МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. ІГОРЯ СІКОРСЬКОГО”

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №3

Криптоаналіз афінної біграмної підстановки

Виконали:

Студенти групи ФБ-22

Шеїна Еліна, Мартинюк Артем

Варіант - 1

КИЇВ 2024

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

Спочатку створимо функцію, яка буде обчислювати обернений до a елемент:

```
def compute_extended_gcd(a, m):
    if a == 0:
        return m, 1, 0
    gcd, x, y = compute_extended_gcd(m % a, a)
    coefficient = x - (m // a) * y
    return gcd, y, coefficient

def find_modular_inverse(a, m):
    gcd, inverse, coefficient = compute_extended_gcd(a, m)
    result = [gcd, inverse, coefficient]
    return result
```

Функція find_modular_inverse реалізує розширений алгоритм Евкліда, приймаючи на вхід число a та модуль, і повертає список, що містить значення, серед яких gcd та обернений елемент.

Далі, створюємо функцію, яка вирішує лінійне порівняння:

```python
def solve_congruence(a, b, m):
    inverse_result = find_modular_inverse(a, m)
    gcd = inverse_result[0]

    if gcd == 1:
        coefficient = inverse_result[2]
        return (b * coefficient) % m
    elif gcd > 1 and b % gcd == 0:
        reduced_inverse = find_modular_inverse(a // gcd, m // gcd)
        reduced_gcd = reduced_inverse[0]
        coefficient = reduced_inverse[2]
        base_solution = (b // gcd * coefficient) % (m // gcd)
        return [base_solution + (m // gcd) * i for i in range(gcd)]
```

Приймає a, b та модуль, повертає або одне значення, або декілька, якщо gcd>1

Знаходимо найчастіші біграми з ШТ:

```python
from collections import Counter
```

✓ 0.0s

```python
def most_frequent_bigrams_with_relative_frequency(filepath):
    with open(filepath, 'r', encoding='utf-8') as file:
        content = file.read()
    filtered_text = ''.join(char.lower() for char in content if char.isalnum() or char.isspace())
    bigrams = [filtered_text[i:i+2] for i in range(len(filtered_text) - 1)]
    bigram_frequency = Counter(bigrams)
    total_count = sum(bigram_frequency.values())
    top_5_bigrams = bigram_frequency.most_common(5)

    for bigram, count in top_5_bigrams:
        frequency_percentage = count / total_count
        print(f"'{bigram}': {count} (frequency: {frequency_percentage:.2%})")

most_frequent_bigrams_with_relative_frequency('01.txt')
```

Вивід:

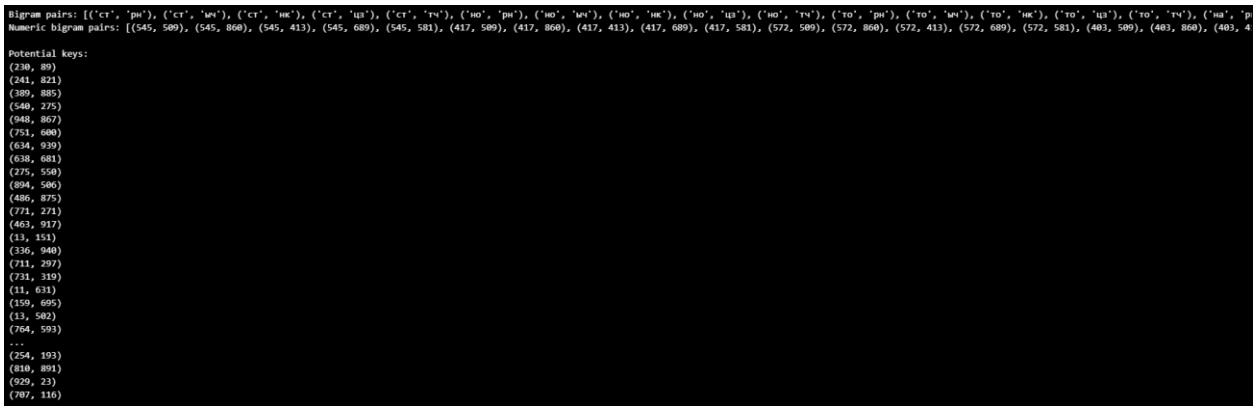
```
'рн': 63 (frequency: 1.28%)
'ыч': 44 (frequency: 0.89%)
'нк': 43 (frequency: 0.87%)
'цз': 37 (frequency: 0.75%)
'тч': 33 (frequency: 0.67%)
```

Біграми «рн», «ыч», «нк», «цз», «тч».

Пишемо код, який зіставляє найбільш поширені біграми ШТ з найпоширенішими біграмами російської мови, наведеними у методичних матеріалах. Потім цей код перетворює кожну біграму з пари в числове значення за формулою: перша_літера*31 + друга_літера. Далі обчислює пару ключів (a, b) для кожної комбінації з двох пар значень:

```
cipher_bigrams = ['рн', 'ыч', 'нк', 'цз', 'тч']
russian_bigrams = ['ст', 'но', 'то', 'на', 'ен']
alphabet = 'абвгдежзийклмнопрстуфхцчшщьыэюя'
letter_to_index = {char: idx for idx, char in enumerate(alphabet)}
modulus = 961

bigram_pairs = [
    (russian_bigram, cipher_bigram)
    for russian_bigram in russian_bigrams
    for cipher_bigram in cipher_bigrams
]
print("Bigram pairs:", bigram_pairs)

numeric_bigram_pairs = [
    (
        31 * letter_to_index[pair[0][0]] + letter_to_index[pair[0][1]],
        31 * letter_to_index[pair[1][0]] + letter_to_index[pair[1][1]]
    )
    for pair in bigram_pairs
]
print("Numeric bigram pairs:", numeric_bigram_pairs)

potential_keys = []
for i in range(len(numeric_bigram_pairs)):
    X1, Y1 = numeric_bigram_pairs[i]

    for j in range(i + 1, len(numeric_bigram_pairs)):
        X2, Y2 = numeric_bigram_pairs[j]
        delta_X = X1 - X2
        delta_Y = Y1 - Y2

        if delta_X == 0:
            continue

        try:
            inverse_result = find_modular_inverse(delta_X, modulus)
            inv_delta_X = inverse_result[2]
            a = (delta_Y * inv_delta_X) % modulus
            b = (Y1 - a * X1) % modulus
            key = solve_congruence(a, b, modulus)
            if key is not None:
                potential_keys.append((a, b))
        except ValueError:
            continue

print("\nPotential keys:")
for key in potential_keys:
```

Вивід:



Обчислювали ключ за допомогою цих формул:

$$Y^* - Y^{**} \equiv a(X^* - X^{**}) \pmod{m^2}. \qquad b = (Y^* - aX^*) \bmod m^2.$$

Далі, для кожного ключа здійснюємо дешифрування тексту за допомогою оберненого афінного шифру. Отримане число перетворюється в біграму за допомогою оберненої схеми до формули (перша_літера*31 + друга_літера):

```python
char_to_index = {char: idx for idx, char in enumerate(alphabet)}
index_to_char = {idx: char for idx, char in enumerate(alphabet)}

with open('01.txt', 'r', encoding='utf-8') as file:
    cipher_text = file.read().replace('\n', '').replace('\r', '')
cipher_indexes = [char_to_index[char] for char in cipher_text]

def affine_decrypt(cipher_indexes, a, b, mod):
    mod_squared = mod ** 2
    decrypted_output = []

    try:
        inverse_result = find_modular_inverse(a, mod_squared)
        a_inverse = inverse_result[2]
    except ValueError:
        print(f"Inverse not found. Skipping key.")
        return None

    for i in range(0, len(cipher_indexes), 2):
        y = cipher_indexes[i]
        next_y = cipher_indexes[i + 1] if i < len(cipher_indexes) - 1 else 0
        combined_index = y * 31 + next_y
        x = (a_inverse * (combined_index - b)) % mod_squared
        x1, x2 = (x // 31), (x % 31)

        decrypted_output.append(index_to_char[x1 % mod])
        decrypted_output.append(index_to_char[x2 % mod])

    return ''.join(decrypted_output)

decrypted_texts = {}
for a, b in potential_keys:
    decryption_result = affine_decrypt(cipher_indexes, a, b, len(alphabet))
    if decryption_result:
        decrypted_texts[(a, b)] = decryption_result
```

Записавши масив рідкісних біграм російської мови, розбиваємо кожен розшифрований текст на біграми та порівнюємо їх з рідкісними. Потім сортуємо список розшифрованих текстів за кількістю рідкісних біграм у порядку зростання і визначаємо найбільш ймовірний ключ – (13, 151).

```
rare_bigrams = ["щт", "ьо", "ыж", "юв", "яы", "аы", "бй", "гй", "дй", "еы", "шщ", "шя", "щб", "щд", "щк", "ьы", "ыа", "ыь", "ыы", "ыэ"]

def split_text_into_bigrams(text):
    return [text[i:i+2] for i in range(0, len(text), 2)]

def count_rare_bigrams_in_text(text, rare_bigrams_list):
    bigrams = split_text_into_bigrams(text)
    return sum(1 for bigram in bigrams if bigram in rare_bigrams_list)

bigram_frequency_count = {}
for key, decrypted_text in decrypted_texts.items():
    rare_bigram_count = count_rare_bigrams_in_text(decrypted_text, rare_bigrams)
    bigram_frequency_count[key] = rare_bigram_count


sorted_bigrams_by_count = sorted(bigram_frequency_count.items(), key=lambda x: x[1])

print("Ключ з найменшою кількістю рідкісних біграм:")
for i, (key, count) in enumerate(sorted_bigrams_by_count[:1]):
    print(f"{i + 1}. Ключ {key} - Кількість рідкісних біграм: {count}")

✓ 0.1s
Ключ з найменшою кількістю рідкісних біграм:
1. Ключ (13, 151) - Кількість рідкісних біграм: 3
```

```
end_key = (13, 151)
print(decrypted_texts[end_key])

✓ 0.0s
многограннуюличностьдостоевскогоможнорассматриватьсчетырехсторонкакписателя
```

Текст розшифровано.

Розшифрований текст:

многограннуюличностьдостоевскогоможнорассматриватьсчетырехсторонкакписателякакн
евротикакакмыслителяэтикаикакгрешникакакжеразобратьсявэтойневольносмущающейнас
сложностинаименееспоренонкакписательместоеговодномрядусшекспиромбратьякарамазо
вывеличайшийроманизвсехкогдалибонаписанныхалегендаовеликоминквизитореодноизвы
сочайшихдостижениймировойлитературыпереоценитькотороеневозможноксожалениюпер
едпроблемойписательскоготворчествапсихоанализдолженсложитьоружиедостоевскийскор
еевсегоуязвимкакморалистпредставляяегочеловекомвысоконравственнымнатомосновании
чтотолькототдостигаетвысшегонравственногосовершенствактопрошелчерезглубочайшиеб
ездныгреховностимыигнорируемодносоображениеведьнравственнымявляетсячеловекреаг
ирующийуженавнутреннеиспытываемоеискушениеприэтомемунеподдаваяськтожепопере
меннотогрешиттораскаиваясьставитсебевысокиенравственныецелитоголегкоупрекнутьвто
мчтоонслишкомудобнодлясебястроитсвоюжизньоннеисполняетосновногопринципанравст
венностинеобходимостиотречениявтовремякакнравственныйобразжизнивпрактическихин
тересахвсегочеловечестваэтимоннапоминаетварваровэпохипереселениянародовварварову
бивавшихизатемкаявшихсявэтомтакчтопокаяниестановилосьтехническимпримеромрасчи
щавшимпутькновымубийствамтакжепоступаливангрозныйэтасделкассовестьюхарактерна
ярусскаячертадостаточнобесславениконечныйитогнравственнойборьбыдостоевскогопосле
исступленнойборьбывоимяпримиренияпритязанийпервичныхпозывовиндивидастребовани
ямичеловеческогообществаонвынужденнорегрессируеткподчинениюмирскомуидуховном
уавторитетукпоклонениюцарюихристианскомубогукрусскомумелкодушномунационализм
укчемуменеезначительныеумыпришлисгораздоменьшимиусилиямичемонвэтомслабоемест
обольшойличностидостоевскийупустилвозможностьстатьучителемиосводителемчелове
честваиприсоединилсяктюремщикамкультурабудущегонемногимбудетемуобязанавэтомпо
всейвероятностипроявилсяегоневрозиззакоторогоонибылосужденнатакуюнеудачупомощи
постиженияисилелюбвиклюдямемубылоткрытдругойапостольскийпутьслужениянампредс
тавляетсяотталкивающимрассматриваниедостоевскоговкачествегрешникаилипреступника
ноэтоотталкиваниенедолжноосновыватьсянаобывательскойоценкепреступникавыявитьпо
длиннуюмотивациюпреступлениянедолгодляпреступникасущественныдвечертыбезгранич
ноесебялюбиеисильнаядеструктивнаясклонностьобщимдляобеихчертипредпосылкойдляи
хпроявленийявляетсябезлюбовностьнехваткаэмоциональноооценочногоотношениякчелове
кутутсразувспоминаешьпротивоположноеэтомуудостоевскогоегобольшуюпотребностьвл
юбвииегоогромнуюспособностьлюбитьпроявившуюсявегосверхдоброteипозволявшуюему
любитьипомогатьтамгдеонимелбыправоненавидетьимститьнапримерпоотношениюкегопе
рвойженеиеелюбовникунотогдавозникаетвопросоткудаприходитсоблазнпричисленияодост
оевскогокпреступникамответиззавыбораегосюжетовэтопреимущественнонасильникиубий
цыэгоцентрическиехарактерычтосвидетельствуетосуществованиитакихсклонностейвеговн
утреннеммиреатакжеиззанекоторыхфактовегожизнистрастиегоказартнымиграмможетбыть
сексуальногорастлениянезрелойдевочкиисповедьэтопротиворечиеразрешаетсяследующим
образомсильнаядеструктивнаяустремленностьдостоевскогокотораямоглабыделатьегопре
ступникомбылавегожизнинаправленаглавнымобразомнасамогосебявовнутрьвместотогочт
обыизнутриитакимобразомвыразиласьвмазохизмеичувствевинывсетакивеголичностинема
лоисадистическихчертвыявляющихсявегораздражительностимучительстванетерпимостид
ажепоотношениюклюбимымлюдяматакжевегоманереобращениясчитателемитаквмелочахо
нсадиствовневажномсадистпоотношениюксамомусебеследовательномазохистиэтомягчай

шийдобродушнейшийвсегдаготовыйпомочьчеловеквсложнойличностидостоевскогомывы
делилитрифактораодинколичественныйидвакачественныхегочрезвычайноповышеннуюаф
фективностьегоустремленностькперверзиикотораядолжнабылапривестиегоксадомазохизм
уилисделатьпреступникомиегонеподдающееясянализутворческоедарованиетакоесочетани
евполнемоглобысуществоватьибезневрозаведьбываютжестопроцентныемазохистыбезнали
чияневрозовпосоотношениюсилпритязаниипервичныхпозывовипротивоборствующихимто
рможенийприсоединяясюдавозможностисублимированиядостоевскоговсеещеможнобылоб
ыотнестикразрядуимпульсивныххарактеровноположениевещейзатемняетсяналичиемневро
занеобязательногокакбылосказаноприданныхобстоятельствахновсежевозникающеготемск
ореечемнасыщенноеосложнениеподлежащеесосторонычеловеческогояпреодолениюневроз
этотолькознактогочтоятакойсинтезнеудалсячтоонопритойпопыткепоплатилосьсвоимедин
ствомвчемжевстрогомсмыслепроявляетсяневроздостоевскийназывалсебясамидругиетакже
считалиегоэпилептикомнатомоснованиичтоонбылподвержентяжелымприпадкамсопровож
давшимисяпотерейсознаниясудорогамиипоследующимупадочнымнастроениемвесьмавероя
тночтоэтатакназываемаяэпилепсиябылалишьсимптомомегоневрозакоторыйвтакомслучае
следуетопределитькакистероэпилепсиютоестькактяжелуюистериюутверждатьэтосполнойу
веренностьюнельзяподвумпричинамвопервыхпотомучтодатыанамнезическихприпадковта
кназываемойэпилепсиидостоевскогонедостаточныиненадежныавовторыхпотомучтопоним
аниесвязанныхсэпилептоидными припадкамиболезненныхсостоянийостаетсянеясныма