**EXPERIMENT NO 7**

**OBJECTIVE**- implement a code for HILL CIPHER

**SOURCE CODE:**

import sys

import numpy as np

def cipher\_encryption():

msg=input("Entermessage:").upper() msg=msg.replace("","")

#ifmessagelengthisoddnumber,append0attheend len\_chk =0

if len(msg) % 2 != 0:

msg += "0"

len\_chk = 1

# msg to matrices row = 2

col = int(len(msg)/2)

msg2d = np.zeros((row, col), dtype=int)

itr1 =0

itr2 =0

for i in range(len(msg)):

if i % 2 == 0:

msg2d[0][itr1] = int(ord(msg[i])-65) itr1 += 1

else:

msg2d[1][itr2] = int(ord(msg[i])-65)

itr2 += 1 # for

key=input("Enter4letterKeyString:").upper() key=key.replace("","")

# key to 2x2

key2d=np.zeros((2,2),dtype=int) itr3 =0

for i in range(2):

for j in range(2):

key2d[i][j] = ord(key[itr3])-65 itr3 += 1

#checkingvalidityofthekey # findingdeterminant

deter=key2d[0][0]\*key2d[1][1]-key2d[0][1]\*key2d[1][0] deter=deter%26

# finding multiplicative inverse mul\_inv = -1

for i in range(26): temp\_inv=deter\*i

iftemp\_inv%26==1: mul\_inv =i

break else:

continue # for

if mul\_inv == -1:

print("Invalid key") sys.exit()

# if

encryp\_text = ""

itr\_count = int(len(msg)/2) if len\_chk == 0:

for i in range(itr\_count):

temp1=msg2d[0][i]\*key2d[0][0]+msg2d[1][i]\*key2d[0][1] encryp\_text+=chr((temp1%26)+65)

temp2=msg2d[0][i]\*key2d[1][0]+msg2d[1][i]\*key2d[1][1] encryp\_text+=chr((temp2%26)+65)

# for else:

for i in range(itr\_count-1):

temp1=msg2d[0][i]\*key2d[0][0]+msg2d[1][i]\*key2d[0][1] encryp\_text+=chr((temp1%26)+65)

temp2=msg2d[0][i]\*key2d[1][0]+msg2d[1][i]\*key2d[1][1] encryp\_text+=chr((temp2%26)+65)

# for # if else

print("Encrypted Text: {}".format(encryp\_text))

def cipher\_decryption():

msg=input("Entermessage:").upper() msg=msg.replace("","")

#ifmessagelengthisoddnumber,append0attheend len\_chk =0

if len(msg) % 2 != 0:

msg += "0"

len\_chk = 1

# msg to matrices row = 2

col = int(len(msg) / 2)

msg2d = np.zeros((row, col), dtype=int)

itr1 =0

itr2 =0

for i in range(len(msg)):

if i % 2 == 0:

msg2d[0][itr1] = int(ord(msg[i]) - 65) itr1 += 1

else:

msg2d[1][itr2] = int(ord(msg[i]) - 65) itr2 += 1

# for

key=input("Enter4letterKeyString:").upper() key=key.replace("","")

# key to 2x2

key2d=np.zeros((2,2),dtype=int) itr3 =0

for i in range(2):

for j in range(2):

key2d[i][j]=ord(key[itr3])-65 itr3 +=1

# for

# finding determinant

deter=key2d[0][0]\*key2d[1][1]-key2d[0][1]\*key2d[1][0] deter=deter%26

# finding multiplicative inverse mul\_inv = -1

for i in range(26):

temp\_inv = deter \* i

if temp\_inv % 26 == 1:

mul\_inv = i break

else:

continue # for

# adjugate matrix # swapping

key2d[0][0], key2d[1][1] = key2d[1][1], key2d[0][0]

#changingsigns key2d[0][1] \*=-1

key2d[1][0] \*=-1

key2d[0][1]=key2d[0][1]%26

key2d[1][0]=key2d[1][0]%26

# multiplying multiplicative inverse with adjugate matrix for i in range(2):

for j in range(2): key2d[i][j] \*= mul\_inv

# modulo

for i in range(2):

for j in range(2):

key2d[i][j] = key2d[i][j] % 26

#ciphertoplain decryp\_text=""

itr\_count=int(len(msg)/2) iflen\_chk==0:

for i in range(itr\_count):

temp1=msg2d[0][i]\*key2d[0][0]+msg2d[1][i]\*key2d[0][1] decryp\_text+=chr((temp1%26)+65)

temp2=msg2d[0][i]\*key2d[1][0]+msg2d[1][i]\*key2d[1][1] decryp\_text+=chr((temp2%26)+65)

# for else:

for i in range(itr\_count - 1):

temp1=msg2d[0][i]\*key2d[0][0]+msg2d[1][i]\*key2d[0][1] decryp\_text+=chr((temp1%26)+65)

temp2=msg2d[0][i]\*key2d[1][0]+msg2d[1][i]\*key2d[1][1] decryp\_text+=chr((temp2%26)+65)

# for # if else

print("Decrypted Text: {}".format(decryp\_text))

def main():

choice=int(input("1.Encryption\n2.Decryption\nChoose(1,2):")) if choice ==1:

print("---Encryption---") cipher\_encryption()

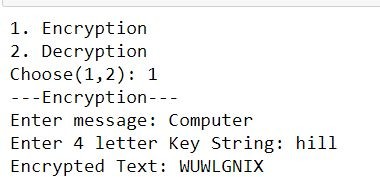
elif choice == 2:

print("---Decryption---") cipher\_decryption()

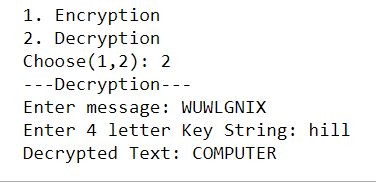
else:

print("Invalid Choice")

ifname == "main": main()

**Output:** Encryption:

Decryption:



**Developed by:** Abhishek Pandey