шифртекст, отриманий за зашифрованим методом Віженера. Це продемонструвало ефективність індексу відповідності у визначенні довжини ключа та подальшого розшифрування тексту. Завдання сприяло розумінню принципів роботи шифру Віженера та методів криптоаналізу для його зламу.