МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий фізико-технічний інститут
Кафедра інформаційної безпеки

Дисципліна «Криптографія»

Комп’ютерний практикум
Робота №3
Криптоаналіз афінної біграмної підстановки

Виконали: студенти гр. ФБ-12 Головко М. С. і Марчук І. С.

Київ – 2023

Мета роботи: Набуття навичок частотного аналізу на прикладі розкриття моноалфавітної підстановки; опанування прийомами роботи в модулярній арифметиці.

Порядок виконання роботи:

0. Уважно прочитати методичні вказівки до виконання комп'ютерного практикуму.
1. Реалізувати підпрограми із необхідними математичними операціями: обчисленням оберненого елементу за модулем із використанням розширеного алгоритму Евкліда, розв'язуванням лінійних порівнянь. При розв'язуванні порівнянь потрібно коректно обробляти випадок із декількома розв'язками, повертаючи їх усі.
2. За допомогою програми обчислення частот біграм, яка написана в ході виконання комп'ютерного практикуму №1, знайти 5 найчастіших біграм запропонованого шифртексту (за варіантом).
3. Перебрати можливі варіанти співставлення частих біграм мови та частих біграм шифртексту (розглядаючи пари біграм із п'яти найчастіших). Для кожного співставлення знайти можливі кандидати на ключ (a,b) шляхом розв'язання системи (1).
4. Для кожного кандидата на ключ дешифрувати шифртекст. Якщо шифртекст не є змістовним текстом російською мовою, відкинути цього кандидата.
5. Повторювати дії 3-4 доти, доки дешифрований текст не буде змістовним.

Хід роботи:

1. На першому етапі лабораторної роботи потрібно реалізувати підпрограми із необхідними математичними операціями: обчисленням оберненого елементу за модулем із використанням розширеного алгоритму Евкліда, розв'язуванням лінійних порівнянь. Цим ми спочатку і зайнялись, використовуючи теоретичні вказівки:

Нагадаємо, що алгоритм Евкліда обчислює найбільший спільний дільник двох чисел $d = \gcd(a,b)$ таким чином. Задаємо $r_0 = a$, $r_1 = b$ та обчислюємо послідовність (r_i) для $i \geq 2$ шляхом ділення з остачею:

$$r_0 = r_1 q_1 + r_2,$$
$$r_1 = r_2 q_2 + r_3,$$
$$\dots$$
$$r_{s-2} = r_{s-1} q_{s-1} + r_s;$$
$$r_{s-1} = r_s q_s.$$

Якщо на відповідному кроці виявилось, що $r_{s+1} = 0$, то $d = r_s$.

Розширений алгоритм Евкліда обчислює дві додаткові послідовності (u_i) та (v_i) такі, що на кожному кроці виконується рівність $r_i = u_i a + v_i b$; зокрема, для найбільшого

спільного дільника матимемо $d = r_s = u_s a + v_s b$. Ці послідовності також можна обчислити рекурентно за допомогою часток q_i:

$$u_0 = 1,\ u_1 = 0,\ u_{i+1} = u_{i-1} - q_i u_i;$$
$$v_0 = 0,\ v_1 = 1,\ v_{i+1} = v_{i-1} - q_i v_i.$$

Звідси обернений елемент до числа a за модулем n знаходиться таким чином: оскільки a обертається лише за умови $\gcd(a,n) = 1$, то за розширеним алгоритмом Евкліда знаходяться такі числа u та v, що $au + nv = 1$. Звідси $au \equiv 1 \pmod n$ та $u \equiv a^{-1} \pmod n$.

Розширений алгоритм Евкліда ми реалізували у функції `gcd`, яка приймає як аргументи числа a і b, НСД яких потрібно знайти. При цьому також шукається коефіцієнт u, який і буде оберненим числом по модулю. Ця функція повертає значення НСД і коефіцієнт u (тобто обернене число по модулю) як пару значень. Ось декілька тестів цієї команди:

```
print(gcd(103,13))
```

Вивід:

```
(1, 8)
```

```
print(gcd(110,14))
```

Вивід:

```
(2, None)
```

Нехай $ax \equiv b \pmod n$ і треба встановити значення x за відомими a та b. Маємо такі випадки:

1) $\gcd(a,n) = 1$. В цьому випадку порівняння має один розв'язок: $x \equiv a^{-1} b \pmod n$.
2) $\gcd(a,n) = d > 1$. Маємо дві можливості:
   2.1) Якщо b не ділиться на d, то порівняння не має розв'язків.
   2.2) Якщо b ділиться на d, то порівняння має рівно d розв'язків x_0, $x_0 + n_1$, $x_0 + 2n_1 \ldots$, $x_0 + (d-1)n_1$, де $a = a_1 d$, $b = b_1 d$, $n = n_1 d$ і x_0 є єдиним розв'язком порівняння $a_1 x \equiv b_1 \pmod{n_1}$: $x_0 = b_1 \cdot a_1^{-1} \bmod n_1$.

Вирішення лінійних порівнянь ми реалізували у функції `solve_expression`, яка приймає як аргументи числа a, b і n з типового лінійного порівняння і повертає число x:

$$ax \equiv b \pmod n$$

Ось приклади того, як функція поводить себе у всіх трьох випадках рівняння:

1) gcd(n, a) = 1

```
print(solve_expression(5,13,21))
```

```
11
```

2) gcd(n, a) = d>1, b не ділиться на d

```
print(solve_expression(328,20,96))
```

```
no solution
None
```

3) gcd(n, a) = d>1, b ділиться на d

```
print(solve_expression(14,21,7))
```

```
[0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0]
```

2. Тепер використовуючи функції з лабораторної роботи 1 (а саме: `clean_text` і `find_freq_without_overlap`), подивимось, які 5 біграм найчастіше зустрічаються у зашифрованому тексті:

```
загальна   відсоток від
кількість  загального
рн: 62,    %= 2.537
ыч: 41,    %= 1.678
нк: 34,    %= 1.391
цз: 32,    %= 1.309
иа: 30,    %= 1.227
```

Бачимо, що це біграми «рн», «ыч», «нк», «цз» і «иа». Зробили функцію `get_top_five`, яка буде з тексту відразу виводити тільки оці 5 біграм, які найчастіше зустрічаються (щоб можна було будь-який текст запихнути і відразу отримати 5 найчастіших біграм), і буде запихувати ці біграми в список:

```
['рн', 'ыч', 'нк', 'цз', 'иа']
```

3. З методичних вказівок можна дізнатися, що в російській мові найчастішими біграмами є «ст», «но», «то», «на» і «ен». У нас тепер є два списка: 5 найчастіших біграм в шифртексті і 5 найчастіших біграм у відкритому тексті. Для того, щоб ці біграми перетворити в числа, скористуємося формулою, наданою в теоретичних відомостях:

$$(x_{2i-1}, x_{2i}) \leftrightarrow X_i = x_{2i-1}m + x_{2i}.$$

Для цього спочатку створили функцію `convert_letter_to_number`, яка приймає на вхід букву і повертає її індекс в алфавіті. І також створили функцію get_Xi, яка приймає першу букву біграми і другу (тобто їх числа), а також кількість букв в алфавіті (за замовчуванням стоїть 31 – кількість букв в російському алфавіті), і переводить біграму у число.

Тепер в нас є два списки переведених у числа біграм:

```
PS D:\uni year 2\aks labi\laba 1> & C:/Users/Igorm/AppData/Local/Microsoft/Wind
['рн', 'ыч', 'нк', 'цз', 'иа'] шифр текст [545, 417, 572, 403, 168]
['ст', 'но', 'то', 'на', 'ен'] відкритий текст [509, 860, 413, 689, 248]
PS D:\uni year 2\aks labi\laba 1>
```

Щоб знайти всі можливі кандидати ключів за системою рівнянь

$$\begin{cases} Y^* \equiv aX^* + b \pmod{m^2} \\ Y^{**} \equiv aX^{**} + b \pmod{m^2} \end{cases},$$

, зробили функції `get_combinations` і `find_keys`, які знаходять всі комбінації систем рівнянь і знаходять можливі ключі (якщо наприклад рівняння немає розв'язку, то цю комбінацію пропускаємо). Таким чином змогли отримати дуже багато кандидатів на ключ:

```
230 89
241 821
389 885
445 156
731 319
11 631
159 695
215 927
720 101
950 642
148 477
204 709
572 313
802 854
813 625
56 921
516 601
746 181
757 913
905 16
```

4. Спробуємо розшифрувати текст всіма ключами, які ми знайшли за формулою:

$$X_i = a^{-1}(Y_i - b) \bmod m^2$$

Для початку ми хотіли б просто подивитися, чи є там щось змістовне, а потім вже зробимо розпізнавач російської мови. Для дешифрування написали функцію `decipher_bi`, яка приймає пару (a, b) і шифртекст, а повертає розшифрований текст. Спробували для кожного ключа вивести перші рядки розшифрованого тексту і шось знайшли таке: