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

Орлов Антон, Ялбуган Федір

(бригада 7)

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 extended_gcd(a, mod):
    if a == 0:
        return mod, 1, 0
    gcd, x, y = extended_gcd(mod % a, a)
    c = x - (mod // a) * y
    return gcd, y, c

def modular_inverse(a, mod):
    gcd, y, c = extended_gcd(a, mod)
    answ = [gcd, y, c]
    return answ
```

Функція **modular_inverse** працює за розширеним алгоритмом Евкліда, приймає на вікд число a та модуль, повертає список зі знгаченнями, серед яких будуть gcd та сам обернений елемент.

Далі, створюємо функцію, яка вирішує лінійне порівняння:

```python
def solve_linear_congruence(a, b, mod):
    answ = modular_inverse(a, mod)
    gcd = answ[0]

    if gcd == 1:
        c = answ[2]
        return b * c % mod
    elif gcd > 1 and b % gcd == 0:
        answ = modular_inverse(int(a / gcd), int(mod / gcd))
        gcd = answ[0]
        c = answ[2]
        multi = int(b / gcd * c % mod)
        return [multi + int(mod / gcd) * i for i in range(gcd)]
```

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

Використовуючи код з першої лабораторної, знаходимо найчастіші біграми з ШТ:

```python
from collections import Counter

def top_character_bigrams_with_relative_frequency(filename):
    with open(filename, 'r', encoding='utf-8') as file:
        text = file.read()
    text = ''.join(c.lower() for c in text if c.isalnum() or c.isspace())
    bigrams = [text[i:i+2] for i in range(len(text) - 1)]
    bigram_counts = Counter(bigrams)
    total_bigrams = sum(bigram_counts.values())
    top_5_bigrams = bigram_counts.most_common(5)

    for bigram, count in top_5_bigrams:
        relative_frequency = count / total_bigrams
```

```
(function) def top_character_bigrams_with_relative_frequency(filename: Any) -> None
```

```python
top_character_bigrams_with_relative_frequency('07.txt')
```

```
✓ 0.0s                                                    Python
'лл': 67 (частота: 0.93%)
'цл': 64 (частота: 0.89%)
'ул': 56 (частота: 0.78%)
'ле': 50 (частота: 0.70%)
'ял': 49 (частота: 0.68%)
```

Як бачимо, в нашому випадку, це біграми **«лл», «цл», «ул», «ле», «ял»**.

Створюємо код, який би співставляв найчастіші біграми ШТ і найчастіши біграми російської мови, представлені у методичці. Далі, цей код перетворює кожну біграму з

пари у числові значення (за схемою: перша_літера*31 + друга_літера). Потім, цей код обчислює ключ (a, b) для кожної комбінації з двох пар значень:

```
cipher_bigrams = ['лл', 'цл', 'ул', 'ле', 'ял']
russian_bigrams = ['ст', 'но', 'то', 'на', 'ен']
alphabet = 'абвгдежзийклмнопрстуфхцчшщьыэюя'
letter_to_number = {letter: index for index, letter in enumerate(alphabet)}
m_squared = 961
pair_list = [(russian_bigram, cipher_bigram)
             for russian_bigram in russian_bigrams
             for cipher_bigram in cipher_bigrams]
print("Pair list:", pair_list)
numeric_pairs = [
    (
        31 * letter_to_number[pair[0][0]] + letter_to_number[pair[0][1]],
        31 * letter_to_number[pair[1][0]] + letter_to_number[pair[1][1]]
    )
    for pair in pair_list
]
print("Numeric pairs:", numeric_pairs)

keys = []
for i in range(len(numeric_pairs)):
    X1, Y1 = numeric_pairs[i]

    for j in range(i + 1, len(numeric_pairs)):
        X2, Y2 = numeric_pairs[j]
        delta_X = X1 - X2
        delta_Y = Y1 - Y2
        if delta_X == 0:
            continue

        try:
            answ = modular_inverse(delta_X, m_squared)
            inv_delta_X = answ[2]
            a = (delta_Y * inv_delta_X) % m_squared
            b = (Y1 - a * X1) % m_squared
            result = solve_linear_congruence(a, b, m_squared)
            if result is not None:
                keys.append((a, b))

        except ValueError:
            continue

print("\nFinal keys:")
for key in keys:
    print(key)
```

Вивід:

```
Pair list: [('ст', 'лл'), ('ст', 'цл'), ('ст', 'ул'), ('ст', 'ле'), ('ст', 'ял'), ('но',
Numeric pairs: [(545, 352), (545, 693), (545, 600), (545, 346), (545, 941), (417, 352),

Final keys:
(916, 852)
(107, 658)
(176, 532)
(181, 690)
(761, 139)
(262, 131)
(889, 532)
(57, 380)
(17, 945)
(45, 100)
(83, 532)
(615, 814)
(45, 807)
(200, 900)
(944, 1)
(138, 94)
(854, 40)
(699, 908)
(916, 846)
(761, 753)
```

Для обчислення ключа використовувались ці формули:

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

Далі, для кожного ключа дешифруємо текст за допомогою оберненого афінного шифру. Число, яке вийшло, перетворювалось у біграму за допомогою оберненої схеми до (перша_літера*31 + друга_літера):

```
char_to_num = {char: i for i, char in enumerate(alphabet)}
num_to_char = {i: char for i, char in enumerate(alphabet)}

with open('07.txt', 'r', encoding='utf-8') as file:
    cipher_text = file.read()
    cipher_text = cipher_text.replace('\n', '').replace('\r', '')
cipher_nums = [char_to_num[char] for char in cipher_text]

def affine_decrypt(cipher_nums, a, b, mod):
    m_squared = mod ** 2
    decrypted_text = []

    try:
        answ = modular_inverse(a, m_squared)
        a_inv = answ[2]
    except ValueError:
        print(f"Оберненого не існує. Пропускаем ключ.")
        return None

    for i in range(0, len(cipher_nums), 2):
        y = cipher_nums[i]
        next_y = cipher_nums[i + 1] if i < len(cipher_nums) - 1 else 0
        ind = y*31 + next_y
        x = (a_inv * (ind - b)) % m_squared
        x1, x2 = (x // 31), (x%31)

        decrypted_text.append(num_to_char[x1 % mod])
        decrypted_text.append(num_to_char[x2 % mod])

    return ''.join(decrypted_text)

decrypted_texts = {}
for a, b in keys:
    result = affine_decrypt(cipher_nums, a, b, len(alphabet))
    if result:
        decrypted_texts[(a, b)] = result
```

Виписавши масив з рідкісними біграмами російської мови, розділяємо кожен розшифрований зразок на біграми та порівнюємо їх з рідкісними. Сортуємо список розшифрованих текстів за зростанням кількості рідкісних біграм і отримуємо найбільш ймовірний ключ – **(200, 900)**.

```python
rare_bigrams = ["щт", "ьо", "ьж", "юв", "яы", "аы", "бй", "гй", "дй", "еы", "шщ", "шя", "щб", "щд", "щж",

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

def count_rare_bigrams(text, rare_bigrams):
    bigrams = split_into_bigrams(text)
    count = sum(1 for bigram in bigrams if bigram in rare_bigrams)
    return count

bigram_counts = {}
for key, decrypted_text in decrypted_texts.items():
    count = count_rare_bigrams(decrypted_text, rare_bigrams)
    bigram_counts[key] = count
sorted_bigram_counts = sorted(bigram_counts.items(), key=lambda x: x[1])
print("Top 10 ключів з найменшою кількістю рідкісних біграм:")
for i, (key, count) in enumerate(sorted_bigram_counts[:10]):
    print(f"{i+1}. Ключ {key} - Кількість рідкісних біграм: {count}")
```

✓ 0.1s Python

```
Top 10 ключів з найменшою кількістю рідкісних біграм:
1. Ключ (200, 900) - Кількість рідкісних біграм: 4
2. Ключ (688, 82) - Кількість рідкісних біграм: 10
3. Ключ (506, 160) - Кількість рідкісних біграм: 16
4. Ключ (754, 160) - Кількість рідкісних біграм: 18
5. Ключ (944, 1) - Кількість рідкісних біграм: 23
6. Ключ (413, 160) - Кількість рідкісних біграм: 23
7. Ключ (661, 160) - Кількість рідкісних біграм: 27
8. Ключ (521, 275) - Кількість рідкісних біграм: 31
9. Ключ (761, 753) - Кількість рідкісних біграм: 32
10. Ключ (781, 795) - Кількість рідкісних біграм: 33
```

Виводимо на екран вміст розшифрованого тексту цим ключем:

```python
final_key = (200, 900)
print(decrypted_texts[final_key])
```

✓ 0.0s

```
атызнаешьсколькоразмывэтомгодуиграливбейсболавпрошломавпозапрошломнистогониссегоспросилтомгубыегодви
```

Текст успішно розшифровано ☺

**ШТ:**

хетжщбеыжцллйшллебторюкечожлхуемебсфбпвгщпсакюбизыщллбющцжбщвлвачоофлеымюэвцф
йжлцщвлиффечозуазщмвьпфйбсфашазлевлазлевлыюфйгблфубфефцинютошрлбыццошшйьтоюшц
хоаимжоцллйшллебктяфлеьабуазгбшйьтошюййчажофщйленефцинебгбгугфязащзещбйяхенефцине
буццбхнюеоиццсфоэбохзьтяфебчфкеаесачсюэбнцдвцпащйлежцаечйхцусфююющцхожцаехпщлобу
ипылщщмвьыйлештьйбныэнесазпюдуипыкнялкллйешщщвлифаоыэыюфйгблфуцлщсфлщулбэйекф
рлмнйехеонялйьпазагблцаьццзеяаюебияоаефцинбоьасфюэфюульукбшеьтчлоюаехулбцьдмэбрлюто
шюэопсфхйуллййуулялйувеаечойлфеяйчэтимжыйшйщлтечоглжюфймимкйейежйыфтцултэуоэечоа
ечяифмфсосакбщблетипчьаьтобшифцхбялчюфййлфеяйчэусасьйдмчоюэйеьтнфлфцфчйофтцссасиф
ылкцрлфлчлвсофртбибнпалйхзжйлеэыаурсзэшцилмипайеымопсафыццтиксуфйшиллцйноццфхомб
обячюэубмильыбошньхйллцрксифрлвлсщзежцялильоусрлгещфйяхепьтюзежцлуялямчпрлцлыцял
шйвтцллевььбйуйшцфаауспяолпэпрбиксаегвпаусубшйьтошньдмэбрлрвринйысрлчюшцхоаимжпф
шйащщфниасчлчйжйэаюэчокбофлйхзжйебгбгоаежймоьаялщбифжаубчбхйвьзэбисазпфюжцчьсаьв
чомйбчиесачсптялгьбщвлифшйояпапршйвтцллебноцфюэсзэзыцлюуйльэдглцнччбхнялжхвбрижэчб
ллтньаоцкффулеьаусзымуусуэиивгмуьаьаюейнсдязешыумеиелцчяйщдтсфашвидмгбвиччмуюююажф
бсфдюцноцдпфжчйжйлзьсжффйлжчхялеихоинюеоиццвбюйшйляфюмивцвбйтчулйяцхожцаелеасуэ
яфллкотипчыэымаечойлфезамкаьсажлафчуещзешщксьлгйсэйщжйсюзащмибхссачсптжлпефцщмвь
бтрлцизаялхифюцлдюцццфютошшйьтбыццошьйилшмчуомэбалилоююеьаялилгйжиоцгонтнцдфщб
кечоксюэяфнцюжкюмиасюэююцлзшдюзэщавцвююййэейгйофрлбфебошмфгфмюзэымебмфшизыян
нзнтжллздйфаеэусююфймиййщбчаюэшавцчсубиложхоюйгугфазлевльафюллшйэбсфаюййшйщлый
викюфййтхйюйсфчьдмэбцщщфэапыююзаьтльаозачлоюаеюэелютошхаажллбияожумйбтгбцщзэдгь
йымдтлзьбрцщидпаещгрлбфебзтжллзгфчбмюыйвиелтаеэеыжцацфуяэеылэюеччщбкеаеебшэдуфуеу
цлобфпейжлгблбофошулхашчянялазултайьюелэуэщмымдтчуошбияофютамжасасыумйбтлцлфлйаэ
чоллвлосзйлежцьййфысоцобгбфечопурзвэщаьттайьеэоцчллитснлщбазэблцссебйэетаегмвьобьючй
юнхепйгбилхнкниелэфжкюлщьахутаоццльйдсщфкбошьййшктлцулщлнфтцйхклююфйцщдмьйещш
цялвсхечойлфеяйвбюэлщвьклмфоюфйхашчфжщбяфялцльййлйеьтяллбялгесачслщщфйтюфьбюещ
маечоялхйьйбэпчллюэвьпаопнайййийавтюебюйьбсфнцьййбтщвьлекюьаллвлйлжечовфвфдэщаулпоз
авьчуйэнчзэмуулйлйшщйыимжгбцалщчунцллжйгщопнчзафлилффсучуйюклщлмфйшоффпсфесшц
фюфйспсфесаечомимкзанйбуилясрбхутаоцьйаювььайэщмымэбтопчюеаехсбнйеуувихевюаькфсжза
ццуэасхерюяйтцссасетялуицжщбыюсащбчлтцвгкбрлципыйеьтыимжчбпфыьоцэигбхуднююлщвлфл
дчзаялилцирюетмулемфллжлпфцлуичьуэкюццфывбцфжазэдгсумйбтнлнэымсаюечоццошйэнчзэобв
бллвсэбюпсафыыэемшйьйззийешклошмиццофгбтеебрйглдсвлььдмхзялхйилхйешулгоаежйошфьгу
жлтюжйттхутаоцазялйшллбифжщфгййшцлтззсчутэкьносайэнчзэобобфпщэюеасцлфйшноцщбьйжл
днзашцнеелуичоцлтюаечлялципыйеьтьтйэымюэмптфюэсфешгбдоьиьаьтусююучуфечофлялжлажаоь
аьтвевьечйщриццвбнцопыихеэтжллзулыйьэщаьтпулекюьаьтщбцихечьдмэбвжоцхзнцльеэастиялмс
йрчуобжеиекьрифбошьтялафцщбццфйюэфкцоюыэнзвссфмсзэщаьтщбьйжллщвлгфчутэмжхоюдюэ
фщксхеьавцшщаеебыймбебееeташйеяжйьгугьгуйбьйчэюеофбнховидмчоьйхулбошюювидмобхйтц
ыюфйклвлхлчбкеоцхзмсбщаеоцфюобьйцщдмчуэбщбнййбщысдчлтээюаеюэмжйрюйлечуэбэребмаь
аоцфыыиксфюксгуюфьйфйялйлэрулзуледгдйюйофмикюрютацпяаццасщаасяллбдмвььахутаоцущы
мцпночлэебцвбщлжлмтзлвцсаюэвьдмэбрлчрьбцфгпебвбшийщлллевцчуюйжлолофгбмйоайесачсшц
рийяассааеьавцпьчьгызаолмбрлаювцялбэасюэчяхутаоцтсебщсдгбиолдсшзщлмфнмэбпювидмлщзел
экнщмфюаеюэфюфйауоюфйобпйленебнцлымвлбэагницнксвцулсфкцллжлтамжасаетиагялхйялйшл
лветиоцшинаьтемдтмфоюажаоюйофзэуэщмфюущтигоаежййюццеоыиолэщюэшачльняльйоьуэцлби
рыщлдэхоефгйчйсшшцвбьйтцацофафччыэусымчбщмюэйщксзэюецчююдгулхулбщлэщзазяейжлви
пчзаыицжфюнтщбаююебцмихойепалэдгшифюцдялаэксщлмсзэтюаьчоымнвэбйббинчшйьйпфчбпэ
ымелциюеыэцлжлющриозянвгхйкенвэблсчоиейщришщщфьтйбошщбыйьэшцошвьцлкитсдгюэлцзй
ийлевцгфьбфечоуэшцфюфйщждпнаюэхооллетипчцлулмиымзааююехктйьтзауоцбйшпзэафюцлклгй
нцбтошчйюнхемуулялощвьбтсфрщ

**ВТ:**

атызнаешьсколькоразмывэтомгодуиграливбейсболавпрошломавпозапрошломнистогонисс
егоспросилтомгубыегодвигалисьбыстробыстроявсезаписалтысячпятьсотшестьдесятвосемь
разасколькоразячистилзубызадесятьлетжизнишестьтысячразарукимылпятнадцатьтысячраз
спалчетыреслишнимтысячиразиэтотольконочьюиселшестьсотперсиковивосемьсотяблокаг
рушвсегодвестиянеоченьтолюблюгрушичтохочешьспросиуменявсезаписаноесливспомнит
ьисосчитатьчтояделалзавседесятьлетпрямотысячимиллионовполучаютсявотвтдумалдугл
асопятьоноближепочемупотомучтотомболтаетноразведеловтомеонвсетрещititрещитспол
нымртомотецсидитмолчанасторожилсякакрысьатомвсеболтаетникакнеугомонитсяшипити
пенитсякаксифонссодовойкнигяпрочелчетыресташтуккиносмотрелитогобольшесорокфиль
мовсучастиембакаджонсатридцатьсджекомхоксисорокпятьстомоммиксомтридцатьдевятьс
хутомгибсономстодевяностодвамультипликационныхпрокотафеликсадесятьсдугласомфер
бенксомвосемьразвиделпризраквопереслономчаничетыреразасмотрелмилтонасиллсадаже
одинпролюбовьсадольфомменжутолькоятогдапросиделцелыхдевяносточасоввкиношнойу
борнойвсеждалчтобэтаерундакончиласьипустиликошкуиканарейкуилилетучуюмышьаужт
утвсецеплялисьдругзадружкуивизжалидвачасабезпередышкииселзаэтовремячетыресталед
енцовтристатянучексемьсотстаканчиковмороженоготомболталещедолгоминутпятьпокаоте
цнепрервалегоаскольконягодтысегоднясобралтомровнодвестипятьдесятшестьнеморгнувгла
зомответилтомотецрассмеялсяинаэтомокончилсязавтраконивновьдвинулисьвлесныетенис
обиратьдикийвиноградикрошечныеягодыземляникивсетроенаклонялиськсамойземлерукиб
ыстроиловкоделалисвоеделоведравсетяжелелиадугласприслушивалсяидумалвотвотоноопя
тьблизкопрямоуменязаспинойнеоглядывайсяработайсобирайягодыкидайвведрооглянешьс
яспугнешьнетужнаэтотразнеупущунокакбыегозаманитьпоближечтобыпоглядетьнанегогля
нутьпрямовглазакакауменявспичечномкоробкеестьснежинкасказалтомиулыбнулсяглядяна
своюрукуонабылавсякраснаяотягодкаквперчаткезамолчичутьнезавопилдугласнонеткричат
ьнельзявсполошитсяэхоивсеспугнетпостойкатомболтаетаоноподходитвсеближезначитоно
небоитсятоматомтолькопритягиваетеготомтоженемножкооноделобылоещевфевралевалил
снегаяподставилкоробоктомхихикнулпоймалоднуснежинкупобольшеираззахлопнулскоре
йпобежалдомойисунулвхолодильникблизкосовсемблизкотомтрещалбезумолкуадугласнесв
одилснегоглазможетотскочитьудратьведьиззалесанакатываетсякакаятогрознаяволнавотсей
часобрушитсяираздавитдасэрзадумчивопродолжалтомобрываякустдикоговинограданавесь
штатиллинойсуменяуодноголетоместьснежинкатакойкладбольшенигденесыщешьхотьтрес
низавтраяееоткроюдугтытожеможешьпосмотретьвдругоевремядугласбытолькопрезритель
нофыркнулнудамолснежинкакакбынетакносейчаснанегомчалосьтоогромноевотвотобруши
тсясногонебаионлишьзажмурилсяикивнултомдотогоизумилсячтодажепересталсобирать
ягодыповернулсяиуставилсянабратадугласзастылсидянакорточкахнукактутудержатьсятом
испустилвоинственныйкличкинулсянанегоопрокинулназемлюонипокатилисьпотравебарах
таясьитузядругдруганетнетниочемдругомнедуматьивдругкажетсявсехорошодаэтастычкап
отасовканеспугнуланабегавшуюволнувотоназахлестнулаихразлиласьшироковокругинесет
обоихпогустойзеленитравывглубьлесакулактомаугодилдугласупогубамвортусталогорячои
солонодугласобхватилбратакрепкостиснулегоионизамерлитолькосердцаколотилисьдадыш
алиобасосвистомнаконецдугласукрадкойприоткрылодинглазвдругопятьничеговотоновсет
утвсекакестьточноогромныйзрачокисполинскогоглазакоторыйтожетолькочтораскрылсяиг
лядитвизумлениинанеговупорсмотр

**ВИСНОВКИ:**

Ми здобули навички частотного аналізу на прикладі розкриття моноалфавітної підстановки, а також опанували засоби роботи в модулярній арифметиці. Знайшли всіх кандидатів на ключ, розшифрували ними текст, відкинули неправильно розшифровані тексти.