text = "HELLO"

shift = 3

for c in text:

print(chr(ord(c) + shift), end="")

#ceaser cipher encryption

text = "KHOOR"

shift = 3

for c in text:

print(chr(ord(c) - shift), end="")

#ceaser cipher decryption

p = 23

g = 5

a = 6

b = 15

A = (g \*\* a) % p

B = (g \*\* b) % p

key1 = (B \*\* a) % p

key2 = (A \*\* b) % p

print(key1, key2)

# Diffie-Hellman Key Exchange:

key = "KEYWORDABCDEFGHIKLMNPQSTUVXZ"

text = "HELLOX"

m = []

for c in key:

if c not in m:

m.append(c)

cipher = ""

for i in range(0, len(text), 2):

a, b = text[i], text[i+1]

r1, c1 = divmod(m.index(a), 5)

r2, c2 = divmod(m.index(b), 5)

if r1 == r2:

cipher += m[r1 \* 5 + (c1 + 1) % 5] + m[r2 \* 5 + (c2 + 1) % 5]

elif c1 == c2:

cipher += m[((r1 + 1) % 5) \* 5 + c1] + m[((r2 + 1) % 5) \* 5 + c2]

else:

cipher += m[r1 \* 5 + c2] + m[r2 \* 5 + c1]

print(cipher)

#Playfair cipher encryption

key = "KEYWORDABCDEFGHIKLMNPQSTUVXZ"

text = "HELLOX"

m = []

for c in key:

if c not in m:

m.append(c)

cipher = ""

for i in range(0, len(text), 2):

r1, c1 = divmod(m.index(text[i]), 5)

r2, c2 = divmod(m.index(text[i+1]), 5)

if r1 == r2:

cipher += m[r1 \* 5 + (c1 + 1) % 5] + m[r2 \* 5 + (c2 + 1) % 5]

elif c1 == c2:

cipher += m[((r1 + 1) % 5) \* 5 + c1] + m[((r2 + 1) % 5) \* 5 + c2]

else:

cipher += m[r1 \* 5 + c2] + m[r2 \* 5 + c1]

print(cipher)

#playfair decryption

plain = "HELLO"

key = "KEY"

cipher = ""

for i in range(len(plain)):

shift = ord(key[i % len(key)]) - 65

cipher += chr((ord(plain[i]) - 65 + shift) % 26 + 65)

print(cipher)

#polyalphabetic substitution.

plain = "HELLO"

key = "QWERTYUIOPASDFGHJKLZXCVBNM"

cipher = ""

for c in plain:

cipher += key[ord(c) - 65]

print(cipher)

#mono alphabetic substitution

plain = "HELLO"

key = "XMCKL"

cipher = ""

for i in range(len(plain)):

cipher += chr((ord(plain[i]) ^ ord(key[i])) + 65)

print(cipher)

#vernam cipher encryption

cipher = "ZEBBW"

key = "XMCKL"

plain = ""

for i in range(len(cipher)):

plain += chr((ord(cipher[i]) ^ ord(key[i])) + 65)

print(plain)

#vernam cipher decryption

text = "KHOOR"

for shift in range(1, 26):

print(f"Shift {shift}: ", end="")

for c in text:

print(chr(ord(c) - shift), end="")

print()

#brute force

plain = "HELLO"

key = "KEY"

cipher = ""

for i in range(len(plain)):

shift = ord(key[i % len(key)]) - 65

cipher += chr((ord(plain[i]) - 65 + shift) % 26 + 65)

print(cipher)

#vigenere cipher encryption

cipher = "RIJVS"

key = "KEY"

plain = ""

for i in range(len(cipher)):

shift = ord(key[i % len(key)]) - 65

plain += chr((ord(cipher[i]) - 65 - shift) % 26 + 65)

print(plain)

#vigenere cipher decryption

plain = "HELLO"

a, b = 5, 8 # Key values

cipher = ""

for c in plain:

cipher += chr(((a \* (ord(c) - 65) + b) % 26) + 65)

print(cipher)

#Affine cipher encryption

plain = "HELLO"

key = 3

cipher = ""

for c in plain:

cipher += chr((ord(c) - 65 + key) % 26 + 65)

print(cipher)

#additive cipher

plain = "HELLOWORLD"

cols = 4

cipher = [""] \* cols

for i in range(len(plain)):

cipher[i % cols] += plain[i]

print("".join(cipher))

#row column encryption

cipher = "HOLWELORDLD"

cols = 4

rows = -(-len(cipher) // cols)

plain = [""] \* rows

for i in range(len(cipher)):

plain[i % rows] += cipher[i]

print("".join(plain))

#row column decryption

plain = "HELLOWORLD"

rails = 3

cipher = [""] \* rails

row, step = 0, 1

for c in plain:

cipher[row] += c

if row == 0:

step = 1

elif row == rails - 1:

step = -1

row += step

print("".join(cipher))

#rail fence encryption

cipher = "HOEWRLLDLOL"

rails = 3

idx, row, step = 0, 0, 1

pos = [""] \* len(cipher)

for \_ in cipher:

pos[idx], step = row, 1 if row == 0 else -1 if row == rails - 1 else step

row, idx = row + step, idx + 1

plain = [""] \* len(cipher)

i = 0

for r in range(rails):

for j in range(len(cipher)):

if pos[j] == r:

plain[j] = cipher[i]

i += 1

print("".join(plain))

#rail fence decryption

p, q = 61, 53

n = p \* q

phi = (p - 1) \* (q - 1)

e = 17

d = pow(e, -1, phi)

msg = 65

cipher = pow(msg, e, n)

plain = pow(cipher, d, n)

print(cipher, plain)

#RSA algo

key = [[6, 24], [1, 13]]

plain = "HI"

text = [ord(plain[0]) - 65, ord(plain[1]) - 65]

cipher = [

(key[0][0] \* text[0] + key[0][1] \* text[1]) % 26,

(key[1][0] \* text[0] + key[1][1] \* text[1]) % 26

]

print(chr(cipher[0] + 65) + chr(cipher[1] + 65))

#hill cipher

import hashlib

text = "HELLO"

print(hashlib.sha1(text.encode()).hexdigest())

#SHAzzz