**Лабораторная работа № 6**

**ИССЛЕДОВАНИЕ ПОТОКОВЫХ ШИФРОВ**

Цель: изучение и приобретение практических навыков разработки и использования приложений для реализации потоковых шифров.

Задачи:

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости потоковых шифров.

2. Разработать приложение для реализации указанных преподавателем методов генерации ключевой информации и ее использования для потокового зашифрования/расшифрования.

3. Выполнить анализ криптостойкости потоковых шифров.

4. Оценить скорость зашифрования/расшифрования реализованных шифров.

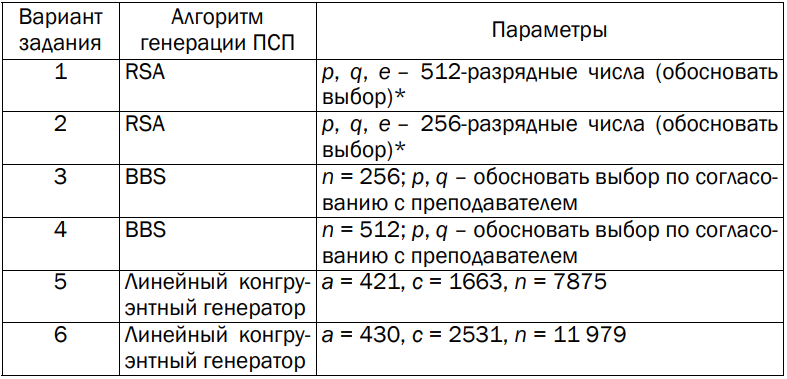
5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

**Практическое задание**

1. Разработать авторские многооконные приложения в соответствии с целью лабораторной работы. При этом можно воспользоваться готовыми библиотеками либо программными кодами, реализующими заданные алгоритмы.

Приложение 1 должно реализовывать генерацию ПСП в соответствии с вариантом из табл. 6.1.

Таблица 6.1 – Варианты для приложения 1



Код реализации приложения 1 представлен в листинге 1.1.

def gcd(a, b):

while b != 0:

a, b = b, a % b

return a

def bbs\_generate(p, q, n, length):

if p % 4 != 3 or q % 4 != 3:

raise ValueError("p и q должны удовлетворять условию p mod 4 = 3 и q mod 4 = 3")

if gcd(p, q) != 1:

raise ValueError("p и q должны быть взаимно простыми числами")

modulus = p \* q

x = random.randint(1, modulus - 1) # начальное значение x0

sequence = []

for \_ in range(length):

x = (x \*\* 2) % modulus

sequence.append(x % 2) # бит (0 или 1)

return sequence

Листинг 1.1 – Код приложения 1

Результат выполнения приложения 1 представлен на рисунке 1.1.

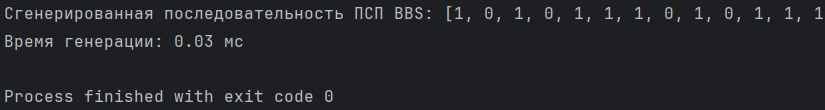


Рисунок 1.1 – Результат выполнения приложения 1

Приложение 2 должно реализовывать алгоритм RC4 в соответствии с вариантом из табл. 6.2, а также дополнительно выполнять оценку скорости выполнения операций генерации ПСП.

Таблица 6.2 – Варианты для приложения 2



Код реализации приложения представлен в листинге 1.2.

def rc4(key, plaintext):

# инициализация S-блока

S = list(range(256))

j = 0

for i in range(256):

j = (j + S[i] + key[i % len(key)]) % 256

S[i], S[j] = S[j], S[i]

# генерация потока ключей

i = j = 0

keystream = []

for \_ in range(len(plaintext)):

i = (i + 1) % 256

j = (j + S[i]) % 256

S[i], S[j] = S[j], S[i]

keystream.append(S[(S[i] + S[j]) % 256])

# шифрование данных

ciphertext = [plaintext[k] ^ keystream[k] for k in range(len(plaintext))]

return ciphertext

Листинг 1.2 – Код приложения 2

Результат выполнения приложения 2 представлен на рисунке 1.2.

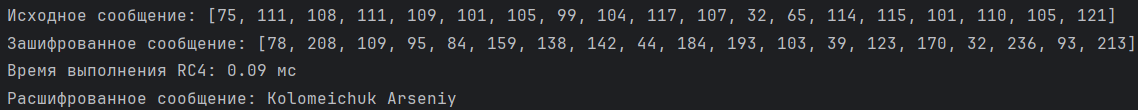


Рисунок 1.2 – Результат выполнения приложения 2

В качестве шифруемого сообщения может быть выбран произвольный текст.

2. Результаты оформить в виде отчета по установленным правилам.