Q1. Implement the concept of Cipher Block Chaining. The encrypt/decrypt function should be DES/RC5.

Code:

```python
from des import DesKey

key = DesKey(b"some key")

encrypted_message = []

decrypted_message = []

def des_encrypt(message):

  for i in message:

    encrypted_message.append(key.encrypt(i,padding=True))

  return encrypted_message

def des_decrypt(message):

  for i in message:

    decrypted_message.append(key.decrypt(i,padding=True))

  return decrypted_message

messages = [b"Hi I am Arun", b"Kathmandu University"]

des_encrypt(messages)

print(encrypted_message)



des_decrypt(encrypted_message)

print(decrypted_message)

encrypted_message = []
```

```
decrypted_message = []

class CipherBlockChain:

  def __init__(self,message):

    self.message = message

  def des_encrypt(self,message):

    for i in self.message:

      encrypted_message.append(key.encrypt(i,padding=True))

    return encrypted_message

  def des_decrypt(self, message):

    for i in message:

      decrypted_message.append(key.decrypt(i,padding=True))

    return decrypted_message

text = CipherBlockChain([b"Hello World"])

print(text.message)

en = text.des_encrypt(text.message)

print(en)



print(text.des_decrypt(en))
```

Description :

Cipher block chaining (CBC) is a mode of operation for a block cipher. Cipher block chaining uses what is known as an initialization vector (IV) of a certain length. One of its key characteristics is that it uses a chaining mechanism that causes the decryption of a block of ciphertext to depend on all the preceding ciphertext blocks. As a result, the entire validity of all preceding blocks is contained in the immediately previous ciphertext block. A single bit error in a ciphertext block affects the decryption of all subsequent blocks. Rearrangement of the order of the ciphertext blocks causes decryption to become corrupted. Basically, in cipher block chaining, each plaintext block is XORed with the immediately previous ciphertext block, and then encrypted.

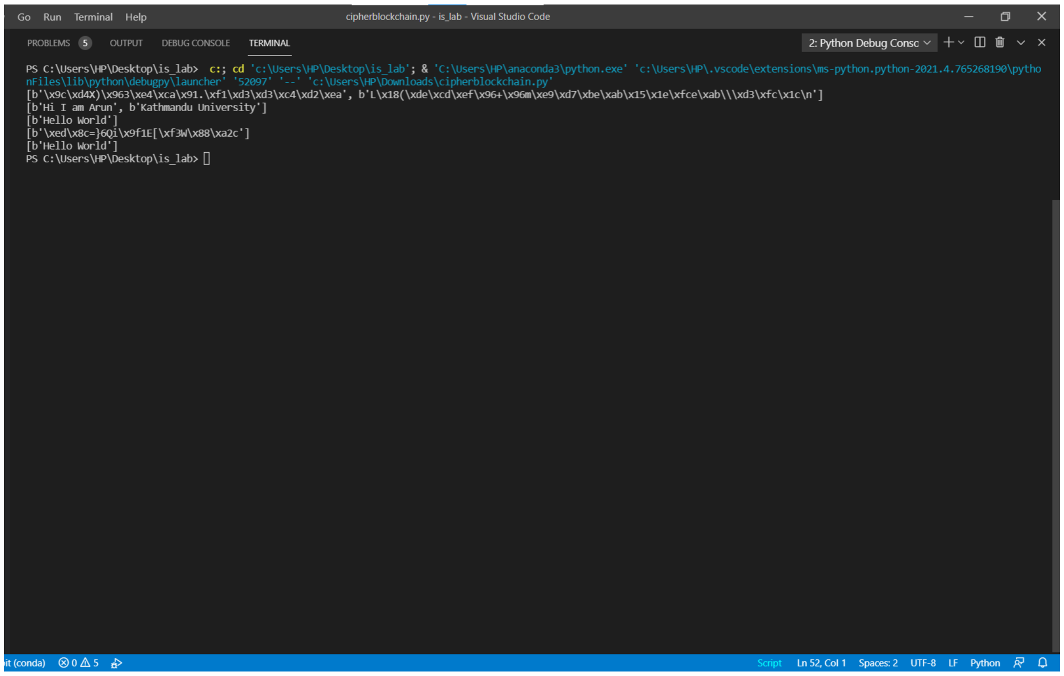
Identical ciphertext blocks can only result if the same plaintext block is encrypted using both the same key and the initialization vector, and if the ciphertext block order is not changed. It has the advantage over the Electronic Code Book mode in that the XORing process hides plaintext patterns.

Key Features:

In cipherblockchain.py, a des library which is a pure python implementation for the famous DES (Data Encryption Standard) algorithm, has been implemented in this code. Firstly, 'DesKey' object is defined by passing the encryption / decryption key. Whether a key is a DES or 3DES algorithm can be known by calling its method is_single() or is_triple(). The messages are encrypted or decrypted by calling the method encrypt() or decrypt() from the DesKey object.

In substitution.py, a string of both lower and upper case letters, called plain text is taken as an input. Then a list of all the characters entered is created along with a dictionary to store the substitution for all characters. For each character, the given character is transformed as per the rule, depending on whether to encrypt or decrypt the text.

Output:



```
PS C:\Users\HP\Desktop\is_lab>  c:; cd 'c:\Users\HP\Desktop\is_lab'; & 'C:\Users\HP\anaconda3\python.exe' 'c:\Users\HP\.vscode\extensions\ms-python.python-2021.4.765268190\pytho
nFiles\lib\python\debugpy\launcher' '52097' '--' 'c:\Users\HP\Downloads\cipherblockchain.py'
[b'\x9c\xd4X)\x963\xe4\xca\x91.\xf1\xd3\xd3\xc4\xd2\xea', b'L\x18(\xde\xcd\xef\x96+\x96m\xe9\xd7\xbe\xab\x15\x1e\xfce\xab\\\xd3\xfc\x1c\n']
[b'Hi I am Arun', b'Kathmandu University']
[b'Hello World']
[b'\xed\x8c=}6Qi\x9f1E[\xf3W\x88\xa2c']
[b'Hello World']
PS C:\Users\HP\Desktop\is_lab> 
```

Q.2 Implement any of the stream cipher techniques (substitution/transposition/product).

```python
# we need 2 helper mappings, from letters to ints and the inverse

L2I = dict(zip("ABCDEFGHIJKLMNOPQRSTUVWXYZ", range(26)))

I2L = dict(zip(range(26), "ABCDEFGHIJKLMNOPQRSTUVWXYZ"))

key = 7

plaintext = input("Enter the plain text: ")

# encipher

ciphertext = ""

for c in plaintext.upper():
    if c.isalpha():


  ciphertext += I2L[(L2I[c] + key) % 26]

    else:

        ciphertext += c

# decipher

plaintext2 = ""

for c in ciphertext.upper():

    if c.isalpha():

        plaintext2 += I2L[(L2I[c] - key) % 26]

    else:

        plaintext2 += c

print (plaintext)

print (ciphertext)

print (plaintext2)
```

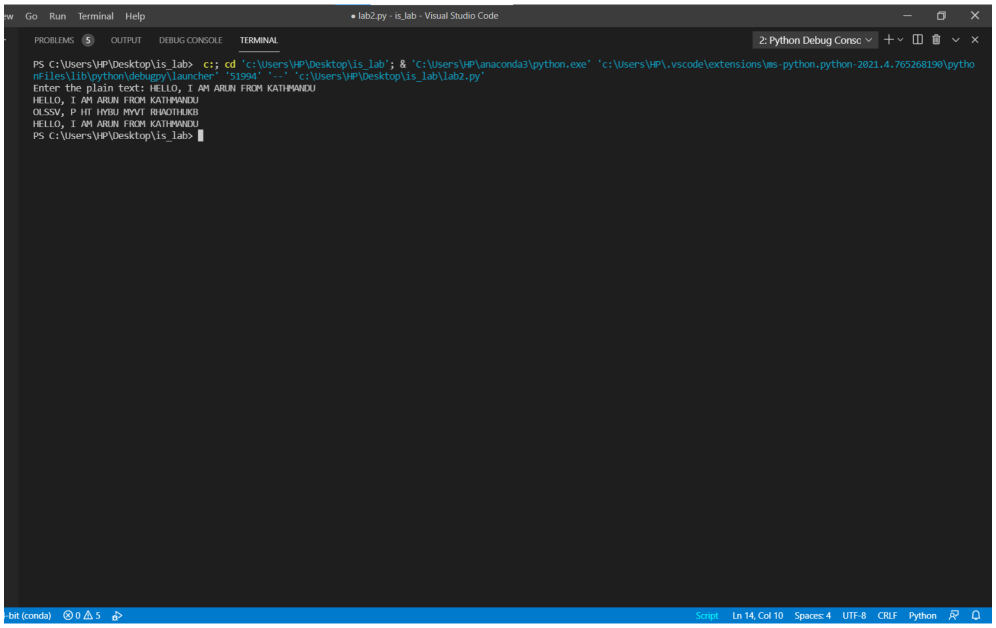
Description :

A stream cipher is a symmetric key cipher where plaintext digits are combined with a pseudorandom cipher digit stream (keystream). In a stream cipher, each plaintext digit is encrypted one at a time with the corresponding digit of the keystream, to give a digit of the ciphertext stream. Since encryption of each digit is dependent on the current state of the cipher, it is also known as a state cipher. In practice, a digit is typically a bit and the combining operation is an exclusive-or (XOR).

I have implemented a substitution technique for stream cipher.

When the program is runned, at first the input text is taken from the user which is then encrypted and decrypted. All the three, entered, encrypted and decrypted text are shown to the user.

Output:


ew Go Run Terminal Help
● lab2.py - is_lab - Visual Studio Code
PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL
2: Python Debug Conso
Script Ln 14, Col 10 Spaces: 4 UTF-8 CRLF Python

```
PS C:\Users\HP\Desktop\is_lab>  c:; cd 'c:\Users\HP\Desktop\is_lab'; & 'C:\Users\HP\anaconda3\python.exe' 'c:\Users\HP\.vscode\extensions\ms-python.python-2021.4.765268190\pytho
nFiles\lib\python\debugpy\launcher' '51994' '--' 'c:\Users\HP\Desktop\is_lab\lab2.py'
Enter the plain text: HELLO, I AM ARUN FROM KATHMANDU
HELLO, I AM ARUN FROM KATHMANDU
OLSSV, P HT HYBU MYVT RHAOTHUKB
HELLO, I AM ARUN FROM KATHMANDU
PS C:\Users\HP\Desktop\is_lab>
```