УДК 004

Создание программы для имитации шифрования машины Enigma на языке Python

Жуков Дмитрий Сергеевич
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
В данной статье показана разработка программы шифрования методом машины Enigma.Программа разработана на языке программирования Python.Была использована существующая библиотека Python.
Ключевые слова: python, gui, шифрование, программа.

Creating a program to simulate encryption of an Enigma machine in Python

Zhukov Dmitry Sergeevich
Sholom Aleichem Priamursky State University
Student

Abstract
This article shows how to develop an encryption program using the Enigma machine method. The program was developed in the Python programming language. The existing Python library was used.
Keywords: python, gui, encryption, program.

В современном мире мы ежедневно пользуемся достижениями криптографии: звонок по сотовому телефону, сообщение в мессенджере, загрузка интернет страницы, даже проход через турникет, используя проездную карту студента, не обходятся без применения шифрования. Так же без шифрования не может обойтись ни одна современная война. Во время 2й мировой войны Германия использовала машину Enigma для шифрования своих сообщений.

Р.М. Алоян, В.В. Шутенко, О.Н. Никитина, Л.С. Мизгирев в статье рассматривают вариант обеспечения криптографической защиты информации в условиях динамично развивающейся цифровой экономики с применением метода многоалфавитной подстановки шифра Виженера для симметричного шифрования информации с закрытым ключом. Целью исследований является создание шифровальщика Виженера на языке программирования Python для практического использования криптографических методов защиты экономической и финансовой информации[4]. В.И. Спиридонов посредством криптографического алгоритма RSA кодировал сообщения, состоящие из чисел. Число может быть длинным, в таком случае необходим соответствующий программный

код[2]. И.К.Крайнов, Е.В.Филина в своей работе создают программу шифрования до указанного времени на языке python[1]. В статье А.О.Кизянов, В.А.Глаголев проводят реализацию алгоритма шифрования Рона Ривеста RC5. Для реализации был использован язык программирования Python. Итогом статьи выступает алгоритм RC5, который полностью реализован на языке Python и продемонстрирован на примере [3]. В статье И.Е.Бронштейн рассматриваются виды дефектов, которые обычно встречаются в программном коде на языке Python. Показывается, что возможные дефекты для Python не похожи на те, что часто встречаются в коде на Си/Си++ и, следовательно, необходимо исследование дефектов в крупных проектах с открытым исходным кодом. Даётся классификация найденных дефектов на основе того, нужен ли для нахождения ошибки вывод типов. Показывается, что существует небольшая доля "простых" дефектов, но для обнаружения большинства дефектов вывод типов необходим. Рассматривается вопрос, какие конструкции языка Python должны поддерживаться при выводе типов для нахождения реальных дефектов [5]. В работе Д.А.Кузнецов рассматривается структура интерфейса программы «Фармацевтическая экономическая безопасность» для фармацевтических организаций. Описывается функциональное предназначение и возможности пунктов основного меню прикладной программы [6]. Ю.А.Котов, А.В.Шаповалов в своей статье рассмотрели интерфейс программной реализации экспертной системы для восстановления простой замены букв текста. Описаны базовые элементы интерфейса, включающие выбор функциональных методов и базовых операций (замена, сдвиг, перебор). Не менее значимы иностранные исследования в данной сфере [8].

Разработку программы следует начать с загрузки нужных библиотек. Нам понадобятся библиотеки:

PyEnigmaMachine – библиотека для шифрования методом машины Enigma.

Easy gui – библиотека для быстрого создания интерфейса.

Скачать нужные библиотеки можно введя в консоль команду «pip install easygui».

PyEnigmaMachine можно скачать с Github[10]

Подключим наши библиотеки, а также некоторые другие в скрипте (Рис.1).

```
import easygui

from enigmaMachine import enigmaMachine
```

Рис.1.Подключаем библиотеки

Считываем сообщение, которое мы хотим зашифровать или расшифровать в переменную msg (Рис.2)

```
msg = easygui.enterbox(msg='Введите сообщение')
```

Рис.2. Считываем сообщение

Пример окна для ввода сообщения представлен на рисунке 3.

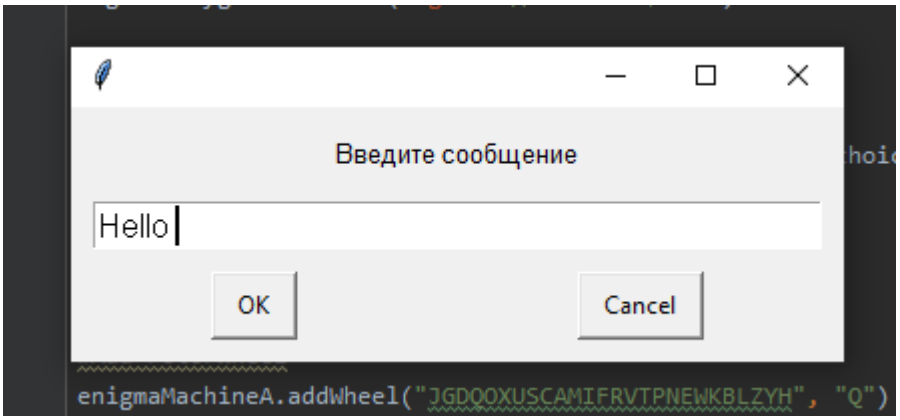

Рис.3. Пример окна для сообщения

Далее создаем диалоговое окно для выбора действия необходимого для выполнения задачи, а именно: зашифровать или расшифровать. Для этого создаем переменную списка с нужными названиями кнопок. И создаем диалоговое окно (Рис 4).

```
choices = ["Зашифровать","Расшифровать"]

reply = easygui.buttonbox(msg="Что сделать?", choices=choices)
```

Рис.4. Создание окна для выбора задачи

Внешний вид окна для выбора нужной задачи представлен на рисунке 5

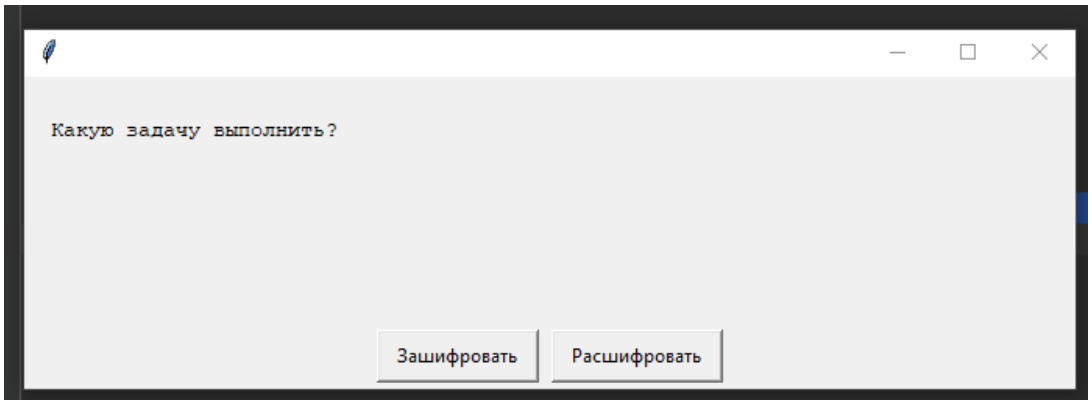

Рис.5. Окно для выбора нужного действия

Далее создается объект класса Enigmamachine который содержит все функции машины шифрования (Рис 6).

```
#Create Machine A
enigmaMachineA = enigmaMachine()
```

Рис.6. Создание объекта класса Enigmamachine

Далее создаем 3 роторных колеса в параметры которых подаем алфавит, который будем шифровать и расшифровывать, буквы которого расположены в случайном порядке. Так вторым параметром подаётся буква, с которой роторное колесо начнет свое движение.

```
#Add rotorWheel
enigmaMachineA.addWheel("JGDQOXUSCAMIFRVTPNEWKBLZYH", "Q")
enigmaMachineA.addWheel("NTZPSFBOKMWRCJDIVLAEYUXHGQ", "E")
enigmaMachineA.addWheel("JVIUBHTCDYAKEQZPOSGXNRMWFL", "V")
```

Рис .7. Создание роторных колес

Далее создаем рефлектор, который нужен для непосредственного шифрования. При создании подаётся алфавит в случайном порядке. От порядка алфавита зависит каким будет зашифрованное сообщение (Рис 8).

```
#Add reflector
enigmaMachineA.addReflector("QYHOGNECVPUZTFDJAXWMKISRBL")
```

Рис.8. Добавление рефлектора

Исходя из кнопки которую выбрал пользователь шифруем или дишефруем сообщение (Рис 9).

```
if reply == 'Зашифровать':
    new_msg = enigmaMachineA.encryptStr(msg)
else:
    new_msg = enigmaMachineA.decryptStr(msg)
```

Рис.9. Выбор зашифровать или расшифровать сообщение

Далее необходимо показать преобразованное сообщение пользователи которое зашифровали либо расшифровали (Рис 10).

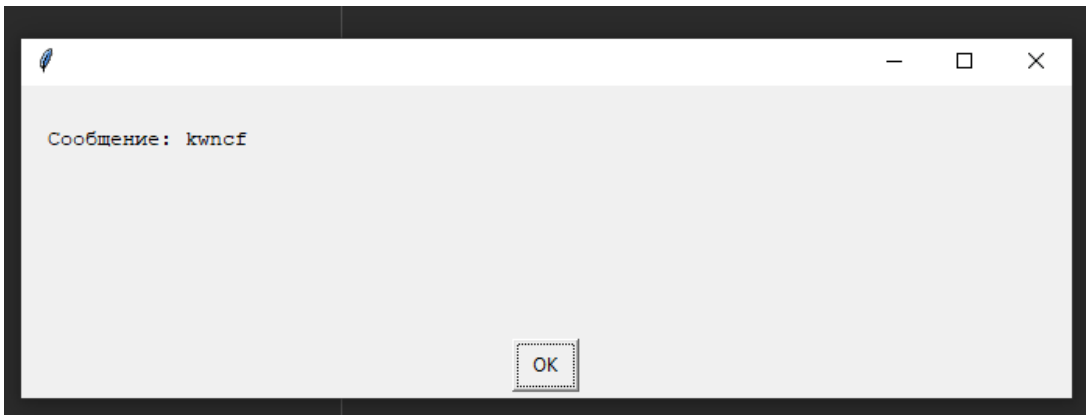

Рис .10. Окно с результатом шифрования

Таким образом, была написана программа, которая шифрует и дешифрует сообщения, эмулируя машину Enigma.

Библиографический список

1. Крайнов И.К., Филина Е.В. Создание программы шифрования до указанного времени на языке python // Юный ученый. 2019. № 8 (28). С. 85-88.
2. Спиридонов В.И. Оптимизация программного кода python для шифрования сообщений из чисел алгоритмом rsa // Вестник современных исследований. 2018. № 12.15 (27). С. 240-242.
3. Кизянов А.О., Глаголев В.А. Реализация шифрования методом rc5 на языке программирования python // Постулат. 2019. № 1-1 (39). С. 64.
4. Алоян Р.М., Шутенко В.В., Никитина О.Н., Мизгирев Л.С. Криптографическая защита, как основа стабильности цифровой экономики // Известия высших учебных заведений. Технология текстильной промышленности. 2017. № 5 (371). С. 232-236.Кузнецов Д.А. Интерфейс программы "фармацевтическая экономическая безопасность" // Российский медико-биологический вестник им. академика И.П. Павлова. 2009. № 2. С. 162-165.
5. Котов Ю.А., Шаповалов А.В. Интерфейс программы восстановления простой замены букв текста // Современные тенденции развития науки и технологий. 2016. № 4-4. С. 57-59.
6. Smith A. W. et al. Application program interface that enables communication for a network software platform : пат. 7117504 США. – 2006.
7. Parikh V., Moore R., Cheng H. Application program interface for a graphics system : пат. 6456290 США. – 2002.
8. PyEnigmaMachine // github URL: https://github.com/mrconter1/PyEnigmaMachine (дата обращения: 14.01.2020).