Лабораторная работа №1

Шифры простой замены

Доборщук Владимир Владимирович, НФИмд-02-22

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Цель данной работы — изучить и программно реализовать шифры простой замены.

2 Задание

Заданием является:

- Реализовать шифр Цезаря с произвольным ключом k;
- Реализовать шифр Атбаш.

3 Теоретическое введение

Шифр простой замены представляет собой замену каждой буквы в исходном слове на определенное число, которому соответствует данная буква [1]. В основе функционирования шифров простой замены лежит следующий принцип: для получения шифртекста отдельные символы или группы символов исходного алфавита заменяются символами или группами символов шифроалфавита.

3.1 Шифр Цезаря

Шифр Цезаря является моноалфавитной подстановкой, т.е. каждой букве открытого текста ставится в соответствие одна буква шифротекста.

Математическая процедура шифрования описывается как

$$T_m = \{T^j\}, j = 0, 1, \cdots, m-1,$$

$$T^j(a) = (a + j) \mod m,$$

где m - длина алфавитаа, j - произвольный ключ (величина сдвига от изначальной позиции буквы), a - текущая позиция буквы в алфавите.

Для латинского алфавита длина составляет 26 символов, а формулу можно привести к виду:

$$T^k(i) = (i + k) \mod 26,$$

где i, k соответствуют a, j, а $m = 26$.

Сам же Цезарь обычно использовал подстановку T^3.

3.2 Шифр Атбаш

Шифр Атбаш является сдвигом на всю длину алфавита. Правило шифрования состоит в замене i-й буквы алфавита буквой с номером $n - i + 1$, где n — число букв в алфавите.

4 Выполнение лабораторной работы

Для реализации шифров мы будем использовать Python, так как его синтаксис позволяет быстро реализовать необходимые нам алгоритмы.

4.1 Реализация шифра Цезаря с произвольным ключом k

Шифр Цезаря реализуем в виде функции `ceasar` следующего вида:

```
# --- Ceasar's Cipher ---
def ceasar(letter: chr, key: int, alphabet: list):
    def ceasar(letter: chr, key: int):
        return alphabet.index(letter) + key

    if letter.lower() not in alphabet:
        return letter

    t_letter = alphabet[ceasar(letter.lower(), key) % len(alphabet)]

    if letter.isupper():
        t_letter = t_letter.upper()

    return t_letter
```

На вход она принимает переменные `letter` (один символ), `key` (произвольный ключ), `alphabet` (алфавит в виде списка).

В ходе обработке мы работаем с индексами элементов массива-строки, предварительно проверяя, является ли символ частью передаваемого алфавита. Если да, то мы вызываем вложенную функцию для расчета сдвига и выполняем к ней операцию деления с остатком (исходя из формулы в теоретическом введении).

В конце мы проверяем, является ли буква заглавной, и, после ситуативной обработки, возвращаем зашифрованную букву.

4.2 Реализация шифра Атбаша

Шифр Атбаш реализуем в виде функции `atbash` следующего вида:

```
# --- Atbash's Cipher ---
def atbash(letter: chr, alphabet: list):
    if letter.lower() not in alphabet:
        return letter

    t_letter = alphabet[len(alphabet) - alphabet.index(letter.lower()) - 1]

    if letter.isupper():
        t_letter = t_letter.upper()

    return t_letter
```

На вход она принимает те же переменные, что и функция Шифра Цезаря, исключая произвольный ключ.

Шифруется символ засчет вычитания из длины алфавита индекс символа, над которым производится шифрование.

Возвращается также зашифрованный символ.

4.3 Тестирование

Для тестирования мы создали следующие функции:

```
# --- Tests ---
def test_ceasar(message: str, key: int, alphabet: list):
    ciphered_message = list(map(
      lambda letter: ceasar(letter, key, alphabet), message)
    )
    return "".join(ciphered_message)

def test_atbash(message: str, alphabet: list):
    ciphered_message = list(map(
      lambda letter: atbash(letter, alphabet), message)
    )
    return "".join(ciphered_message)
```

Данные тесты возвращают строку шифро-текста.

Для их вызова, реализуем функцию `main` следующим образом:

```
# --- Main function ---
def main():
    latin_alphabet = list(map(
      chr, range(97, 123)
    )) # Latin alphabet list
    cyrillic_alphabet = list(map(
      chr, range(1072, 1104)
    )) + list(chr(32)) # Cyrillic alphabet list

    latin_message = "Veni, vidi, vici"
    latin_message_new = "Happy New Year, my darling friend!"
```

```
cyrillic_message = "".join(cyrillic_alphabet)

print("\nCEASAR'S CIPHER TEST 1\n-----------")
print(f"Original: {latin_message}\n\
  Ciphered: {test_ceasar(latin_message, 3, latin_alphabet)}\
  \n-----------\n")

print("CEASAR'S CIPHER TEST 2\n-----------")
print(f"Original: {latin_message_new}\n\
  Ciphered: {test_ceasar(latin_message_new, 3, latin_alphabet)}\
  \n-----------\n")

print("ATBASH'S CIPHER TEST STRING OUTPUT\n-----------")
print(f"Original: {cyrillic_message}\n]\
  Ciphered: {test_atbash(cyrillic_message, cyrillic_alphabet)}\
  \n-----------\n")

print("ATBASH'S CIPHER TEST LIST OUTPUT\n-----------")
print(f"Original: {list(cyrillic_message)}\n\
  Ciphered: {list(test_atbash(cyrillic_message, cyrillic_alphabet))}\
  \n-----------\n")
```

4.4 Результаты тестирования

Запустив наш программный код, получим результат, изображенный в приложении 6.1.

Для шифра Цезаря с ключом $k = 3$ получаем следующий результат:

```
CEASAR'S CIPHER TEST 1
-----------
```

```
Original: Veni, vidi, vici
Ciphered: Yhql, ylgl, ylfl
-----------
```

Сравнивая результат шифрования с примером из описания лабораторной работы, можем убедиться, что наша реализация корректна.

Дополнительно проверим механизм шифрования, передав другую строку из букв латинского алфавита:

```
CEASAR'S CIPHER TEST 2
-----------
Original: Happy New Year, my darling friend!
Ciphered: Kdssb Qhz Bhdu, pb gduolqj iulhqg!
-----------
```

Видим, что шифрование прошло успешно.

Шифр Атбаш мы проверяем на кириллическом алфавите, содержащим также в себе символ пробела. Для проверки, передадим в него также весь русский алфавит с пробелом в виде одной строки:

```
ATBASH'S CIPHER TEST STRING OUTPUT
-----------
Original: абвгдежзийклмнопрстуфхцчшщъыьэюя
Ciphered:  яюэьыъщшчцхфутсрпонмлкйизжедгвба
-----------
```

Видим, что наша строка “отзеркалилась”, а значит - алгоритм шифрования работает корректно и сдвиг произошел на всю длину алфавита. Чтобы в этом убедиться, выведем результат в формате спсика, где сможем рассмотреть каждый обработанный символ отдельно:

```
ATBASH'S CIPHER TEST LIST OUTPUT
```

```
-----------
Original: ['а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н',
           'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы',
           'ь', 'э', 'ю', 'я', ' ']
Ciphered: [' ', 'я', 'ю', 'э', 'ь', 'ы', 'ъ', 'щ', 'ш', 'ч', 'ц', 'х', 'ф', 'у',
           'т', 'с', 'р', 'п', 'о', 'н', 'м', 'л', 'к', 'й', 'и', 'з', 'ж', 'е',
           'д', 'г', 'в', 'б', 'а']
-----------
```

Видим, что каждый из символов был корректно заменен.

5 Выводы

В рамках выполненной лабораторной работы мы изучили и реализовали следующие шифры простой замены: шифр Цезаря (с произвольным ключом k) и шифр Атбаш.

6 Приложения

```
(base) wdoborschuk@wxcore ~/work/2022-2023/МОЗИиИБ/infosec/laboratory/lab01 (develop)$ python task.py

CEASAR'S CIPHER TEST 1
──────────
Original: Veni, vidi, vici
Ciphered: Yhql, ylgl, ylfl
──────────

CEASAR'S CIPHER TEST 2
──────────
Original: Happy New Year, my darling friend!
Ciphered: Kdssb Qhz Bhdu, pb gduolqj iulhqg!
──────────

ATBASH'S CIPHER TEST STRING OUTPUT
──────────
Original: абвгдежзийклмнопрстуфхцчшщъыьэюя
Ciphered:  яюэьыъщшчцхфутсрпонмлкйизжедгвба
──────────

ATBASH'S CIPHER TEST LIST OUTPUT
──────────
Original: ['а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', ' ']
Ciphered: [' ', 'я', 'ю', 'э', 'ь', 'ы', 'ъ', 'щ', 'ш', 'ч', 'ц', 'х', 'ф', 'у', 'т', 'с', 'р', 'п', 'о', 'н', 'м', 'л', 'к', 'й', 'и', 'з', 'ж', 'е', 'д', 'г', 'в', 'б', 'а']
──────────
```

Рис. 6.1: Вывод программы с реализованными шифрами простой замены

Список литературы

1. Золотин Ф., Сорокин М. Криптографические алгоритмы // ББК 74.480 Н 52. С. 51.