

## Лабораторна робота №2

**Виконав:** Ваврикович Михайло ПМІ-33

**Тема:** Шифр Тритеміуса

**Мета:** Розробити криптосистему на основі шифру Тритеміуса

### Хід роботи

1. Модифікував інтерфейс криптографічної системи симетричного шифрування з лабораторної роботи №1, забезпечивши можливість використання в якості ключа:
  - а. 2-вимірного вектору для зберігання коефіцієнтів лінійного рівняння шифрування

#### Trithemius cipher

Text

Lviv National University

Encrypted text

Zttt Ddptfddz Rdtllntpz

Key type

☒ Linear equation ☐ Nonlinear equation

Coef A

2

Coef B

3

Coef C

b. 3-вимірного вектору для зберігання коефіцієнтів нелінійного рівняння шифрування

**Trithemius cipher**

Text

Lviv National University

Encrypted text

Ztgt Xkjgckz Mxgtcpkgjm

Key type

☐ Linear equation ☒ Nonlinear equation

Coef A

2

Coef B

3

Coef C

10

c. Текстового рядка (гасла)

Text

Lviv National University

Encrypted text

Wqqq Yvbdziig Fiqqpmadet

\* Alphabet

abcdefghijklmnopqrstuvwxyz

\* Motto

Lviv

Save as

☒ Plain text ☐ File

2. Додати до системи класів з лабораторної роботи №1 класи та методи, необхідні для реалізації симетричного шифрування методом Тритеміуса, передбачивши в них методи валідації ключа, валідації шифрування і розшифрування даних.

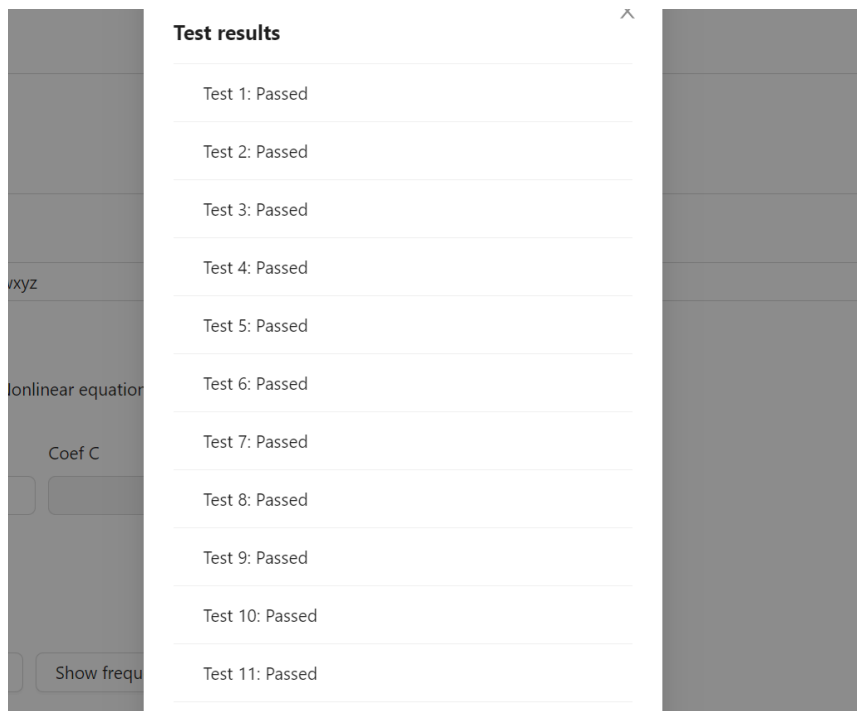
```
File Edit Selection View Go Run Terminal Help trithemius.ts - cryptosystem - Visual Studio Code

src > ciphers > TS trithemius.ts > ...
1 import { isUpperCase, mod, modInv } from '@utils';
2
3 const helper = (text: string, getKey: (x: number) => number, alphabet: string) => {
4   let res = '';
5
6   for (let i = 0; i < text.length; i++) {
7     const x = alphabet.indexOf(text[i].toLowerCase());
8
9     if (x === -1) {
10      res += text[i];
11      continue;
12    }
13
14    const isUpper = isUpperCase(text[i]);
15
16    const newChar = alphabet[mod(getKey(x), alphabet.length)];
17    res += isUpper ? newChar.toUpperCase() : newChar;
18  }
19
20  return res;
21 };
22
23 export default {
24   encode: {
25     linear: (text: string, coefA: number, coefB: number, alphabet: string) => {
26       const getKey = (x: number) => coefA * x + coefB;
27       return helper(text, getKey, alphabet);
28     },
29     nonlinear: (text: string, coefA: number, coefB: number, coefC: number, alphabet: string) => {
30       const getKey = (x: number) => coefA * x ** 2 + x * coefB + coefC;
31       return helper(text, getKey, alphabet);
32     },
33   },
34   decode: {
35     linear: (text: string, coefA: number, coefB: number, alphabet: string) => {
36       const getKeyInv = (y: number) => (y - coefB) * modInv(coefA, alphabet.length);
37     },
38   },
39 }
```

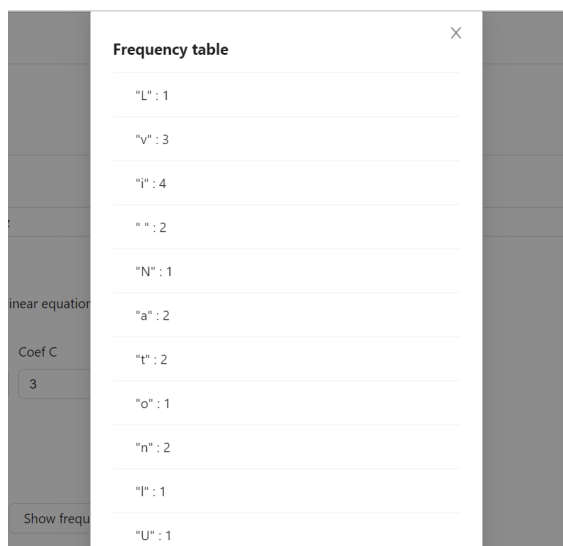
```
File Edit Selection View Go Run Terminal Help vigenere.ts - cryptosystem - Visual Studio Code

src > ciphers > TS vigenere.ts > helper
1 import { isUpperCase, mod } from '@utils';
2
3 const helper = (text: string, motto: string, alphabet: string, getKey: (x: number, y: number) => number) => {
4   let res = '';
5   let count = 0;
6
7   for (let i = 0; i < text.length; i++) {
8     const x = alphabet.indexOf(text[i].toLowerCase());
9
10    if (x === -1) {
11      count++;
12      res += text[i];
13      continue;
14    }
15
16    const isUpper = isUpperCase(text[i]);
17
18    const y = alphabet.indexOf(motto[mod(i - count, motto.length)]);
19    const newChar = alphabet[mod(getKey(x, y), alphabet.length)];
20    res += isUpper ? newChar.toUpperCase() : newChar;
21  }
22
23  return res;
24 };
25
26 export default {
27   encode: (text: string, motto: string, alphabet: string) => helper(text, motto, alphabet, (x, y) => x + y),
28   decode: (text: string, motto: string, alphabet: string) => helper(text, motto, alphabet, (x, y) => x - y),
29 };
30
```

### 3. Виконав тестування роботи системи.



### 4. Застосував частотні таблиці для української та англійської мов для атаки.



**Висновок:** я розробив криптосистему на основі шифру Тритеміуса.