readme

倪浚桐

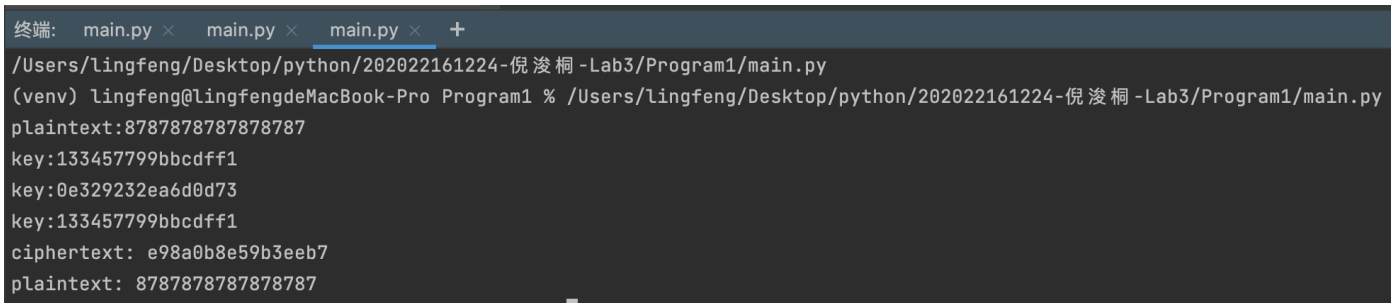
202022161224

Lab3-progarm1

macOS Big Sur 11.5.2(20G95)

Pycharm 11.0.11 x86-64

Python 3.9.5




```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from libdes import DES_Encrypt, DES_Decrypt


def validate_des_key(key: bytes) -> bool:
    for keyByte in key:
        binStr: str = "{0:0>8b}".format(keyByte)
        if sum([1 if b == '1' else 0 for b in binStr]) % 2 == 0:
            return False
    return True


def tri_DES_encrypt(plaintext_Hex: str, key1_Hex: str, key2_Hex: str, key3_Hex:
str) -> bytes:
    ciphertext_ans: bytes = DES_Encrypt(
        bytes.fromhex(plaintext_Hex),
        bytes.fromhex(key1_Hex),
    )
    ciphertext_ans: bytes = DES_Decrypt(
        ciphertext_ans,
        bytes.fromhex(key2_Hex),
    )
```

```
    ciphertext_ans: bytes = DES_Encrypt(
        ciphertext_ans,
        bytes.fromhex(key3_Hex),
    )
    return ciphertext_ans


def tri_DES_decrypt(ciphertext_Hex: str, key1_Hex: str, key2_Hex: str, key3_Hex:
str) -> bytes:
    plaintext_ans: bytes = DES_Decrypt(
        bytes.fromhex(ciphertext_Hex),
        bytes.fromhex(key3_Hex),
    )
    plaintext_ans: bytes = DES_Encrypt(
        plaintext_ans,
        bytes.fromhex(key2_Hex),
    )
    plaintext_ans: bytes = DES_Decrypt(
        plaintext_ans,
        bytes.fromhex(key1_Hex),
    )
    return plaintext_ans


if __name__ == '__main__':
    plaintextHex: str = input('plaintext:')
    key1Hex: str = input('key:')
    key2Hex: str = input('key:')
    key3Hex: str = input('key:')

    if not validate_des_key(bytes.fromhex(key1Hex)):
        raise Exception('Parity check failed on the key.')
    if not validate_des_key(bytes.fromhex(key2Hex)):
        raise Exception('Parity check failed on the key.')
    if not validate_des_key(bytes.fromhex(key3Hex)):
        raise Exception('Parity check failed on the key.')

    ciphertext: bytes = tri_DES_encrypt(
        plaintextHex,
        key1Hex,
        key2Hex,
        key3Hex,
    )

    print('ciphertext:', ciphertext.hex())

    plaintext: bytes = tri_DES_decrypt(
        ciphertext.hex(),
        key1Hex,
```

```
        key2Hex,
        key3Hex,
    )

    print('plaintext:', plaintext.hex())

```