НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

з лабораторної роботи №*2*

з дисципліни «Криптографічні методи захисту інформації»

на тему:

*Розробка криптосистем на основі шифру Віженера та шифру з використанням двійкового гамування*

|  |  |
| --- | --- |
| Виконав(-ла): | Керівник: |
| студент(ка) групи КМ-01 | *ст. викладач Бай Ю. П.* |
| *Дюбакін Р. С.* |  |

Київ — 2022

ЗМІСТ

[ЗАВДАННЯ 2](#_Toc96528890)

[Основні теоретичні відомості з шифру Віженера 3](#_Toc96528891)

[Завдання з шифру Віженера 3](#_Toc96528892)

[Основні теоретичні відомості з шифру двійкового гамування 4](#_Toc96528893)

[Завдання з шифру двійкового гамування 4](#_Toc96528894)

[Список літератури 5](#_Toc96528895)

[Додаток 1 6](#_Toc96528896)

[Додаток 2 7](#_Toc96528897)

# ЗАВДАННЯ

***Мета роботи:*** розробити криптосистеми на основі шифрів Віженера та двійкового гамування.

1. Обрати початковий текст англійською, українською або російською мовою довжиною від 8 до 20 символів, та ключове слово від 3 до 8 символів. Зашифрувати текст, користуючись шифром Віженера. Алфавіт, криптотекст та ключ записати в [Таблицю\_3](https://docs.google.com/spreadsheets/d/1Fqu6vkzaJYX1mQPbSPU4A1CSVkOK_MHggPLZTmA6gzA/edit?usp=sharing) в СВІЙ РЯДОК у стовпчики AF, AG, AH. Перед шифруванням пробіли з початкового тексту необхідно видалити. (*1 бал*).

2. Користуючись наданим ключем, розшифрувати отриманий криптотекст за алгоритмом Віженера (завдання дивитись в [Таблиці\_3](https://docs.google.com/spreadsheets/d/1Fqu6vkzaJYX1mQPbSPU4A1CSVkOK_MHggPLZTmA6gzA/edit?usp=sharing) у рядку НАД СВОЇМ). Відповідь записати в рядок НАД СВОЇМ у стовпчик AI. (*1 бал*)

3. Написати програмний код для шифрування - розшифрування за шифром Віженера. Продемонструвати роботу створеної програми на контрольних прикладах. (*2 бали*)

4. Обрати початковий текст англійською, українською або російською мовою довжиною від 8 до 15 символів (ім'я відомої особистості, відома назва пісні чи групи, комп'ютерна гра, тощо). Записати символи початкового тексту у двійковому вигляді, користуючись кодовою таблицею сp866.

5. Згенерувати ключ - псевдовипадкову гамму, довжина якої дорівнює або більша довжини початкового тексту. Записати ключ у двійковому вигляді в [Таблицю\_4](https://docs.google.com/spreadsheets/d/1TzJmWgV5XCRY0FRKcvfvVlcJ3Yhk3TgyJc02ud0XWRo/edit?usp=sharing) в СВІЙ рядок, стовпчик AG. Значення ключа повинно бути унікальним в стовпчику AG. (*1 бал*)

6. **Шифрування:** для початкового тексту та згенерованого ключа виконати побітово операцію XOR. Отриманий криптотекст у двійковому вигляді записати в [Таблицю\_4](https://docs.google.com/spreadsheets/d/1TzJmWgV5XCRY0FRKcvfvVlcJ3Yhk3TgyJc02ud0XWRo/edit?usp=sharing) в СВІЙ рядок, стовпчик AH. (*1 бал*)

7. **Розшифрування** (завдання дивитись в [Таблиці\_4](https://docs.google.com/spreadsheets/d/1TzJmWgV5XCRY0FRKcvfvVlcJ3Yhk3TgyJc02ud0XWRo/edit?usp=sharing) у рядку НАД СВОЇМ): користуючись наданим ключем, розшифрувати отриманий криптотекст за алгоритмом XOR. Відповідь записати в рядок НАД СВОЇМ: у стовпчик AI - у двійковому вигляді, у стовпчик AJ - у вигляді тексту згідно з таблицею cp866. (2 *бали*)

8. Написати програмний код для шифрування - розшифрування за шифром двійкового гамування. Продемонструвати роботу створеної програми на прикладах. (*1 бал*)

9. Оформити звіт. (*1 бал*)

# Основні теоретичні відомості з шифру Віженера

Шифр Віженера — поліалфавітний шифр, який як ключ використовує слово.

Якщо пронумерувати літери алфавіту від 0 до 32 (а → 0, б → 1, в → 2, …), то шифрування Віженера можна подати формулою:

Ci = (Pi + Kj) mod 33,

де Kj — j-та літера ключового слова, Pi — і-а літера вихідного слова.

Ключове слово повторюється, поки не отримано гаму, рівну довжині повідомлення.

Дешифрування відбувається за наступною формулою:

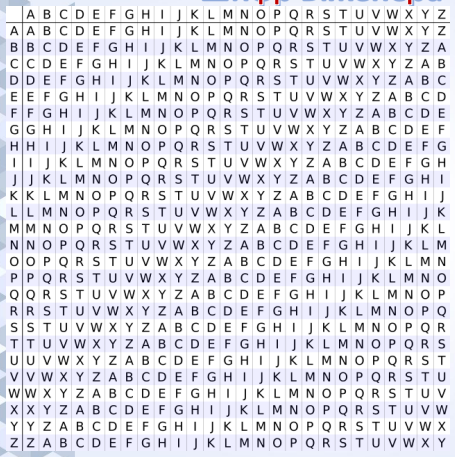
Ci = (Pi + 33 - Kj) mod 33

Отримав назву на честь Блеза де Віженера, хоча насправді його винайшов італійський криптограф Джованні Баттіста Белласо.

Табличне представлення

По вертикалі вибираємо літери відкритого тексту, а по горизонталі — ключа, на перетині цих значень отримуємо знаки шифротексту.

# Завдання з шифру Віженера

1. Оберемо у вигляді початкового слова англійське wonderful. Ключове слово – dog. Зашифруємо обране слово за допомоги шифру Віженера, скористаємося для цього таблицею представлення.

Спочатку обираємо у верхньому рядку літеру, яку потрібно зашифрувати. Далі у відповідному стовпчику знаходимо літеру ключового слова та у відповідному лівому рядку знаходимо відповідну зашифровану літеру.

Таким чином отримаємо: zctgsxiir.

1. Таким же чином розшифруємо це слово. Знаходимо літеру у верхньому рядку зашифрованого слова та у відповідному стовпчику шукаємо літеру ключового слова і у лівому стовпчику знаходимо розшифроване слово.

Отримаємо знову wonderful.

1. У додатках доданий програмний код, написаний мовою Python, що реалізовує шифрування та дешифрування за Віженером.

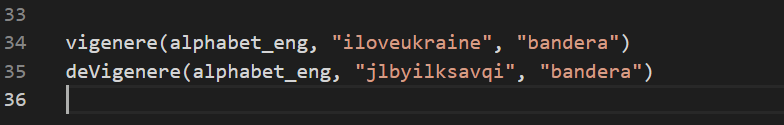
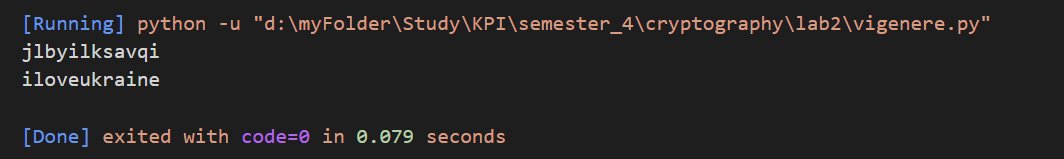


Рис.1 – Результат виконання програми шифрування та дешифрування за Віженером

# Основні теоретичні відомості з шифру двійкового гамування

Принцип шифрування полягає в формуванні генератором псевдовипадкових чисел (ГПСЧ) гами шифру і накладення цієї гами на відкриті дані оборотним чином, наприклад, шляхом додавання по модулю два. Процес дешифрування даних зводиться до повторної генерації гами шифру і накладення гами на зашифровані дані. Ключем шифрування в даному випадку є початковий стан генератора псевдовипадкових чисел. При одному і тому ж початковому стані ГПСЧ формуватиме одні і ті ж псевдо випадкові послідовності.

Перед шифруванням відкриті дані зазвичай розбивають на блоки однакової довжини, наприклад, по 64 біта. Гамма шифру також виробляється у вигляді послідовності блоків тієї ж довжини.

Стійкість шифрування методом гамування визначається головним чином властивостями гами - довжиною періоду і рівномірністю статистичних характеристик. Остання властивість забезпечує відсутність закономірностей у появі різних символів в межах періоду. Отриманий зашифрований текст є досить важким для розкриття. По суті справи гамма шифру повинна змінюватися випадковим чином для кожного шифруемого блоку.

Зазвичай поділяють два різновиди гамування - з кінцевої і нескінченної гамами. При хороших статистичних властивостях гами якість шифрування визначається тільки довжиною періоду гами. При цьому, якщо довжина періоду гами перевищує довжину шіфруемоготексту, то такий шифр теоретично є абсолютно стійким, т. Е. Його не можна розкрити за допомогою статистичної обробки зашифрованого тексту, а можна розкрити тільки прямим перебором. Крипостійкість в цьому випадку визначається розміром ключа.

Для утворення шифротексту повідомлення об'єднується операцією XOR з ключем (названим одноразовим блокнотом або шифроблокнотом). При цьому ключ повинен мати три критично важливі властивості:

1. Бути справді випадковим;
2. Збігатися за розміром з заданим відкритим текстом;
3. Застосовуватися тільки один раз;
4. Повинен зберігатися в повній таємниці сторонами, що спілкуються.

Алгоритм шифру двійкового гамування

1. Інформація (текст, цифрові дані, зображення, мова) подається у вигляді двійкового коду.
2. Формується псевдовипадкова числова послідовність (ключ) теж у вигляді двійкового коду.
3. Вихідний текст і ключ розташовуються один під одним.
4. До кожної пари двійкових символів вихідного тексту та ключа застосовується операція ХОR – так одержується зашифроване повідомлення у двійковому коді.
5. Для дешифрування необхідно згенерувати таку саму псевдовипадкову двійкову послідовність, як і початковий ключ, і накласти її на зашифрований текст, тобто застосувати посимвольно ХОR до шифротексту та ключа.

# Завдання з шифру двійкового гамування

1. Оберемо початковий текст англійською мовою (ok google) та запишемо його в двійковому вигляді: 01101111 01101011 00100000 01100111 01101111 01101111 01100111 01101100 01100101.
2. Згенеруємо ключ-псевдовипадкову гамму:

165 077 202 024 037 048 187 029 109

Перепишемо у двійковому вигляді:

10100101 01001101 11001010 00011000 00100101 00110000 10111011 00011101 01101101

1. Для початкового тексту та згенерованого ключа виконаємо побітово операцію XOR. Отриманий криптотекст у двійковому вигляді:

11001010 00100110 11101010 01111111 01001010 01011111 11011100 01110001 00001000

1. Візьмемо зашифрований текст, записаний у двійковому вигляді:

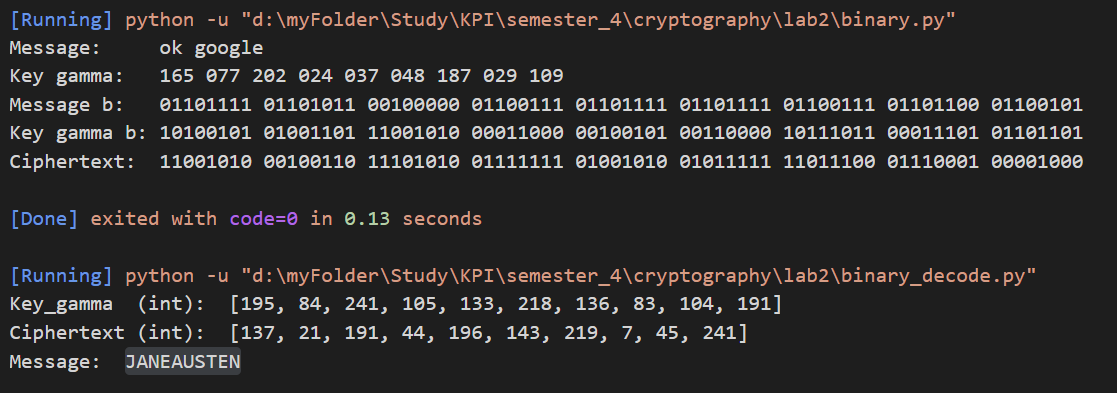
10001001 00010101 10111111 00101100 11000100 10001111 11011011 00000111 00101101 11110001,

та ключ:

11000011 01010100 11110001 01101001 10000101 11011010 10001000 01010011 01101000 10111111.

Для зашифрованого тексту та ключа виконаємо побітово операцію XOR. Отриманий розшифрований текст: JANEAUSTEN.

1. У додатках доданий програмний код мовою Python, що реалізовує шифрування та дешифрування двійковим гамуванням.

Рис.2 – Приклад роботи програми шифрування та дешифрування двійковим гамуванням.

# Список літератури

1. Тарнавський Ю.А. Технології захисту інформації [Електронний ресурс] / Ю. А. Тарнавський. – Київ: КПІ ім. Ігоря Сікорського, 2018. – 162 с.
2. Шнайер Б. Прикладная криптография: Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – М.: Диалектика, 2003. – 610 с.
3. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии. – М.: Гелиос АРВ, 2001. – 480 с.
4. Столлингс В. Криптография и защита сетей: принципы и практика, 2-е изд.: Пер. с англ. – М.: «Вильямс», 2001. – 672 с.
5. Сингх С. Книга шифров. Тайная история шифров и их расшифровки, Пер. с англ. – М.: «Астрель», 2007. – 447 с.

# Додаток 1

Текст програми, що реалізує криптосистему Віженера

alphabet\_eng = "abcdefghijklmnopqrstuvwxyz"

def vigenere(alphabet, plainText, keyword):

    cryptoText = ''

    times = (len(plainText) // len(keyword)) + 1

    keyword = keyword \* times

    keyword = keyword[ :(len(plainText) - len(keyword))]

    for letter, keyLetter in zip(plainText.lower(), keyword.lower()):

        if letter in alphabet:

            value = (alphabet.find(letter) + alphabet.find(keyLetter)) % len(alphabet)

            cryptoText += alphabet[value]

        else:

            cryptoText += letter

    print(cryptoText)

    return cryptoText

def deVigenere(alphabet, cryptoText, keyword):

    plainText = ''

    times = (len(cryptoText) // len(keyword)) + 1

    keyword = keyword \* times

    keyword = keyword[:(len(cryptoText) - len(keyword))]

    for letter, keyLetter in zip(cryptoText.lower(), keyword.lower()):

        if letter in alphabet:

            value = (alphabet.find(letter) -

                     alphabet.find(keyLetter) + len(alphabet)) % len(alphabet)

            plainText += alphabet[value]

        else:

            plainText += letter

    print(plainText)

    return plainText

vigenere(alphabet\_eng, "wonderful", "dog")

deVigenere(alphabet\_eng, "zctgsxiir", "dog")

# Додаток 2

Текст програми, що реалізує криптосистему з використанням двійкового гамування

#-\*- coding: utf-8 -\*-

import random

ENCODING = 'cp866'

def convert\_to\_bin(line):

    return ' '.join(format(x, "08b") for x in line)

message = 'ok google'

print('Message:    ', message)

message\_b = message.encode(ENCODING)

random.seed(7)

key\_gamma\_b = bytes(random.randint(0, 255) for i in range(len(message\_b)))

print('Key gamma:  ', ' '.join(format(x, "03d") for x in key\_gamma\_b))

print(f'Message b:   {convert\_to\_bin(message\_b)}')

print(f'Key gamma b: {convert\_to\_bin(key\_gamma\_b)}')

encoded\_b = bytes(x ^ y for x, y in zip(message\_b, key\_gamma\_b))

print(f'Ciphertext:  {convert\_to\_bin(encoded\_b)}')

#-\*- coding: utf-8 -\*-

ENCODING = 'cp866'

gamma\_binary = \

    '01110101 00101100 10101000 11011001 10100101 00101010 00000001 11011100 11111100'

ciphertext\_binary = \

    '00011010 01000111 10001000 10111110 11001010 01000101 01100110 10110000 10011001'

gamma\_int = [int(x, 2) for x in gamma\_binary.split()]

print("Key\_gamma  (int): ", gamma\_int)

ciphertext\_int = [int(x, 2) for x in ciphertext\_binary.split()]

print("Ciphertext (int): ", ciphertext\_int)

message\_b = bytes(x ^ y for x, y in zip(ciphertext\_int, gamma\_int))

print("Message: ", message\_b.decode(ENCODING))