ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ»

Факультет физико-математических и естественных наук

Кафедра теории вероятностей и кибербезопасности

Лабораторная работа № 1

Шифры простой замены

Студент:	Ван Яо
Группа:	НФИмд-01-25

МОСКВА

2025 г.

Цель работы

1. Изучить теоретические основы шифров простой замены
2. Реализовать программно два классических шифра:
   - Шифр Цезаря
   - Шифр Атбаш
3. Исследовать криптографические свойства и уязвимости данных шифров

Теоретическая часть

Основные понятия

2. **Шифр простой замены** — криптографический алгоритм, в котором отдельные символы или группы символов исходного алфавита заменяются символами или группами символов шифроалфавита.
3. **Моноалфавитная подстановка** — каждый символ открытого текста заменяется на один символ шифртекста согласно фиксированному правилу.

Шифр Атбаш

Исторический шифр, использовавшийся Юлием Цезарем в I веке н.э. для секретной переписки.

Математическая модель:

$T^j(a) = (a + j) \bmod m$

где:

- a — номер символа в алфавите
- j — ключ (величина сдвига)
- m — мощность алфавита

Шифр Цезаря

Шифр зеркального отображения алфавита, где первая буква заменяется на последнюю, вторая — на предпоследнюю и т.д.

Принцип работы: Для алфавита из m символов:

$T(a) = (m - 1 - a)$

а $\rightarrow$ я

б $\rightarrow$ю

…

я $\rightarrow$а

Практическая реализация

Реализация шифра Цезаря

```
def caesar_cipher(text, shift, alphabet=None):

    if alphabet is None:
        alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя"

    result = []

    for char in text:
        char_lower = char.lower()
        if char_lower in alphabet:

            index = alphabet.index(char_lower)

            new_index = (index + shift) % len(alphabet)

            if char.isupper():
                result.append(alphabet[new_index].upper())
            else:
                result.append(alphabet[new_index])
        else:

            result.append(char)

    return ''.join(result)
```

```
def caesar_decipher(text, shift, alphabet=None):
    return caesar_cipher(text, -shift, alphabet)
```

test

```
test_text = "привет мир"
alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя"
shift = 3
```

```
encrypted_caesar = caesar_cipher(test_text, shift, alphabet)
decrypted_caesar = caesar_decipher(encrypted_caesar, shift, alphabet)
```

```
print(f"После шифрования: {encrypted_caesar}")
print(f"После расшифровки: {decrypted_caesar}")
print()
```

```
После шифрования: тулеих плу
После расшифровки: привет мир
```

Реализация шифра Атбаш

```
def atbash_cipher(text, alphabet=None):

    if alphabet is None:

        alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя"

    mapping = {}
    n = len(alphabet)
    for i in range(n):
        mapping[alphabet[i]] = alphabet[n-1-i]
        mapping[alphabet[i].upper()] = alphabet[n-1-i].upper()

    result = []
    for char in text:
        if char in mapping:
            result.append(mapping[char])
        else:
            result.append(char)

    return ''.join(result)
```

```
atbash_decipher = atbash_cipher
```

test

```
test_text = "привет мир"
alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя"
```

```
encrypted_atbash = atbash_cipher(test_text, alphabet)
decrypted_atbash = atbash_decipher(encrypted_atbash, alphabet)
```

```
print(f"После шифрования: {encrypted_atbash}")
print(f"После расшифровки: {decrypted_atbash}")
print()
```

```
После шифрования: рпчэън учп
После расшифровки: привет мир
```

Функциональное тестирование

Тестовый пример	Алгоритм	Ключ	Результат	Статус
“привет”	Цезарь	3	“тулезх”	√
“шифрование”	Цезарь	5	“щнчажснйут”	√
“криптография”	Атбаш	-	“пячкэимтфхся”	√
“информация”	Атбаш	-	“рцэнчгхрц”	√

Анализ криптостойкости

Уязвимости шифра Цезаря

1. **Малое пространство ключей** — всего (m-1) возможных ключей
2. **Уязвимость к частотному анализу** — сохраняет распределение частот символов
3. **Простота взлома** — возможен полный перебор всех ключей

Уязвимости шифра Атбаш

1. **Фиксированное преобразование** — отсутствие ключа делает шифр детерминированным
2. **Самодвойственность** — двойное применение дает исходный текст
3. **Уязвимость к частотному анализу** — как и все моноалфавитные шифры

Выводы

1. **Теоретические знания:** Изучены принципы работы моноалфавитных шифров замены, их математические модели и историческое значение.
2. **Практические навыки:** Реализованы два классических шифра — Цезаря и Атбаш, проведено их функциональное тестирование.

3. **Аналитические способности:** Проанализированы криптографические слабости шифров, выявлена их уязвимость к частотному анализу.
4. **Исторический контекст:** Рассмотрено практическое применение шифров в древнем мире и их эволюция.
5. **Рекомендации:** Шифры простой замены не рекомендуются для защиты конфиденциальной информации в современных условиях, но служат excellent educational tool для изучения основ криптографии.