**PRACTICAL 1:**

**Implement Caesar cipher encryption - decryption.**

Program:

def encrypt(plain\_text,s):

result=""

for x in range(len(plain\_text)):

char = plain\_text[x].upper()

if(char==""):

result+=""

else:

result+=chr((ord(char)+s-65)%26+65)

return result

def decrypt(crypted\_text,s):

result=""

for i in range(len(crypted\_text)):

char = crypted\_text[i].upper()

if(char==""):

result+=""

else:

result+=chr((ord(char)-s-65)%26+65)

return result

text=input("Enter the text to be encrypted: ")

shift=int(input("enter additive keys: "))

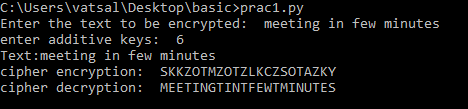
encrypted\_text=encrypt(text,shift)

print("Text:"+text)

print("cipher encryption: "+encrypted\_text)

print("cipher decryption: "+decrypt(encrypted\_text,shift))

Output:



**PRACTICAL 2:**

**Implement Monoalphabetic /transposition cipher encryption - decryption.**

Program:

transposition\_cipher = {'a': 'X','b': 'D','c': 'G','d': 'S','e': 'Z','f': 'A','g': 'N','h': 'Y',

'i': 'O','j': 'B','k': 'T','l': 'M','m': 'J','n': 'C','o': 'E','p': 'V','q': 'F','r': 'H',

's': 'K','t': 'W','u': 'P','v': 'L','w': 'Q','x': 'U','y': 'R','z': 'I',' ': ' '}

inverse\_transpostion\_cipher={}

for key,value in transposition\_cipher.items():

inverse\_transpostion\_cipher[value]=key

def encrypt(message):

encrypted\_message=[]

for letter in message:

encrypted\_message.append(transposition\_cipher.get(letter,letter))

print("".join(encrypted\_message))

def decrypt(message):

decrypted\_message=[]

for letter in message:

decrypted\_message.append(inverse\_transpostion\_cipher.get(letter,letter))

print("".join(decrypted\_message))

mode=input("Do you want to encrypt or decrypt? ")

message=input("Please enter the message :")

if(mode.lower()=="encrypt"):

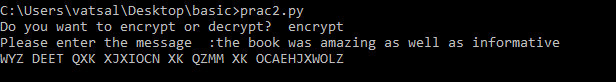
encrypt(message)

else:

decrypt(message)

Output:

Encryption :



Decryption:

