**Шифр RSA**

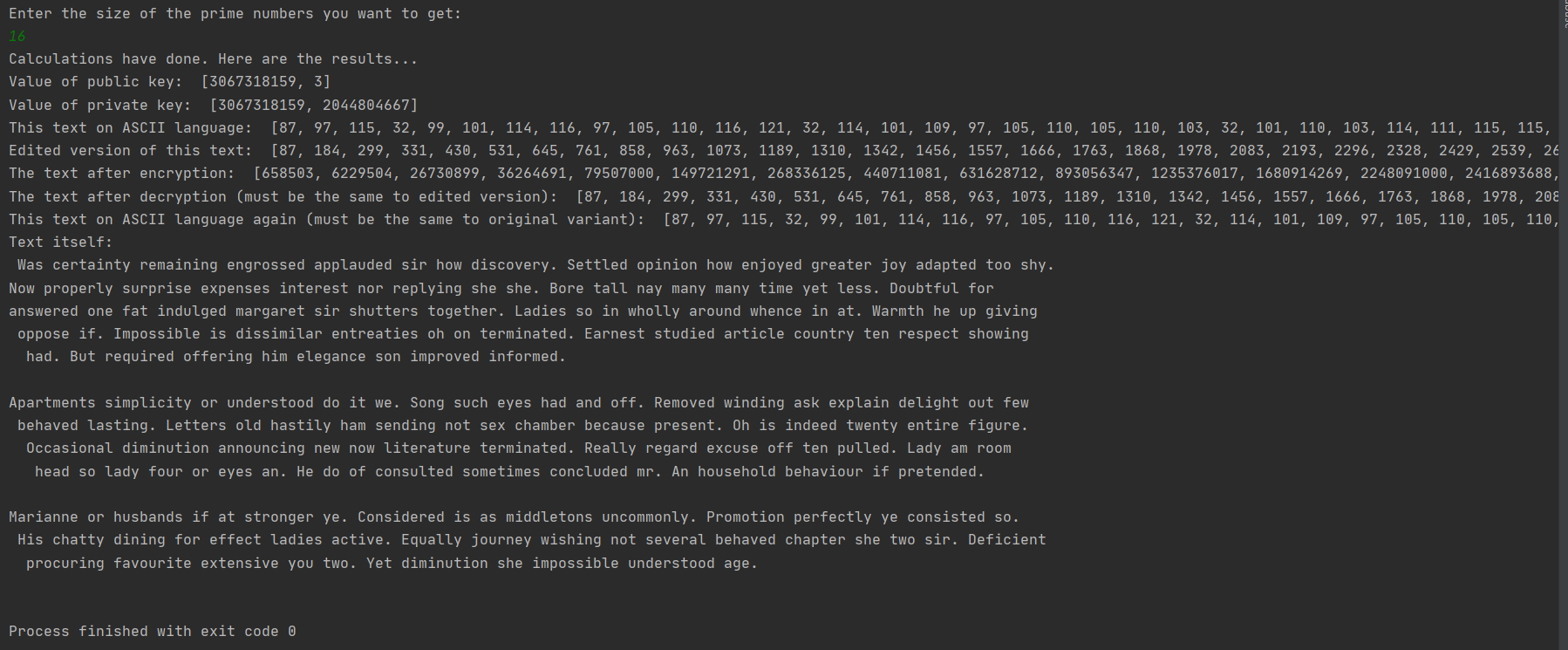
Данная программа является реализацией шифра RSA, которая принимает на вход размер простых чисел для их генерации. После этого сгенерированные простые числа используются для зашифровки алгоритмом RSA.

Исходный код:

from Crypto.Util.number import inverse, getPrime  
from copy import deepcopy  
from math import gcd  
  
  
def BasicCalculation(Data, PrimeNumbersSize):  
 P = getPrime(PrimeNumbersSize) # Какое-то простое число нашего размера  
 Q = getPrime(PrimeNumbersSize) # Какое-то простое число нашего размера  
  
 N = P \* Q  
  
 Fi = (P - 1) \* (Q - 1) # Функция Эйлера  
  
 E = 0 # Число в диапазоне (1, Fi), которое также взаимопростео с Fi  
 for i in range(2, Fi):  
 if gcd(i, Fi) == 1:  
 E = i  
 break  
  
 D = inverse(E, Fi) # Обратное по модулю числа Fi  
  
 Result = [RSA\_EncryptionAndDecryption(Data, N, E, D), [N, E], [N, D]]  
  
 return Result  
  
  
def RSA\_EncryptionAndDecryption(Data, N, E, D):  
 TextInNumbers = [] # Текст, переведённый в юникод ASCII  
 EncryptedText = [] # Зашифрованный текст  
 DecryptedTextNum = [] # Расшифрованный текст, состоящий из чисел по ASCII (по сути символов)  
 DecryptedText = [] # Расшифрованный текст  
 Total = [] # В итоговом массиве хранятся все вышеперечисленные массивы  
  
 for letter in Data:  
 TextInNumbers.append(ord(letter)) # ord - переводит букву в число ASCII  
 Total.append(deepcopy(TextInNumbers)) # deepcopy - полностью новая копия объекта  
  
 for i in range(1, len(TextInNumbers)): # Чтобы преобразовать текст так, чтобы не было повторов символов  
 TextInNumbers[i] = (TextInNumbers[i] + TextInNumbers[i - 1]) % N # остаток на N  
 Total.append(deepcopy(TextInNumbers)) # deepcopy - полностью новая копия объекта  
  
 for i in TextInNumbers:  
 EncryptedText.append(pow(i, E, N)) # Зашифровка  
 Total.append(deepcopy(EncryptedText)) # deepcopy - полностью новая копия объекта  
  
 for i in EncryptedText:  
 DecryptedTextNum.append(pow(i, D, N)) # Расшифровка с манипуляциями для отсутствия повторов  
 Total.append(deepcopy(DecryptedTextNum)) # deepcopy - полностью новая копия объекта  
  
 for i in reversed(range(1, len(DecryptedTextNum))): # Расшировка в числах (ASCII)  
 DecryptedTextNum[i] = pow((DecryptedTextNum[i] - DecryptedTextNum[i - 1]), 1, N)  
 Total.append(deepcopy(DecryptedTextNum)) # deepcopy - полностью новая копия объекта  
  
 for letter in DecryptedTextNum:  
 DecryptedText.append(chr(letter)) # Просто расшировка  
 Total.append(''.join(deepcopy(DecryptedText))) # deepcopy - полностью новая копия объекта  
  
 return Total

from RSA import BasicCalculation  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 Input = open("OpenedText", "r")  
 Data = Input.read()  
  
 PrimeNumbersSize = input("Enter the size of the prime numbers you want to get:\n")  
 result = BasicCalculation(Data, int(PrimeNumbersSize))  
  
 print("Calculations have done. Here are the results...")  
 print("Value of public key: ", result[1])  
 print("Value of private key: ", result[2])  
 print("This text on ASCII language: ", result[0][0])  
 print("Edited version of this text: ", result[0][1])  
 print("The text after encryption: ", result[0][2])  
 print("The text after decryption (must be the same to edited version): ", result[0][3])  
 print("This text on ASCII language again (must be the same to original variant): ", result[0][4])  
 print("Text itself:\n", result[0][5])

Результат работы на размерности 16:



В целом, можно сказать, что шифр RSA достаточно криптоустойчив. Чтобы его эффективно взламывать, требуется решить задачу факторизации, что в наши дни не представляется возможным.