Отчет по лабораторной работе 8

Основы Инфомационной безопасности

Хрусталев Влад Николаевич

Содержание

# 1 Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

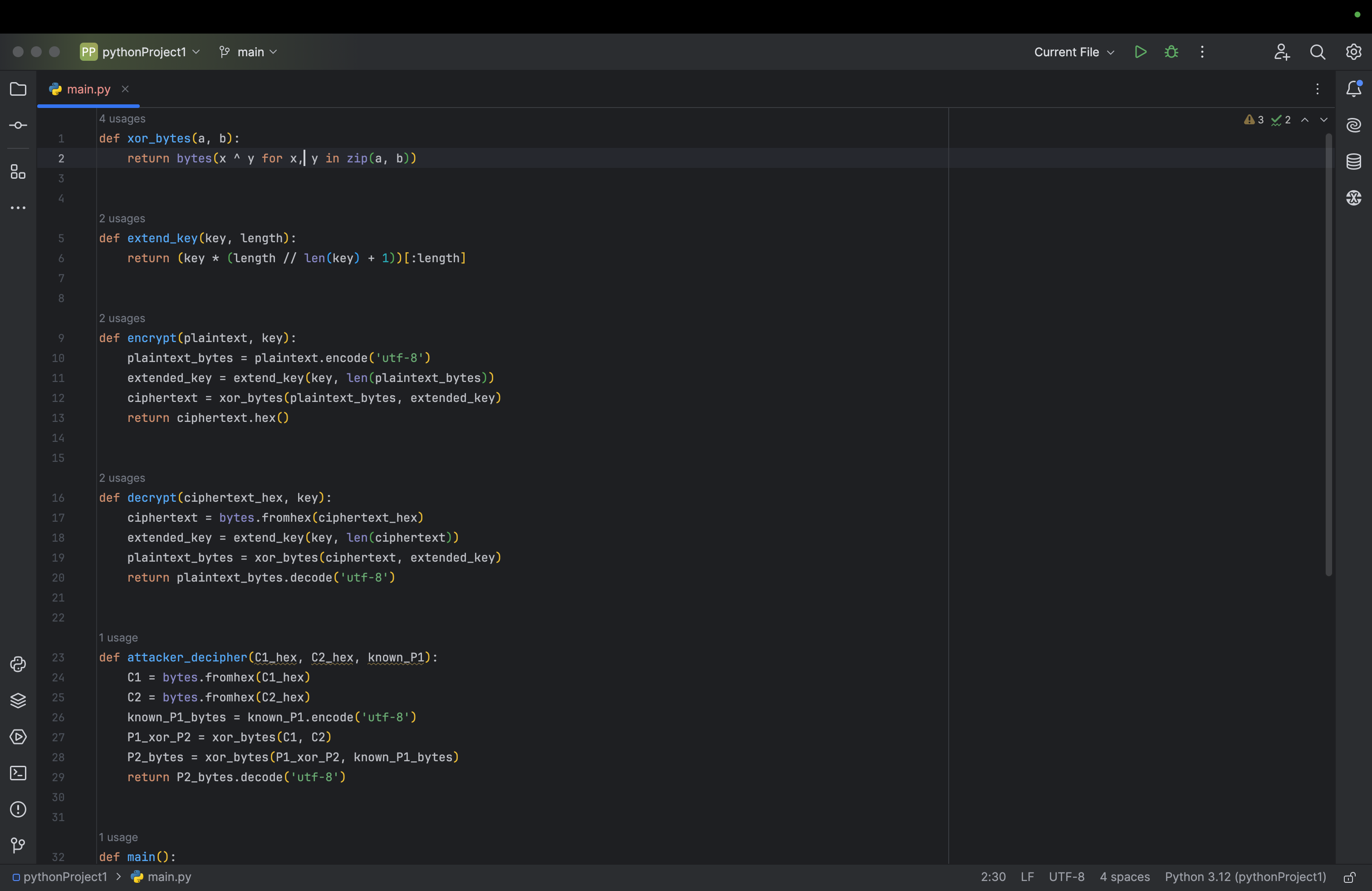
# 2 Ход работы

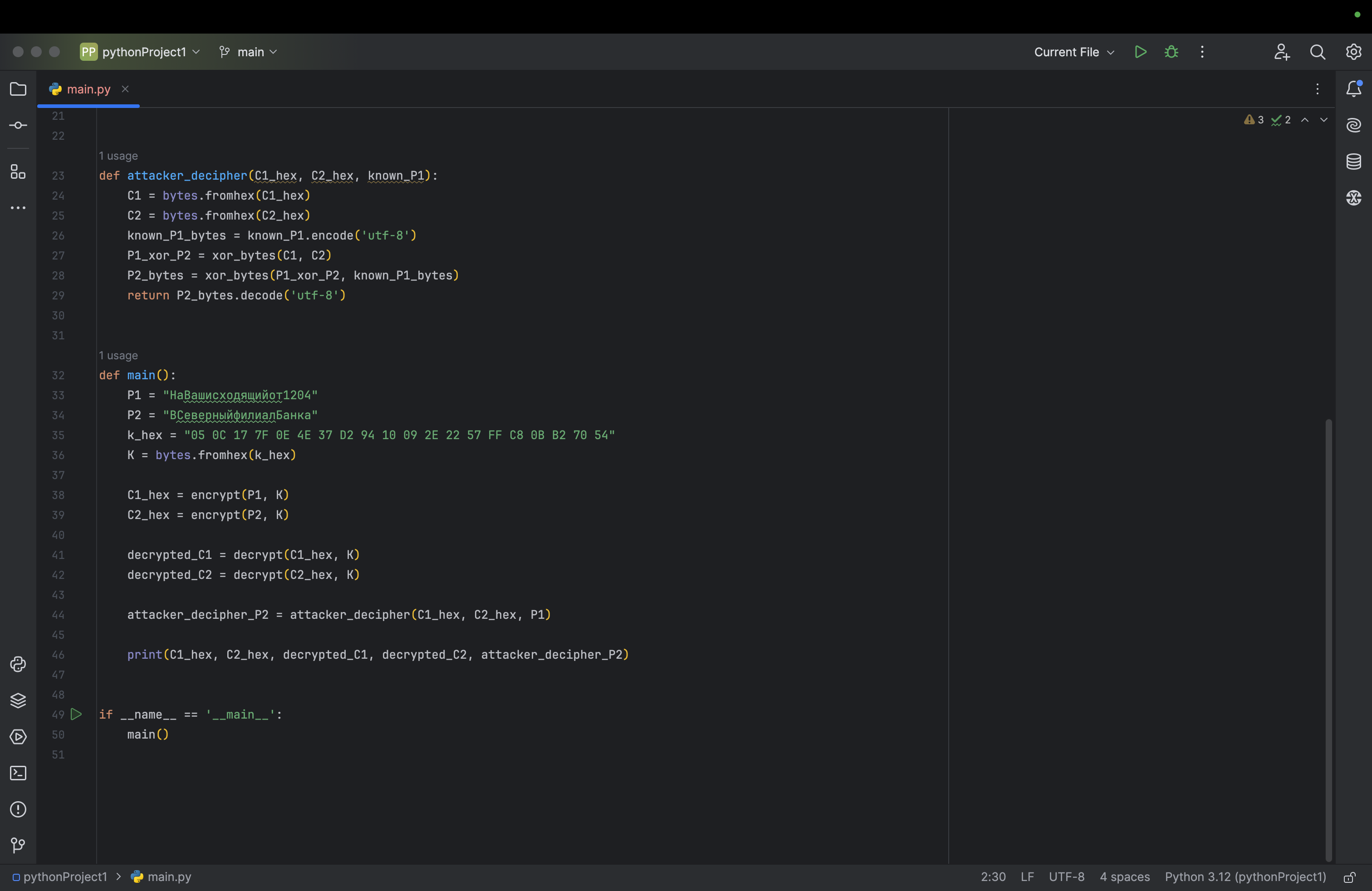
## 2.1 Выполение лабораторной

Нампишем программу эмулирующую задачу

def xor\_bytes(a, b):  
return bytes(x ^ y for x, y in zip(a, b))  
  
  
def extend\_key(key, length):  
return (key \* (length // len(key) + 1))[:length]  
  
  
def encrypt(plaintext, key):  
plaintext\_bytes = plaintext.encode('utf-8')  
extended\_key = extend\_key(key, len(plaintext\_bytes))  
ciphertext = xor\_bytes(plaintext\_bytes, extended\_key)  
return ciphertext.hex()  
  
  
def decrypt(ciphertext\_hex, key):  
ciphertext = bytes.fromhex(ciphertext\_hex)  
extended\_key = extend\_key(key, len(ciphertext))  
plaintext\_bytes = xor\_bytes(ciphertext, extended\_key)  
return plaintext\_bytes.decode('utf-8')  
  
  
def attacker\_decipher(C1\_hex, C2\_hex, known\_P1):  
C1 = bytes.fromhex(C1\_hex)  
C2 = bytes.fromhex(C2\_hex)  
known\_P1\_bytes = known\_P1.encode('utf-8')  
P1\_xor\_P2 = xor\_bytes(C1, C2)  
P2\_bytes = xor\_bytes(P1\_xor\_P2, known\_P1\_bytes)  
return P2\_bytes.decode('utf-8')  
  
  
def main():  
P1 = "НаВашисходящийот1204"  
P2 = "ВСеверныйфилиалБанка"  
k\_hex = "05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54"  
K = bytes.fromhex(k\_hex)  
  
C1\_hex = encrypt(P1, K)  
C2\_hex = encrypt(P2, K)  
  
decrypted\_C1 = decrypt(C1\_hex, K)  
decrypted\_C2 = decrypt(C2\_hex, K)  
  
attacker\_decipher\_P2 = attacker\_decipher(C1\_hex, C2\_hex, P1)  
  
print(C1\_hex, C2\_hex, decrypted\_C1, decrypted\_C2, attacker\_decipher\_P2)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
main()

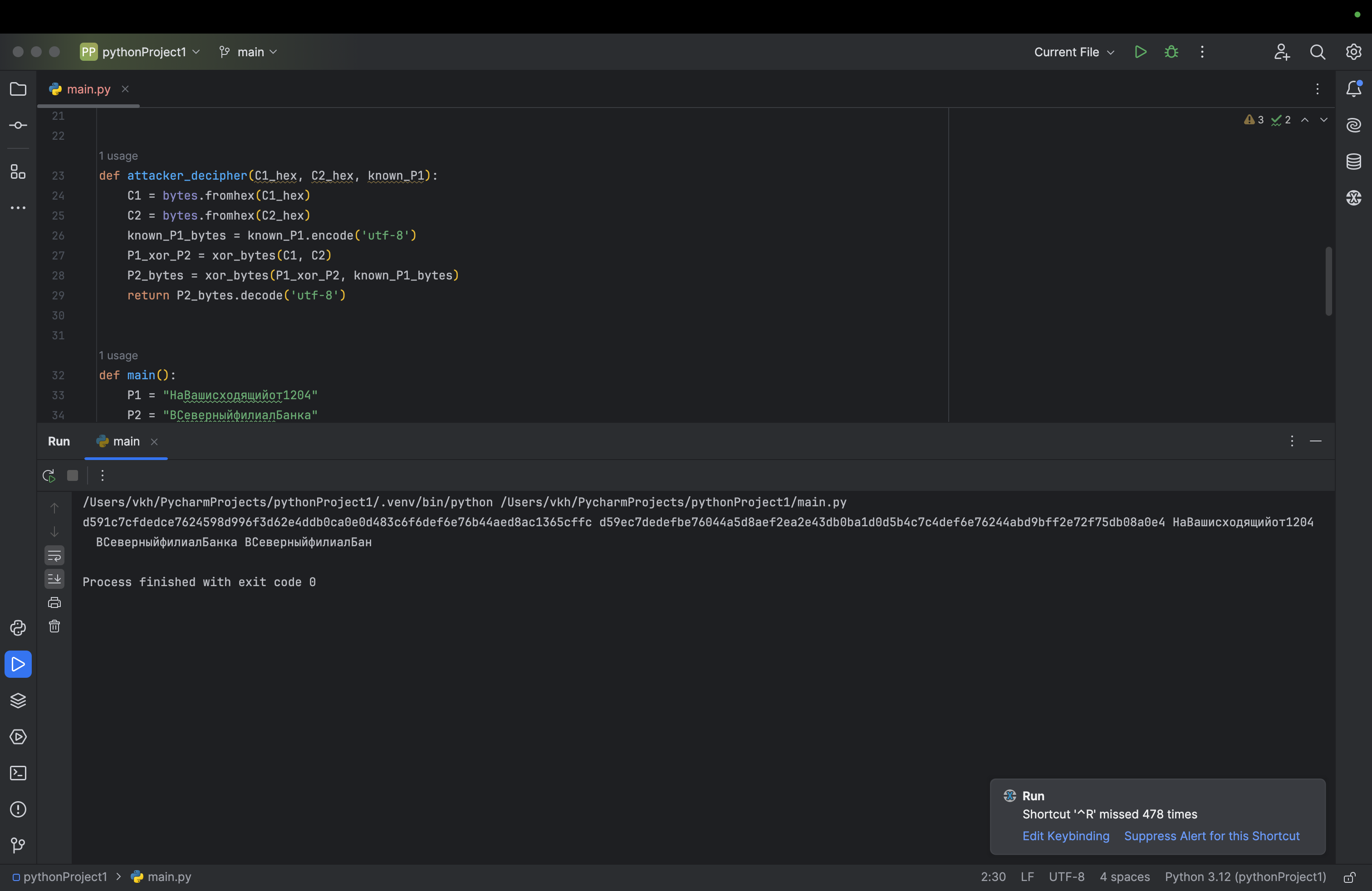
Выше сам код для болеьшей наглядности (рис. **¿fig:001?** , рис. **¿fig:002?**)





Весь код не включая функции “attacker\_decipher” эмулирует работу шифровальщика в одним ключём за счёт чего становится крайне не безопасным. Из-за свойств функции xor злоумышленик может расшифровать закодированные значения не зная ключи шифрования. Для этого ему надо пару Расшифрованное значение и Зашифрованное и другое сообщение которое хочет расшифровать. То есть нужно знать C1\_hex C2\_hex и P1\_расшифрованный.

Посмотрим на выввод программы с имитацией атаки (рис. **¿fig:003?**)



Как мы видим злоумышленик успешно смог расшифровать значение P2 при том что не узнавал ключ шифрования. Почему это так?

В режиме однократного гаммирования (one-time pad), если один ключ используется для шифрования двух разных сообщений, возникает уязвимость. Это происходит из-за свойства операции XOR.

Для двух шифротекстов (C1) и (C2), зашифрованных с использованием одного ключа (K):

C1 = P1 xor K

C2 = P2 xor K

Злоумышленник может вычислить:

C1 xor C2 = (P1 xor K) xor (P2 xor K)

Из свойства операции XOR:

(P1 xor K) xor (P2 xor K) = P1 xor P2

Таким образом, злоумышленник получает:

P1 xor P2 = C1 xor C2

Предположим, что злоумышленник знает текст (P1) (или его формат). Тогда он может использовать известный текст (P1) для вычисления (P2).

1.Злоумышленник вычисляет (C1 xor C2):

P1 xor P2 = C1 xor C2

2.Зная (P1), злоумышленник может вычислить (P2):

P2 = (C1 xor C2) xor P1

## 2.2 Ответы на контрольные вопросы

1.Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Если злоумышленник знает один из текстов (P1) или (P2) и имеет доступ к шифротекстам (C1) и (C2), он может вычислить другой текст, используя следующее свойство XOR:

1.Вычислите (C1 xor C2):

P1 xor P2 = C1 xor C2

2.Используя известный текст (P1), вычислите (P2):

P2 = (C1 xor C2) xor P1

2.Что будет при повторном использовании ключа при шифровании текста?

При повторном использовании ключа для шифрования двух разных текстов возникает уязвимость, так как злоумышленник может вычислить XOR шифротекстов:

C1 xor C2 = P1 xor P2

Это позволяет злоумышленнику, зная один из открытых текстов, определить второй текст, не зная ключа.

3.Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

Режим однократного гаммирования с одним ключом для двух текстов реализуется следующим образом:

1.Задается ключ (K).

2.Каждый текст шифруется с использованием ключа K и операции XOR:

C1 = P1 xor K

C2 = P2 xor K

4.Перечислите недостатки шифрования одним ключом двух открытых текстов.

Недостатки шифрования одним ключом двух открытых текстов включают:

1.Уязвимость к атаке на основе известного текста: Если злоумышленник знает один из текстов, он может вычислить второй текст.

2.Повторное использование ключа: Приводит к уязвимости, позволяющей вычислить XOR двух текстов.

3.Отсутствие безопасности: Повторное использование ключа нарушает принцип безопасности однократного гаммирования.

4.Перечислите преимущества шифрования одним ключом двух открытых текстов.

Преимущества шифрования одним ключом двух открытых текстов включают:

1.Простота реализации: Легко реализовать с использованием операции XOR.

2.Эффективность: Быстрое шифрование и дешифрование, так как операция XOR выполняется быстро.

3.Минимальное использование ресурсов: Не требует сложных вычислений или большого объема памяти.

Однако эти преимущества применимы только в контексте простоты и эффективности реализации, и не гарантируют безопасности, если ключ используется более одного раза.

# 3 Вывод

Используя этот метод, злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить. Достаточно иметь доступ к одному из шифротекстов и знать или предполагать формат одного из открытых текстов.

Этот подход демонстрирует уязвимость однократного гаммирования при повторном использовании ключа для шифрования разных сообщений.