from collections import deque

from io import StringIO

from itertools import repeat

from sys import version\_info

import binascii

class Trivium:

def \_\_init\_\_(self, key, iv):

self.state = None

self.counter = 0

self.key = key

self.iv = iv

init\_list = list(map(int, list(self.key)))

init\_list += list(repeat(0, 13))

init\_list += list(map(int, list(self.iv)))

init\_list += list(repeat(0, 4))

init\_list += list(repeat(0, 108))

init\_list += list([1, 1, 1])

self.state = deque(init\_list)

for i in range(4\*288):

self.\_gen\_keystream()

def \_gen\_keystream(self):

t\_1 = self.state[65] ^ self.state[92]

t\_2 = self.state[161] ^ self.state[176]

t\_3 = self.state[242] ^ self.state[287]

z = t\_1 ^ t\_2 ^ t\_3

t\_1 = t\_1 ^ self.state[90] & self.state[91] ^ self.state[170]

t\_2 = t\_2 ^ self.state[174] & self.state[175] ^ self.state[263]

t\_3 = t\_3 ^ self.state[285] & self.state[286] ^ self.state[68]

self.state.rotate()

self.state[0] = t\_3

self.state[93] = t\_1

self.state[177] = t\_2

return z

def keystream(self):

while self.counter < 2\*\*64:

self.counter += 1

yield self.\_gen\_keystream()

\_allbytes = dict([("%02X" % i, i) for i in range(256)])

def \_hex\_to\_bytes(s):

return [\_allbytes[s[i:i+2].upper()] for i in range(0, len(s), 2)]

def hex\_to\_bits(s):

return [(b >> i) & 1 for b in \_hex\_to\_bytes(s)

for i in range(8)]

#Chuyển từ bit sang hexa

def bits\_to\_hex(b):

return "".join(["%02X" % sum([b[i + j] << j for j in range(8)])

for i in range(0, len(b), 8)])

def strxor(a, b):

return "".join([str(hex(x ^ y)) for x, y in zip(a, b)])

KEY = hex\_to\_bits("0F62B5085BAE0154A7FA")[::-1]

IV = hex\_to\_bits("288FF65DC42B92F960C7")[::-1]

trivium = Trivium(KEY, IV)

if version\_info[0] == 2:

next\_key\_bit = trivium.keystream().next

elif version\_info[0] == 3:

next\_key\_bit = trivium.keystream().\_\_next\_\_

else:

print("invalid python version")

exit()

print("Input:",end='')

stt = input()

st = stt.encode('ascii')

for i in range(1):

keystream = []

for j in range(len(stt)\*8):

keystream.append(next\_key\_bit())

key = binascii.unhexlify(''.join(bits\_to\_hex(keystream)))

string = str(binascii.hexlify(st),'ascii')

for i in range(1):

stringstream = []

for j in range(len(stt)\*8):

stringstream.append(next\_key\_bit())

stringst = binascii.unhexlify(''.join(bits\_to\_hex(stringstream)))

xorst = strxor(key,stringst)

print(f"Ouput:{(''.join(xorst.split('0x'))).upper()}")