КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №3

Криптоаналіз афінної біграмної підстановки

Виконали:

ФБ-31 Голомовза Дар`я

ФБ-31 Караман Любов

Варіант 7

Мета роботи:

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

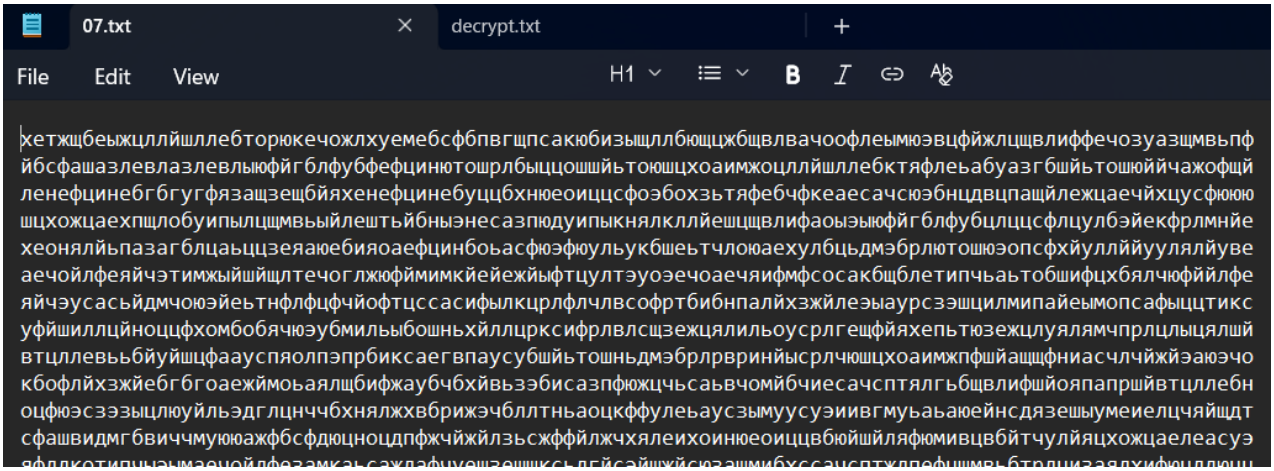
Порядок виконання роботи

0. Уважно прочитати методичні вказівки до виконання комп'ютерного практикуму. Нам надається текст, що є результатом шифрування за допомогою афінної підстановки біграм відкритого тексту, написаного російською мовою без пробілів, знаків пунктуації та великих літер. Буква «ё» заміщена буквою «е», а «ъ» – буквою «ь» (або навпаки). Таким чином, алфавіт відкритого тексту складається з 31 букви, що занумеровані в алфавітному порядку: а = 0 , б =1, …, я = 30 .

Найчастіші біграми: **«ст», «но», «то», «на», «ен».**

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.

Маємо такий зашифрований текст за Варіантом 7




```python
def egcd(a, b):
    if a == 0:
        return b, 0, 1
    else:
        g, y, x = egcd(b % a, a)
        return g, x - (b // a) * y, y

def modinv(a, mod):
    g, x, y = egcd(a, mod)
    if g != 1:
        return None
    return x % mod
```

1. egcd(a, b) - Розширений алгоритм Евкліда

Знаходить НСД двох чисел та коефіцієнти Безу
Вхід: a, b
Вихід: (НСД, x, y) де a·x + b·y = НСД

2. modinv(a, mod) - Обернений елемент за модулем

Знаходить число a^{-1} таке, що $a \times a^{-1} \equiv 1 \pmod m$
Повертає None якщо оберненого не існує

```python
def solve_linear(a, b, mod):
    """Розв'язує a*x ≡ b (mod m), повертає список можливих x"""
    g, _, _ = egcd(a, mod)
    if b % g != 0:
        return []
    a1, b1, m1 = a // g, b // g, mod // g
    inv_a1 = modinv(a1, m1)
    if inv_a1 is None:
        return []
    x0 = (inv_a1 * b1) % m1
    return [(x0 + i * m1) % mod for i in range(g)]
```

3. solve_linear(a, b, mod) - Розв'язок лінійного порівняння

Розв'язує a·x ≡ b (mod m)
Повертає список всіх можливих розв'язків
Якщо розв'язків немає - повертає пустий список

2. За допомогою програми обчислення частот біграм, яка написана в ході виконання комп'ютерного практикуму №1, знайти 5 найчастіших біграм запропонованого шифртексту (за варіантом).

```
ТОП-5 біграм шифртексту: ['цл', 'ял', 'ае', 'ле', 'чо']
```

Наша функція відображає скільки разів біграма зустрілась в шифротексті, і 5 найчастіших виводить на екран, також ми знаємо які найчастіші біграми для природньої мови це: «ст», «но», «то», «на», «ен», тому далі будемо співставляти наші біграми шифротексту і природньої мови, та аналізувати.

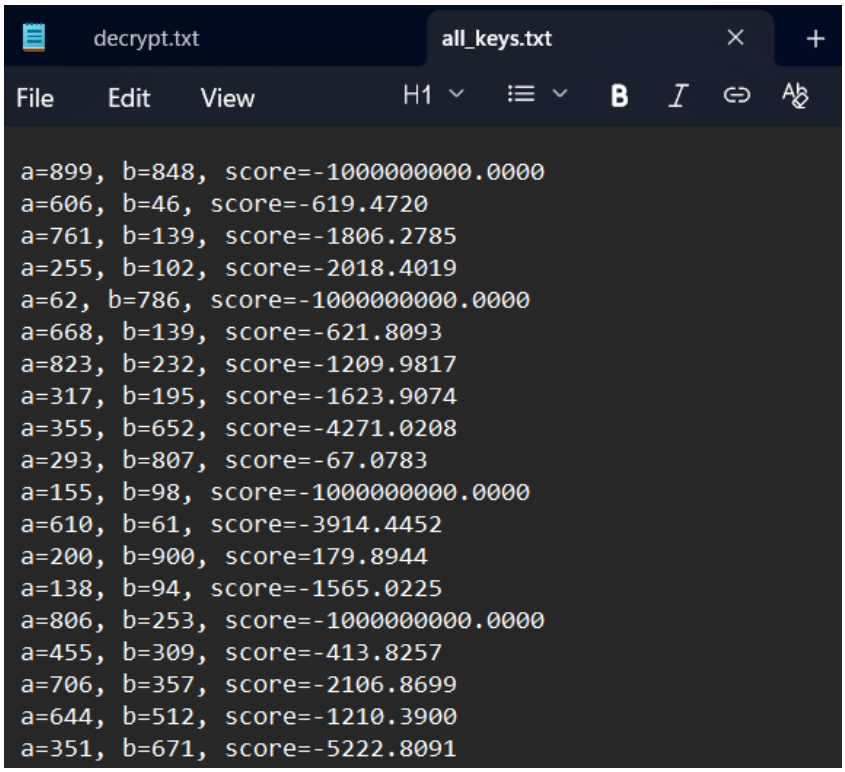
3. Перебрати можливі варіанти співставлення частих біграм мови та частих біграм шифртексту (розглядаючи пари біграм із п'яти найчастіших). Для кожного співставлення знайти можливі кандидати на ключ (a,b) шляхом розв'язання системи (1).

```
=== СПІВСТАВЛЕННЯ БІГРАМ ТА ПОШУК КЛЮЧІВ ===

[1] Мова: (ст, но) ↔ Шифртекст: (цл, ял)
[2] Мова: (ст, но) ↔ Шифртекст: (цл, ае)
[3] Мова: (ст, но) ↔ Шифртекст: (цл, ле)
[4] Мова: (ст, но) ↔ Шифртекст: (цл, чо)
[5] Мова: (ст, но) ↔ Шифртекст: (ял, цл)
[6] Мова: (ст, но) ↔ Шифртекст: (ял, ае)
[7] Мова: (ст, но) ↔ Шифртекст: (ял, ле)
[8] Мова: (ст, но) ↔ Шифртекст: (ял, чо)
[9] Мова: (ст, но) ↔ Шифртекст: (ае, цл)
[10] Мова: (ст, но) ↔ Шифртекст: (ае, ял)

=== ПІДБІР ЗАВЕРШЕНО ===
```

```
Усі ключі та співставлення збережено у all_keys.txt
```



4. Для кожного кандидата на ключ дешифрувати шифртекст. Якщо шифртекст не є змістовним текстом російською мовою, відкинути цього кандидата.

```
[НАЙКРАЩИЙ КЛЮЧ]: a = 200, b = 900, score = 179.89

--- РОЗШИФРОВАНИЙ ТЕКСТ (перші 300 символів) ---

атызнаешьсколькоразмывэтомгодуиграливбейсболавпрошломавпозапрошломнист
ятвосемьразасколькоразячистилзубызадесятьлетжизнишестьтысячразарукимыл
ив

Повний текст збережено у decrypt.txt
Усі ключі та співставлення збережено у all_keys.txt
```

Всього ми маємо **N кандидатів** - це N можливих розшифрованих шифротекстів.
Щоб перевірити кожен вручну, потрібно витратити значний людський ресурс - **час та уважність**.
Щоб автоматизувати цей процес, використовується **автомат розпізнавання мови**, який дозволяє оцінити, наскільки отриманий текст **нагадує природну українську мову**.

```python
def score_text(txt):
    """Оцінює якість розшифровки (чим вище, тим краще)"""
    vowels = "аеёиоуыэюя"
    if not txt:
        return -1e9
    freq = Counter(txt)
    v_ratio = sum(freq[ch] for ch in vowels) / len(txt)
    score = 0
    score -= abs(v_ratio - 0.45) * 100  # середній відсоток голосних
    score -= txt.count("ьь") * 100
    score -= txt.count("ыь") * 100
    score += sum(freq[ch] for ch in "еаотинсрл") / len(txt) * 300
    return score
```

Функція score_text(s) реалізує цей автомат. Вона перевіряє такі ознаки:

- **Частота частих літер** - перевіряється, наскільки часто в тексті трапляються типові для української мови літери («о», «а», «е», «і», «н», «т»). Їхня висока частка свідчить про природність тексту.

- **Частота рідкісних літер** — аналізується кількість маловживаних символів («ф», «щ», «ь»). Якщо їх занадто багато, текст імовірно не є осмисленим.

- **Співвідношення голосних** — визначається частка голосних літер («а», «е», «і», «о», «у», «я», «ю», «є», «ї»). Для природної мови вона має бути приблизно в межах 0.32–0.62.

- **Перевірка біграм** — знаходяться заборонені або нехарактерні для української мови сполучення літер («ьь», «ыь», «аь» тощо). Їх наявність знижує оцінку змістовності.

- **Комплексна оцінка (score)** — підсумковий бал формується з урахуванням усіх параметрів:

    - додаються бали за часті літери;
    - віднімаються бали за надлишок рідкісних літер і неправильну кількість голосних;
    - штрафуються заборонені біграми.

Це дозволяє перевірити, чи є текст "змістовним", тобто з оптимальним балансом поширених і рідкісних літер.

Шифрований текст:

```
хетжщбеыжцллйшллебторюкечожлхуемебсфбпвгщпсакюбизыщллбющцжбщвлвачоофлеымюэвцфйжлцщвлиффечозуазщмвьпф
йбсфашазлевлазлевлыюфйгблфубфефцинютошрлбыццошшйьтоюшцхоаимжоцллйшллебктяфлеьабуазгбшйьтошюййчажофщй
ленефцинебгбгугфязащзещбйяхенефцинебуццбхнюеоицсфоэбохзьтяфебчфкеаесачсюэбнцдвцпащйлежцаечйхцусфююю
шцхожцаехпщлобуипылцщмвьыйлештьйбныэнесазпюдуипыкнялкллйешцщвлифаоыэыюфйгблфубцлццсфлцулбэйекфрлмнйе
хеонялйьпазагблцаьццзеяаюебияоаефцинбоьасфюэфюульукбшеьтчлоюаехулбцьдмэбрлютошюэопсфхйуллййуулялйуве
аечойлфеяйчэтимжыйшйщлтечоглжюфймимкйейежйыфтцултэуоэечоаечяифмфсосакбщблетипчьаьтобшифцхбялчюфййлфе
яйчэусасьйдмчоюэйеьтнфлфцфчйофтцссасифылкцрлфлчлвсофртбибнпалйхзжйлеэыаурсзэшцилмипайеымопсафыццтикс
уфйшиллцйноццфхомбобячюэубмильыбошньхйллцрксифрлвлсщзежцялильоусрлгещфйяхепьтюзежцлуялямчпрлцлыцялшй
```

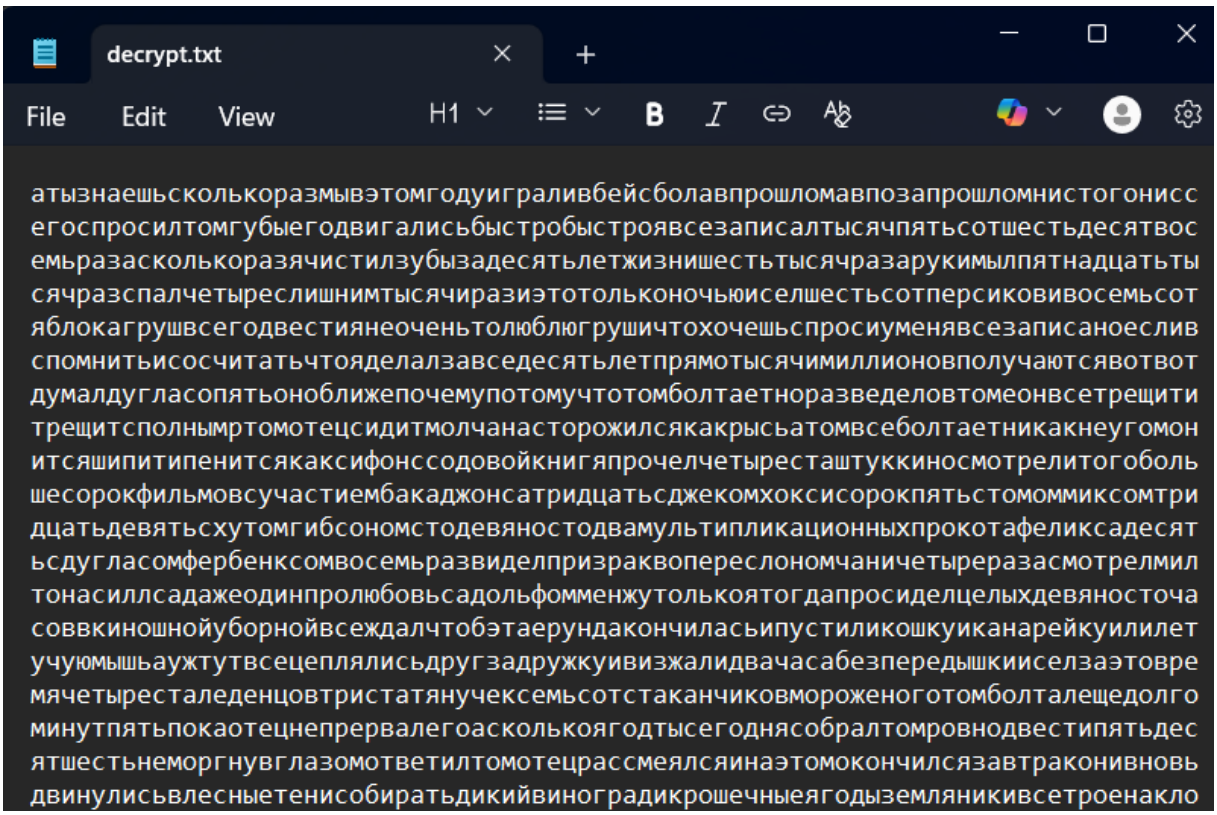
Дешифрування:

```
[НАЙКРАЩИЙ КЛЮЧ]: a = 200, b = 900, score = 179.89

--- РОЗШИФРОВАНИЙ ТЕКСТ (перші 300 символів) ---

атызнаешьсколькоразмывэтомгодуиграливбейсболавпрошломавпозапрошломнистог
ятвосемьразасколькоразячистилзубызадесятьлетжизнишестьтысячразарукимылпя
ив

Повний текст збережено у decrypt.txt
Усі ключі збережено у all_keys.txt
```



Висновки:

Ця робота дає змогу набути навичок виявлення криптографічних слабкостей моноалфавітної підстановки, зокрема за допомогою частотного аналізу біграм. Використання цього підходу дозволяє декодувати зашифровані повідомлення, знаходячи відповідність між поширеними біграмами в шифртексті та типовими біграмами мови. Це є важливим кроком для поглиблення розуміння криптографії, частотного аналізу та методів дешифрування в криптоаналізі. Робота також сприяє вивченню методів модулярної арифметики та лінійних рівнянь, а також розвитку практичних навичок у галузі комп’ютерної безпеки та криптографії.