Министерство науки и высшего образования РФ

Федеральное государственное автономное образовательное учреждение

высшего образования

«Московский политехнический университет»

(Московский политех)

Отчёт по курсу «Программирование криптографических алгоритмов»

Лабораторная работа 1. Шифрование шифрами однозначной замены



Выполнил:

Студент группы 221-352

Иванов В. В.

Проверил преподаватель: Бутакова Н. Г.

Москва 2024г.

**Аннотация**

* **Среда программирования**
  + Visual Studio Code
* **Язык программирования**
  + Python
* **Процедуры для запуска программы** 
  + Visual Studio Code (main.py)
* **Пословица-тест**
  + Тот, кто ложится на два стула, падает на ребра.
* **Текст для проверки работы (не меньше 1000 знаков (1430))**

Жизнь - это удивительное приключение, полное разнообразных событий и встреч. В каждом моменте мы находим что-то новое и уникальное. Стремление к росту и саморазвитию вдохновляет нас на поиск новых горизонтов. Важно помнить, что каждый шаг вперед приносит с собой уроки и опыт.

Разнообразие культур, языков и традиций делает наш мир удивительно богатым. Общение с людьми разных национальностей расширяет кругозор, позволяя нам понимать и уважать друг друга. Взаимное уважение и терпимость создают основу для гармоничного сосуществования.

Природа тоже играет важную роль в нашей жизни. Красота закатов, шум океана, пение птиц - все это напоминает нам о величии мира природы. Забота о окружающей среде становится неотъемлемой частью ответственного образа жизни.

Работа и творчество придают смысл нашим усилиям. Стремление к достижению целей мотивирует нас на новые начинания. Каждый проект, даже самый маленький, приносит удовлетворение и чувство выполненного долга.

Семья и друзья являются надежной опорой в нашей жизни. Обмен историями, веселые посиделки и поддержка в трудные моменты создают теплую атмосферу взаимопонимания и любви.

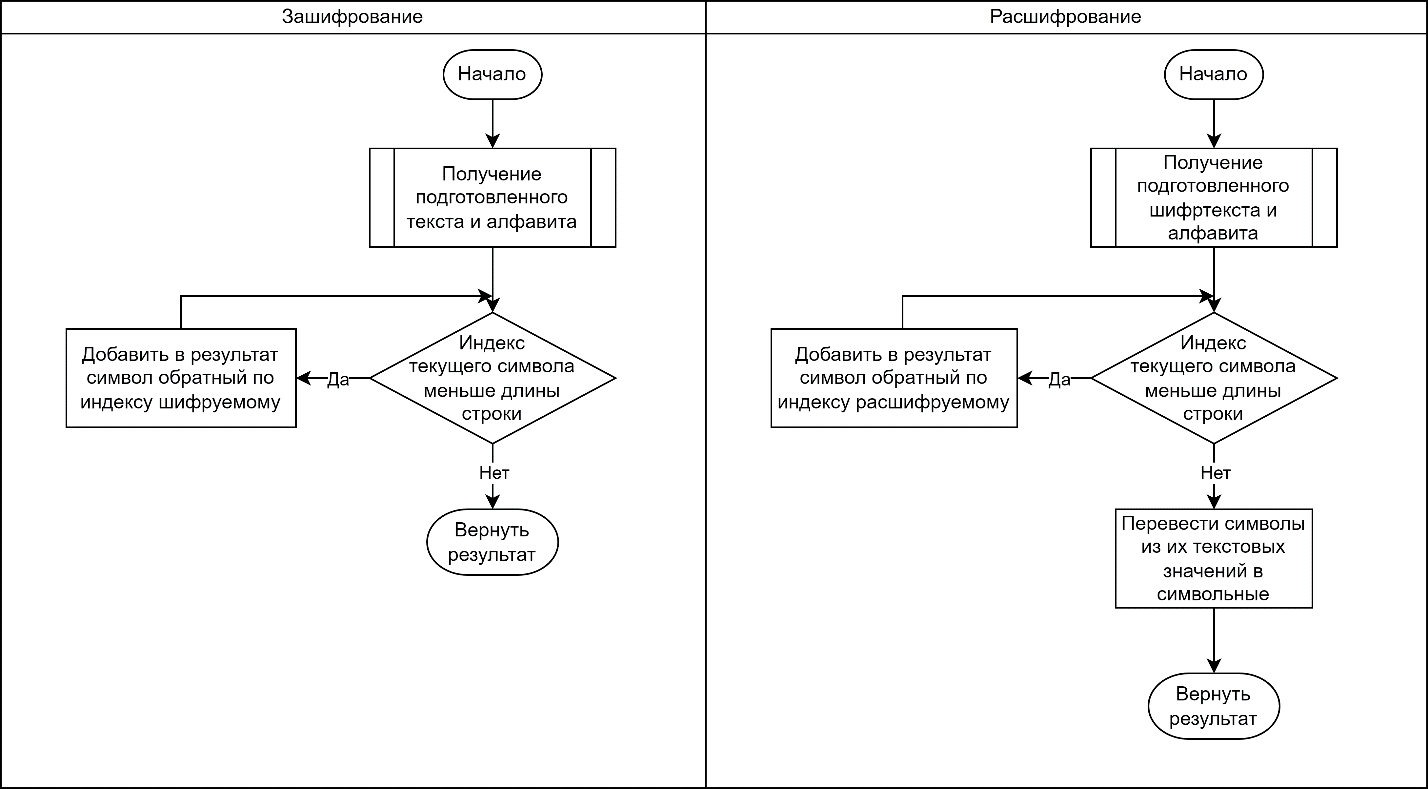
Таким образом, наша жизнь - это мозаика различных моментов, соединенных воедино. Важно ценить каждый момент и стремиться делать мир вокруг нас ярче и лучше. С любовью, терпением и целеустремленностью мы можем создавать свою уникальную историю, наполненную смыслом и радостью.

* **Код программы-интерфейса**
* import sys
* from PyQt5.QtWidgets import QApplication, QMainWindow, QLabel, QTextEdit, QPushButton, QVBoxLayout, QWidget, QComboBox, QLineEdit # элементы интерфейса
* from PyQt5.QtCore import Qt # импортируем константы и классы
* class CipherApp(QMainWindow):
* def \_\_init\_\_(self):
* super().\_\_init\_\_()
* self.current\_mode = 0
* self.encryption\_dict = {} # пустой словарь для хранения информации о шифрах и ключах
* self.initUI()
* def initUI(self): # пользовательский интерфейс
* input\_label = QLabel('Введите текст:')
* self.input\_text = QTextEdit()
* cipher\_label = QLabel('Выберите шифр:')
* self.cipher\_combobox = QComboBox()
* self.cipher\_combobox.addItems(["Шифр Атбаш", "Шифр Цезаря", "Квадрат Полибия"])
* key\_label = QLabel('Введите ключ (только для Шифра Цезаря):')
* self.key\_entry = QLineEdit()
* encrypt\_button = QPushButton('Зашифровать', self)
* encrypt\_button.clicked.connect(self.on\_encrypt\_button\_click)
* decrypt\_button = QPushButton('Расшифровать', self)
* decrypt\_button.clicked.connect(self.on\_decrypt\_button\_click)
* toggle\_button = QPushButton('Переключить режим', self)
* toggle\_button.clicked.connect(self.toggle\_mode)
* self.mode\_label = QLabel('Режим: Обычный')
* output\_label = QLabel('Результат:')
* self.output\_text = QTextEdit()
* # Создание центрального виджета и установка компоновки
* central\_widget = QWidget()
* layout = QVBoxLayout()
* layout.addWidget(input\_label)
* layout.addWidget(self.input\_text)
* layout.addWidget(cipher\_label)
* layout.addWidget(self.cipher\_combobox)
* layout.addWidget(key\_label)
* layout.addWidget(self.key\_entry)
* layout.addWidget(encrypt\_button)
* layout.addWidget(decrypt\_button)
* layout.addWidget(toggle\_button)
* layout.addWidget(self.mode\_label)
* layout.addWidget(output\_label)
* layout.addWidget(self.output\_text)
* central\_widget.setLayout(layout)
* # Установка центрального виджета в главное окно
* self.setCentralWidget(central\_widget)
* self.setWindowTitle('Шифры')
* self.setGeometry(100, 100, 640, 420)
* def toggle\_mode(self):
* self.current\_mode = 1 - self.current\_mode
* self.mode\_label.setText("Режим: " + ("Обычный" if self.current\_mode == 0 else "Расширенный"))
* if self.cipher\_combobox.currentText() == "Расширенный":
* self.output\_text.setPlainText("Выберите шифр")
* else:
* # Update the output based on the current mode
* text\_to\_process = self.input\_text.toPlainText()
* if self.current\_mode == 1:
* result = self.extended\_cipher(text\_to\_process, action="encrypt")
* else:
* result = self.extended\_cipher(text\_to\_process, action="decrypt")
* self.output\_text.setPlainText(result)
* def on\_encrypt\_button\_click(self):
* text\_to\_encrypt = self.input\_text.toPlainText()
* selected\_cipher = self.cipher\_combobox.currentText()
* if selected\_cipher == "Шифр Цезаря":
* key = int(self.key\_entry.text())
* result = self.caesar\_cipher(text\_to\_encrypt, key)
* elif selected\_cipher == "Шифр Атбаш":
* result = self.atbash\_cipher(text\_to\_encrypt)
* elif selected\_cipher == "Квадрат Полибия":
* result = self.polybius\_square\_encrypt(text\_to\_encrypt)
* elif selected\_cipher == "Расширенный":
* result = self.extended\_cipher(text\_to\_encrypt)
* else:
* result = "Выберите шифр"
* self.output\_text.setPlainText(result)
* def on\_decrypt\_button\_click(self):
* text\_to\_decrypt = self.input\_text.toPlainText()
* selected\_cipher = self.cipher\_combobox.currentText()
* if selected\_cipher == "Шифр Цезаря":
* key = int(self.key\_entry.text())
* result = self.caesar\_decipher(text\_to\_decrypt, key)
* elif selected\_cipher == "Шифр Атбаш":
* result = self.atbash\_decipher(text\_to\_decrypt)
* elif selected\_cipher == "Квадрат Полибия":
* result = self.polybius\_square\_decrypt(text\_to\_decrypt)
* elif selected\_cipher == "Расширенный":
* result = self.extended\_cipher(text\_to\_decrypt)
* else:
* result = "Выберите шифр"
* self.output\_text.setPlainText(result)
* if \_\_name\_\_ == '\_\_main\_\_':
* app = QApplication([])
* cipher\_app = CipherApp()
* cipher\_app.show()
* app.exec\_()

**1. Шифр простой замены АТБАШ**

Шифр Атбаш — это простой шифр подстановки для алфавитного письма, в котором каждая n-я буква алфавита заменяется буквой m − n +1, где m — общее число букв в алфавите. Другими словами, первая буква заменяется на последнюю, вторая — на предпоследнюю и так далее.

**Блок-схема программы**

****

**Код программы с комментариями**

def atbash\_cipher(self, input\_str):

        alphabet = "АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"

        punctuations = {

            ",": "ЗПТ",

            ".": "ТЧК",

            "-": "ТИРЕ",

        }

        preprocessed\_input = input\_str.upper()

        for char, replacement in punctuations.items():

            preprocessed\_input = preprocessed\_input.replace(char, replacement)

        result = ""

        for char in preprocessed\_input:

            if char.upper() in alphabet:

                index = alphabet.index(char.upper())

                new\_char = alphabet[len(alphabet) - index - 1]

                if char.isupper():

                    result += new\_char

                else:

                    result += new\_char.lower()

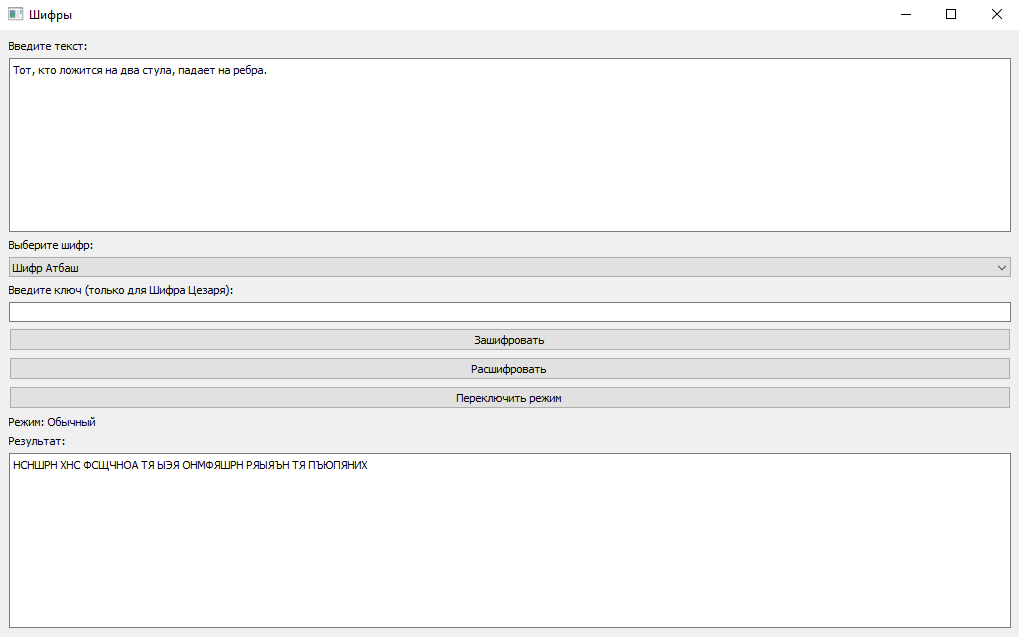
            else:

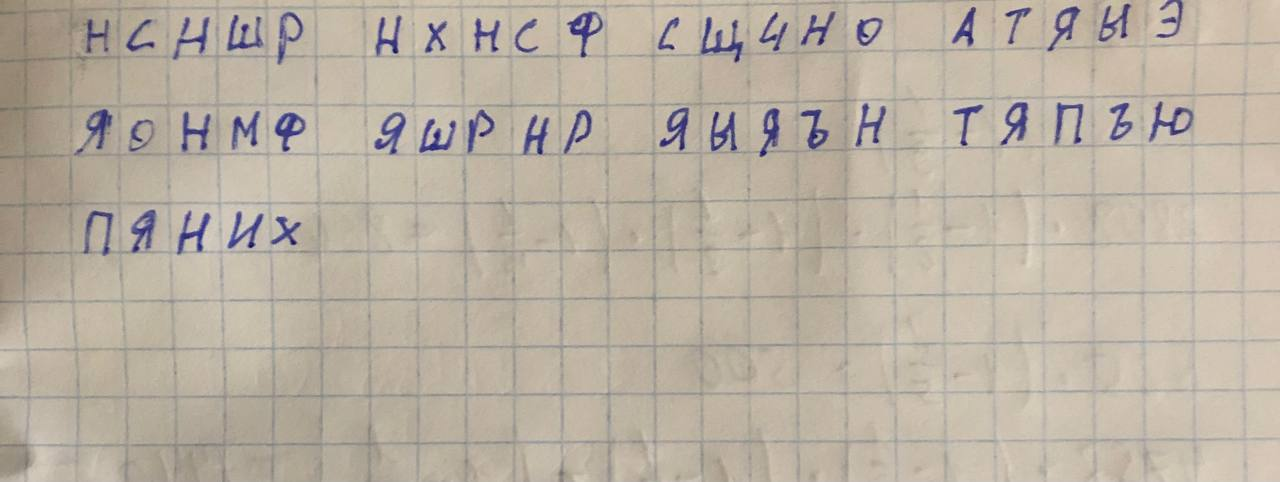
                result += char

        return result

}

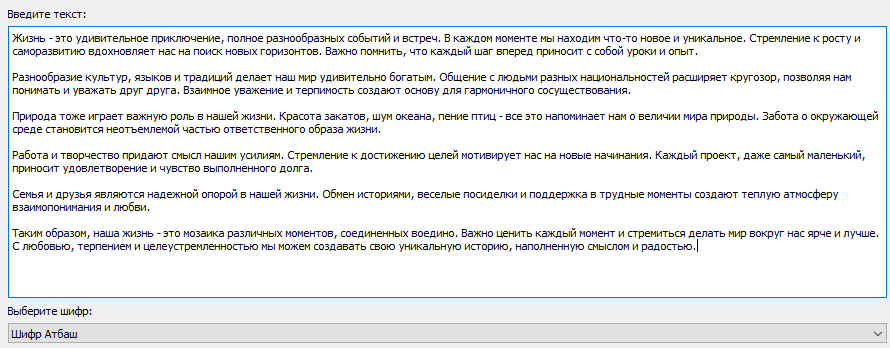
**Тестирование**

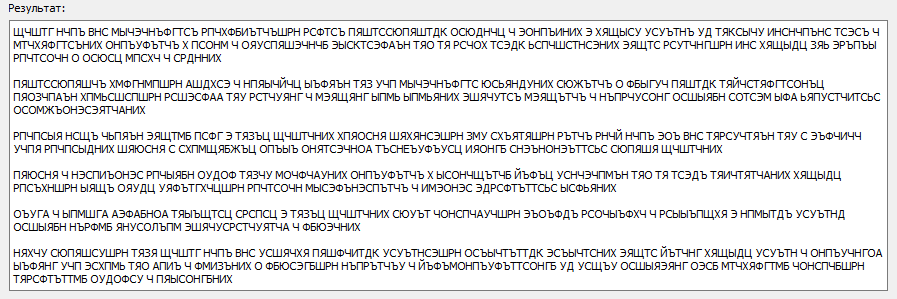




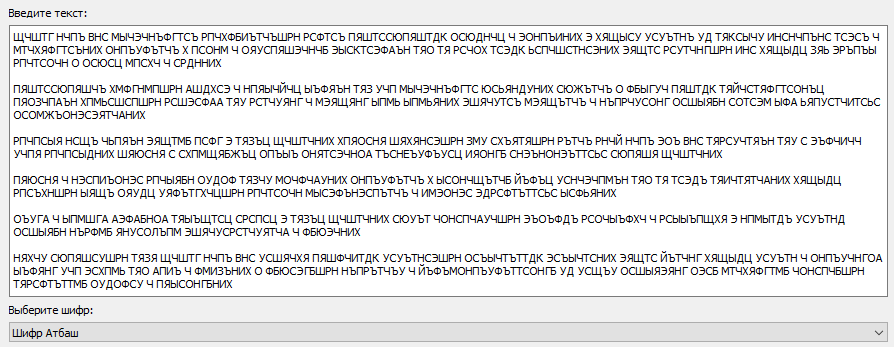
**Работа с текстом не менее 1000 знаков**

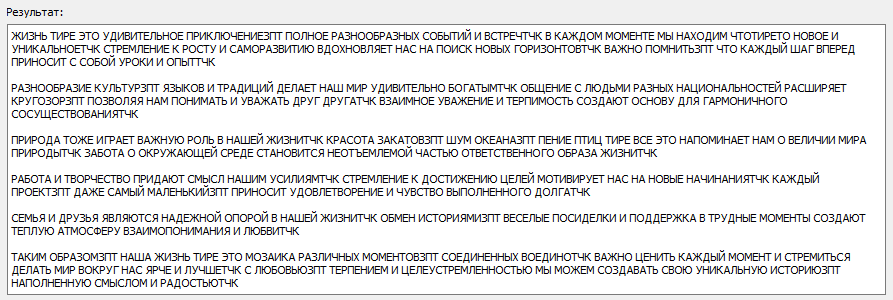
**Зашифрование**





**Расшифрование**

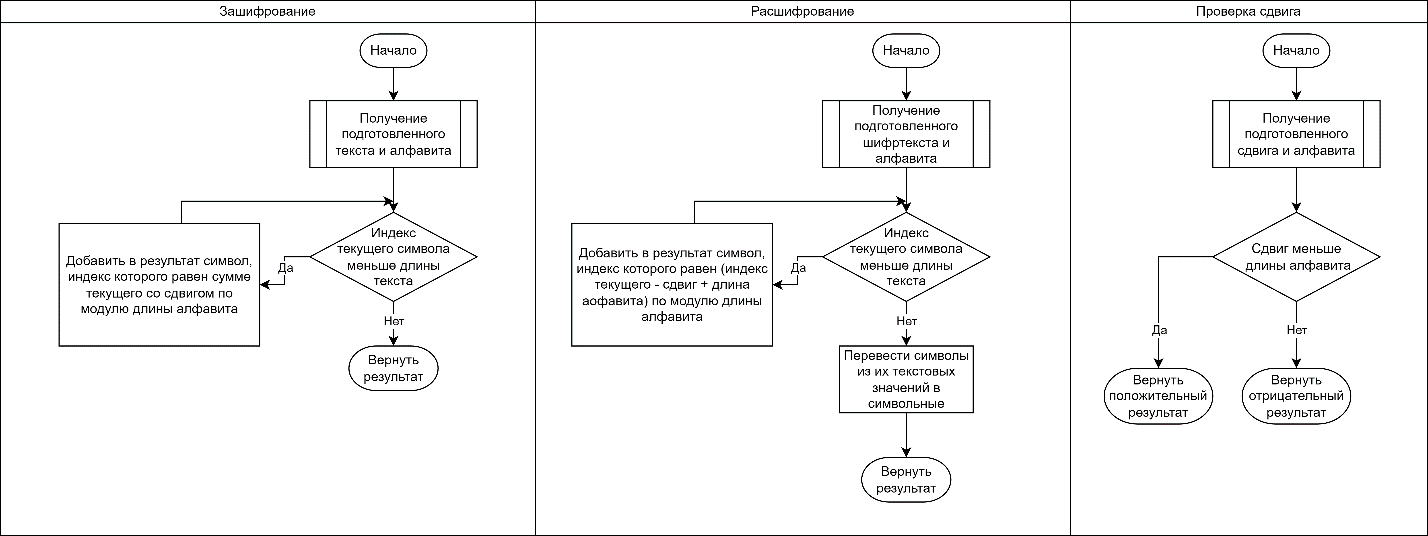


**2. ШИФР ЦЕЗАРЯ**

Шифр Цезаря — это шифр, при использовании которого каждая буква из открытого текста заменяется на такую букву, которая в алфавите находится на некотором постоянном числе позиций левее или правее от рассматриваемой буквы.

Например, при сдвиге букв русского алфавита вправо на 3 позиции буква «А» заменяется на букву «Г», буква «Б» заменяется на букву «Д» и так далее

**Блок-схема программы**

****

**Код программы с комментариями**

def caesar\_cipher(self, input\_str, key):

        alphabet = "АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"

        punctuations = {

            ",": "зпт",

            ".": "тчк",

            "-": "тире",

        }

        # Заменяем знаки препинания

        for original\_char, replacement in punctuations.items():

            input\_str = input\_str.replace(original\_char, replacement)

        result = ""

        for char in input\_str:

            if char.isalpha(): # проверка является ли символ буквой

                is\_upper = char.isupper() # является ли буква заглавной

                char = char.upper() # привод к верхнему регистру

                index = (alphabet.index(char) + key) % len(alphabet) # новый индекс символа

                new\_char = alphabet[index]

                if not is\_upper: # восстанавливаем регистр

                    new\_char = new\_char.lower()

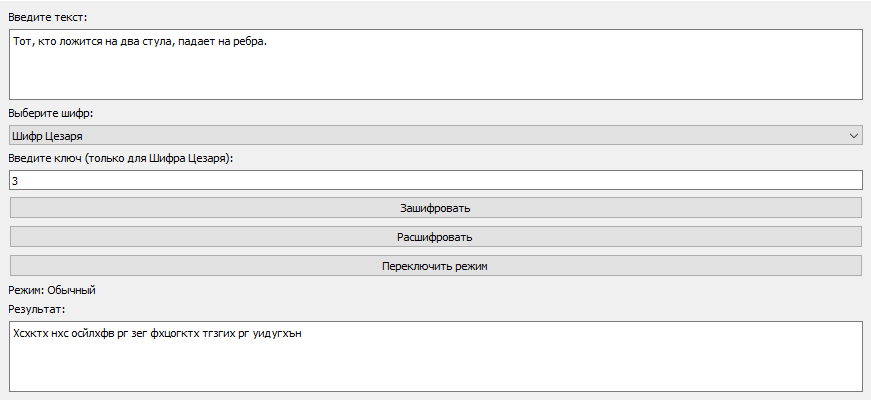
                result += new\_char

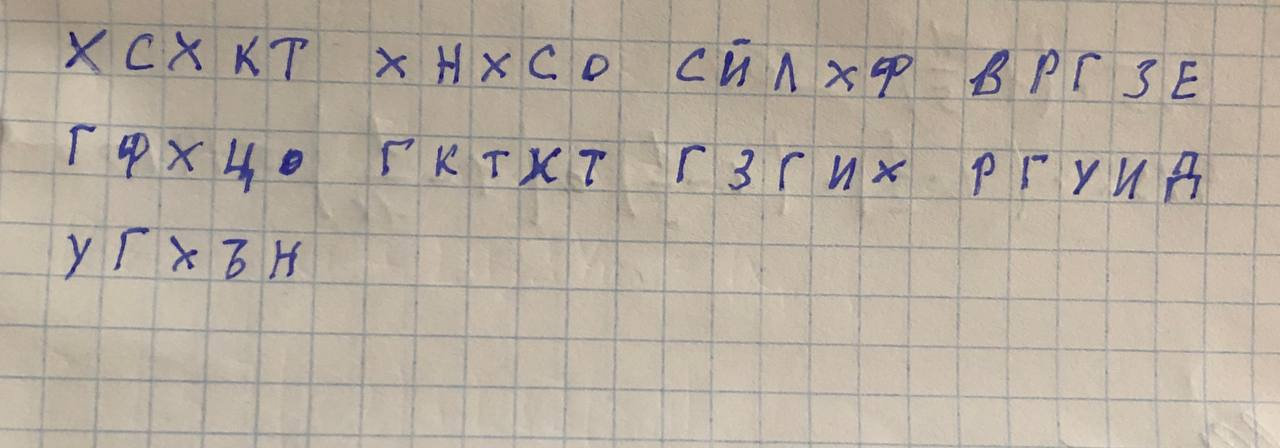
            else:

                result += char # символ не является буквой, оставляем его

        return result

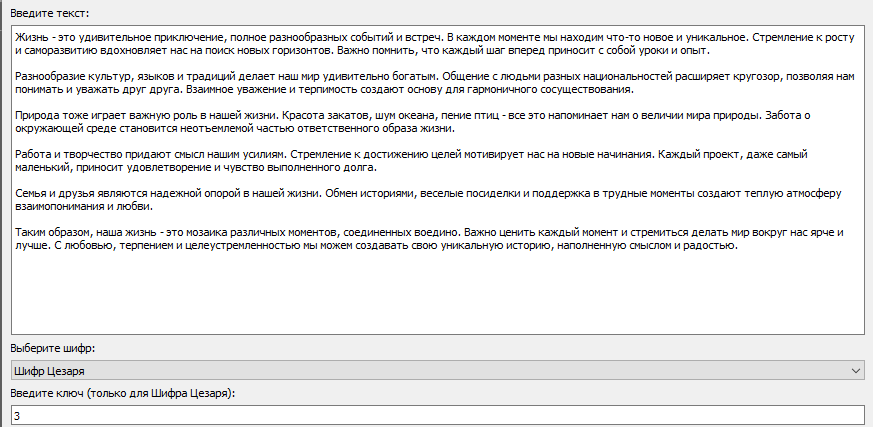
**Тестирование**

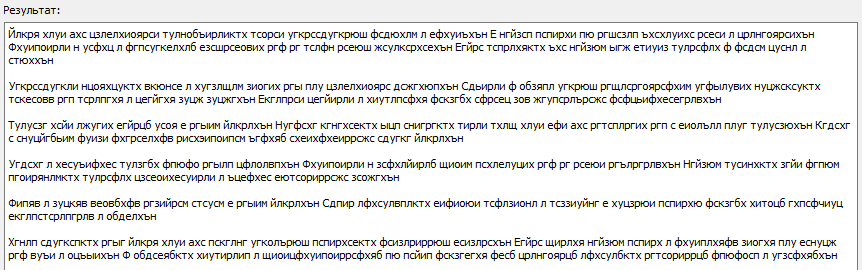




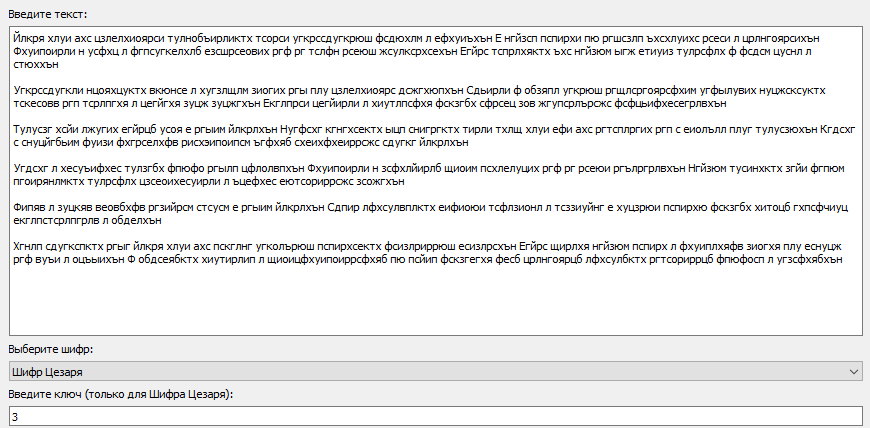
**Работа с текстом не менее 1000 знаков**

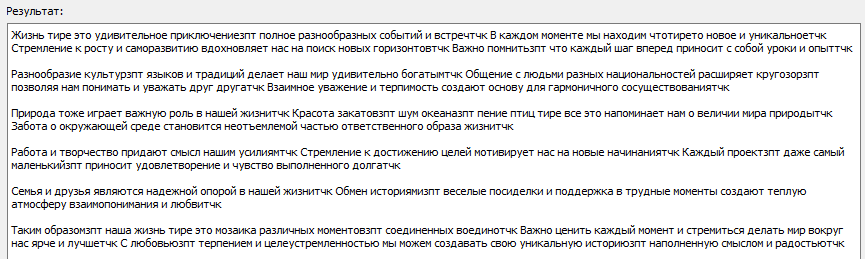
**Зашифрование**





**Расшифрование**



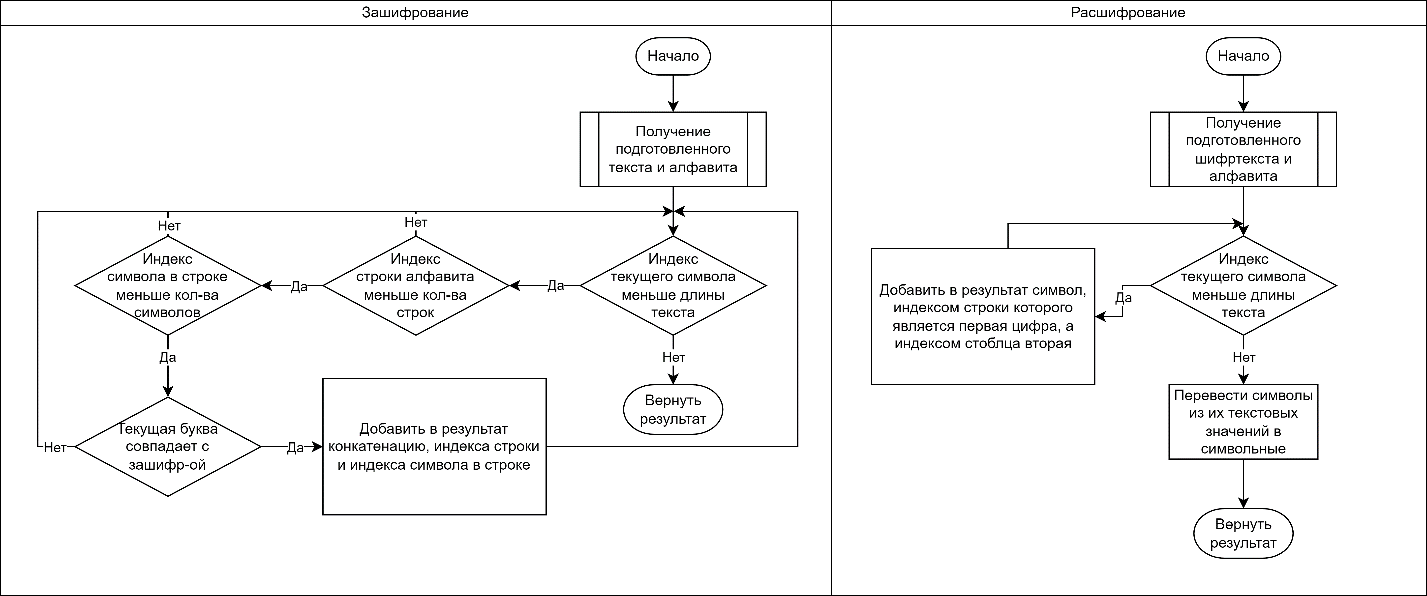


**3. Квадрат Полибия**

Квадрат Полибия — это способ шифрования, разработанный древнегреческим историком и полководцем Полибием.

Он представляет собой квадрат 6х6, столбцы и строки которого нумеруются цифрами от 1 до 6. Все буквы алфавита вписываются в квадрат по одной на клетку. Буквы расположены в алфавитном порядке

**Блок-схема программы**

****

**Код программы с комментариями**

def polybius\_square\_encrypt(self, input\_str):

        square = [

            ['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', ',', '.', '-', ' '],

        ]

        input\_without\_spaces = input\_str.replace(' ', '')

        result = ""

        for char in input\_without\_spaces.upper():

            found = False

            for i in range(len(square)):

                for j in range(len(square[i])):

                    if square[i][j] == char:

                        result += f'{i + 1}{j + 1}'

                        found = True

                        break

                if found:

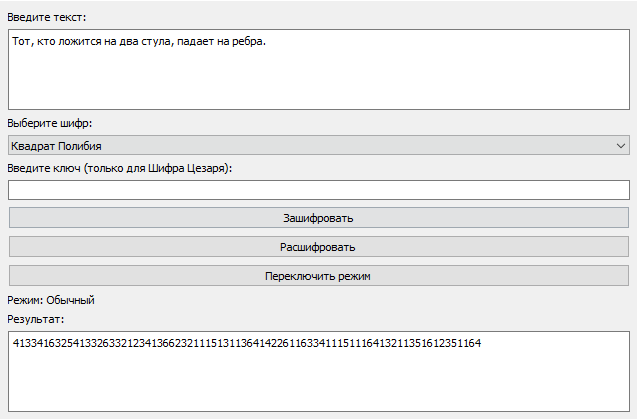
                    break

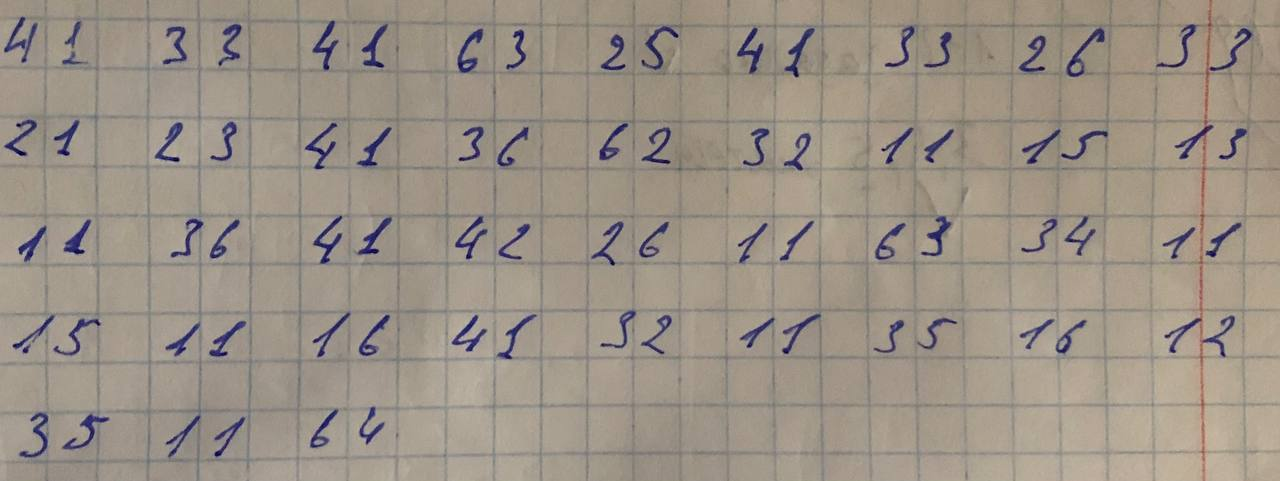
            if not found:

                result += char

        return result

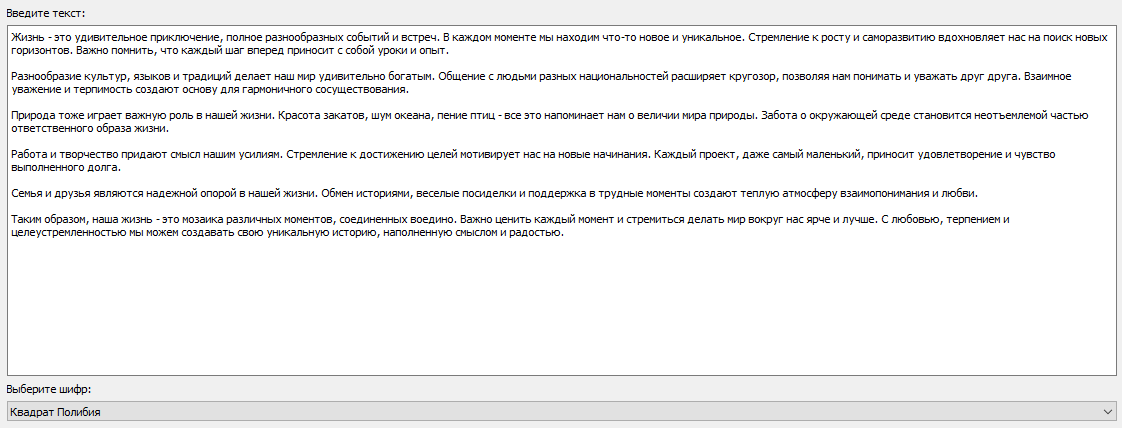
**Тестирование**

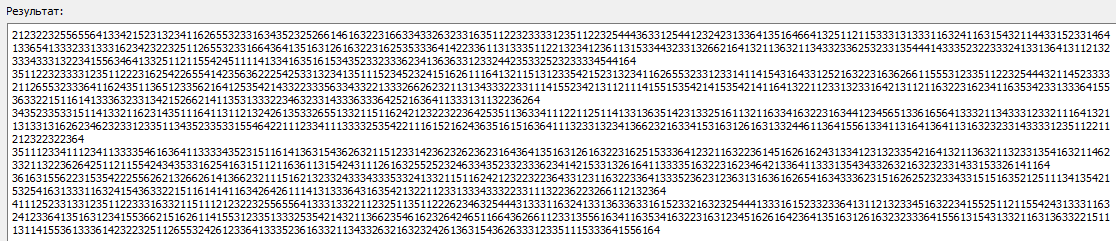




**Работа с текстом не менее 1000 знаков**

**Зашифрование**





**Расшифрование**

