Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Фізико-технічний інститут

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №3

Криптоаналіз афінної біграмної підстановки

Виконали:
ФБ-21 Худоба Арсен,
ФБ-21 Шабанов Кирило

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.

```
def mod_inverse(a, m):
    d, x, _ = extended_gcd(a, m := M**2)
    return x % m if d == 1 else None


3 usages
def extended_gcd(a, b):
    if b == 0: return a, 1, 0
    d, x1, y1 = extended_gcd(b, a % b)
    return d, y1, x1 - (a // b) * y1
```

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

```
def count_bigrams(text):
    return Counter([text[i:i+2] for i in range(len(text)-1)])
```

```
5 найчастіших біграм у шифротексті:
лл: 68
цл: 64
ул: 56
ле: 50
ял: 49
```

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).

```
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('ст', 'но'), Ключ: (155, 445)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('ст', 'то'), Ключ: (155, 445)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('ст', 'на'), Ключ: (248, 693)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('ст', 'ен'), Ключ: (124, 42)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('но', 'ст'), Ключ: (806, 600)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('но', 'на'), Ключ: (868, 693)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('но', 'ен'), Ключ: (620, 321)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('то', 'ст'), Ключ: (806, 600)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('то', 'на'), Ключ: (868, 693)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('то', 'ен'), Ключ: (620, 321)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('на', 'ст'), Ключ: (713, 352)
Шифр-біграми: ('лл', 'цл'), Мова-біграми: ('на', 'но'), Ключ: (93, 352)
```

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

Побудова автомату розпізнавання російської мови:

- перевірку частот **частих** літер **(«о», «а», «е»,** частоти можуть розглядатись окремо або в сукупності);

- перевірку частот **рідкісних** літер (**«ф», «щ», «ь»**, частоти також можуть розглядатись окремо або в сукупності);
- перевірку частот біграм, підраховану для біграм «на перетині» (у вищенаведених позначеннях – біграм виду $(x_{2i}, x_{2i+1}))$
- перевірку частот триграм та довільних l-грам.

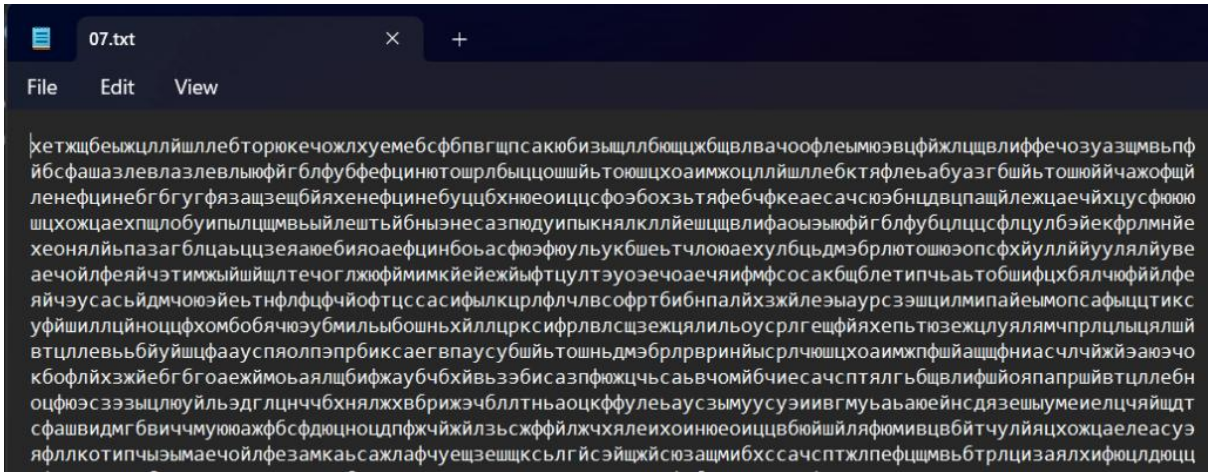
```
# Автомат розпізнавання тексту
1 usage
def is_meaningful_text(text):
    freq_letters = Counter(text)
    total = sum(freq_letters.values())
    common = sum(freq_letters[char] for char in {"о", "а", "е"})
    rare = sum(freq_letters[char] for char in {"ф", "щ", "ь"})
    return common / total >= 0.2 and rare / total <= 0.05
```

Ця функція `is_meaningful_text` перевіряє, чи є текст змістовним на основі частоти використання певних літер.

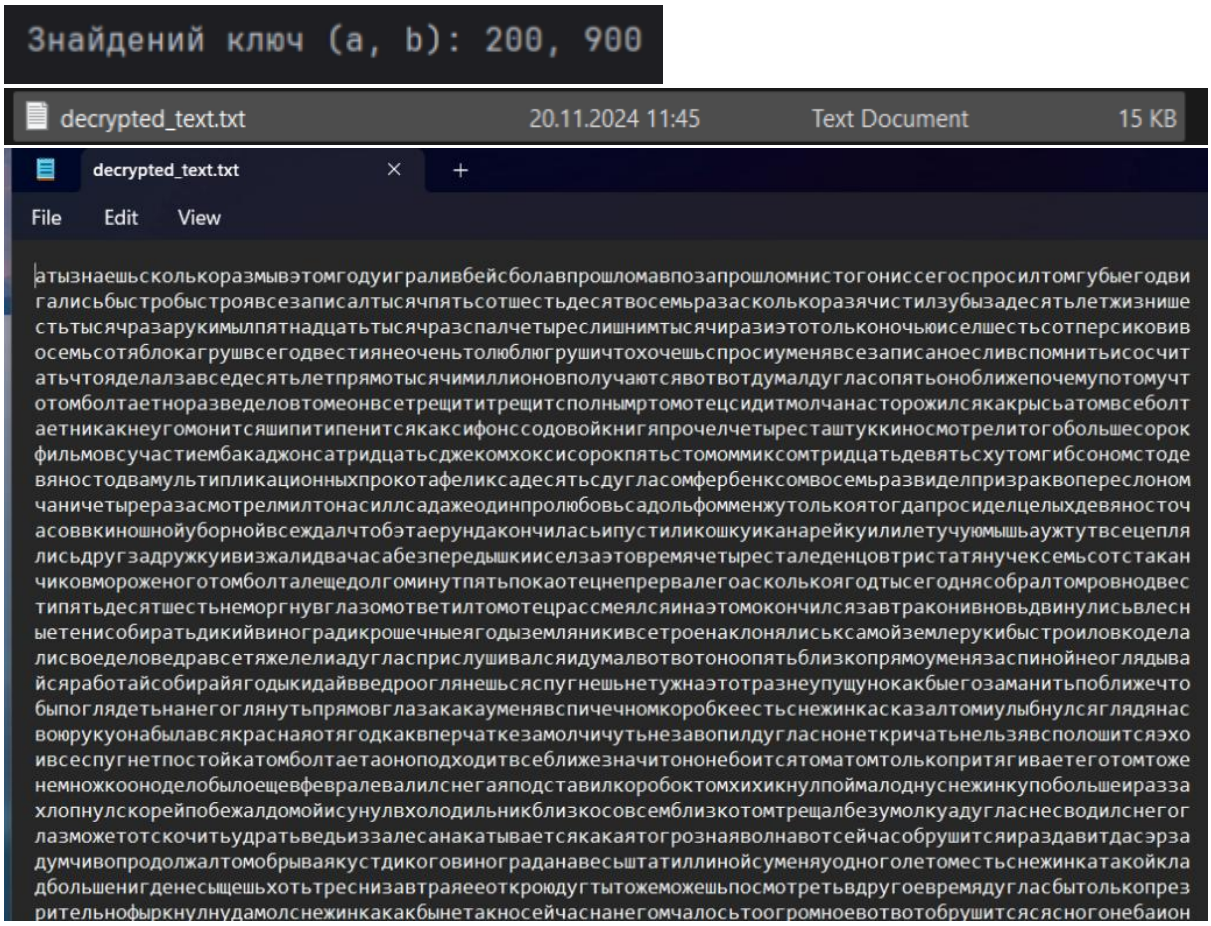
1. **Підрахунок частоти літер**:
   - Використовуємо `Counter(text)` для підрахунку кількості кожної літери в рядку `text`. Це створює словник, де ключі — це літери, а значення — їхня кількість.
2. **Загальна кількість літер**:
   - `total = sum(freq_letters.values())` підраховує загальну кількість літер у тексті (сума всіх значень у словнику).
3. **Частота поширених літер**:
   - `common = sum(freq_letters[char] for char in {"о", "а", "е"})` підраховує загальну кількість літер "о", "а" та "е" в тексті, використовуючи множину для визначення, які літери рахувати.
4. **Частота рідкісних літер**:
   - `rare = sum(freq_letters[char] for char in {"ф", "щ", "ь"})` підраховує загальну кількість літер "ф", "щ" та "ь".
5. **Перевірка умов**:
   - Функція повертає `True`, якщо відношення частоти поширених літер до загальної кількості літер більше або рівне 0.2 (тобто понад 20% тексту складається з літер "о", "а" та "е"), а відношення частоти рідкісних літер до загальної кількості літер менше або рівне 0.05 (менше 5% тексту складається з літер "ф", "щ" та "ь").

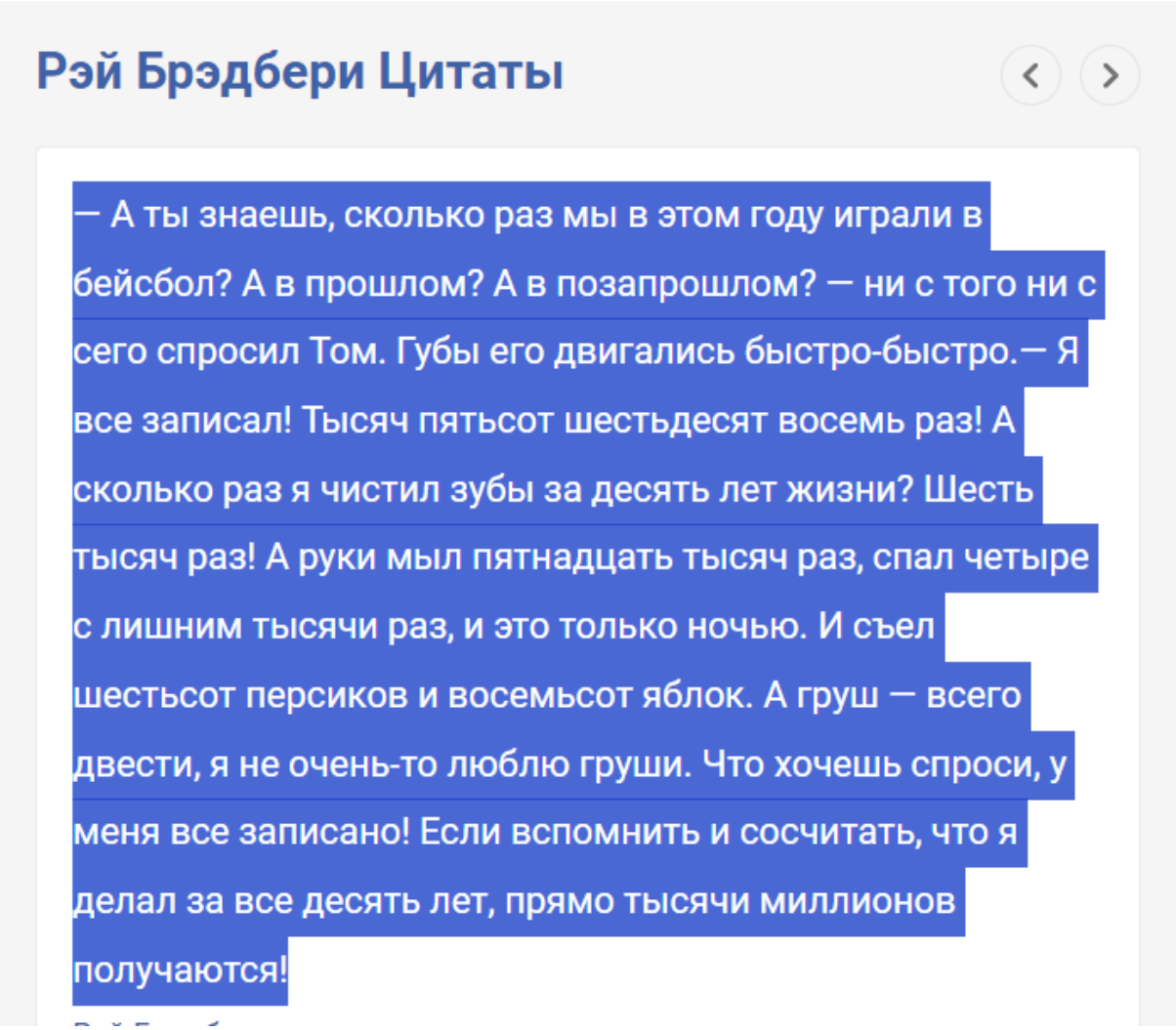
Це дозволяє перевірити, чи є текст "змістовним", тобто з оптимальним балансом поширених і рідкісних літер.

Шифрований текст:



Дешифрування:





Висновки: виконання цієї роботи дозволяє здобути навички виявлення криптографічних слабкостей моноалфавітної підстановки, зокрема через частотний аналіз біграм. Такий підхід дозволяє декодувати зашифровані повідомлення, виявляючи відповідність між часто вживаними біграмами в шифртексті та природними біграмами мови. Робота є важливим етапом для розвитку розуміння криптографії, частотного аналізу та методів дешифрування в криптоаналізі. Вона сприяє глибокому засвоєнню методів модулярної арифметики та лінійних рівнянь, а також розвитку практичних навичок у галузі комп’ютерної безпеки та криптографії.