**Practical-3**

**Aim : Alice wants to send some confidential information to Bob over a secure network, you have to create perform following task :  
1)Provide Security using Caesar Cipher Algorithm  
2)Find the all possible Cipher Text & Plaintext pairs  
3)Provide Security Mono-alphabetic Cipher Algorithm**

**Caesar Cipher**

def encrypt(plain\_text, key):

    cipher\_text = ""

    for char in plain\_text:

        if char.isalpha():

            ascii\_offset = ord('a')

            encrypted\_char = chr((ord(char.lower()) - ascii\_offset + key) % 26 + ascii\_offset)

            cipher\_text += encrypted\_char

        else:

            cipher\_text += char

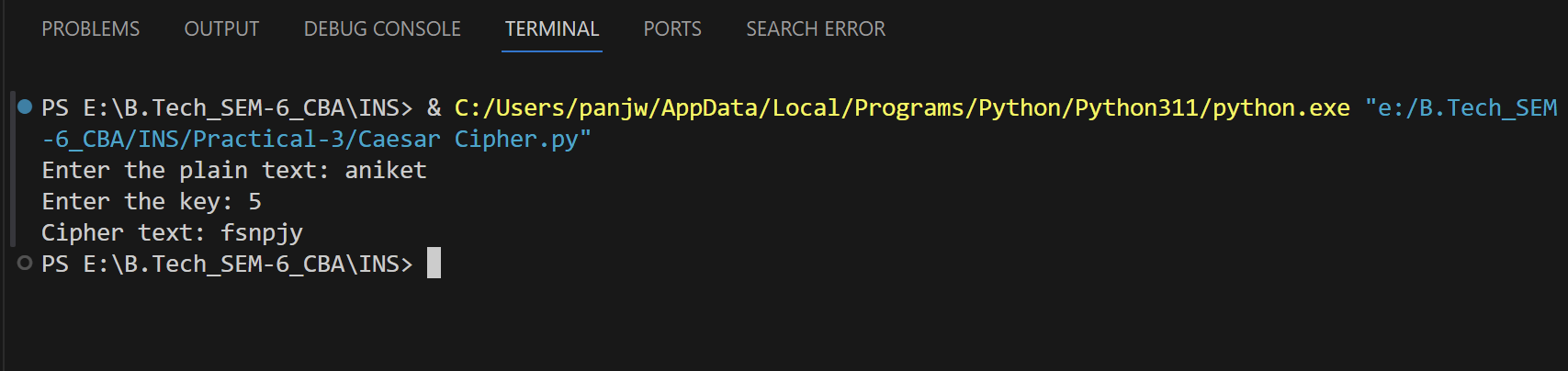
    return cipher\_text

plain\_text = input("Enter the plain text: ").lower()

key = int(input("Enter the key: "))

cipher\_text = encrypt(plain\_text, key)

print("Cipher text:", cipher\_text)

****

**Find the all possible Cipher Text & Plaintext pairs**

def decrypt(cipher\_text, key):

    plaintext = ""

    for char in cipher\_text:

        if char.isalpha():

            ascii\_val = ord(char)

            decrypted\_val = (ascii\_val - key) % 26

            decrypted\_char = chr(decrypted\_val + ord('A'))

            plaintext += decrypted\_char

        else:

            plaintext += char

    return plaintext

def find\_all\_plaintexts(cipher\_text):

    all\_plaintexts = []

    for key in range(26):

        plaintext = decrypt(cipher\_text, key)

        all\_plaintexts.append(plaintext)

    return all\_plaintexts

cipher\_text = input("Enter the cipher text: ")

plaintexts = find\_all\_plaintexts(cipher\_text)

print("All possible plaintexts:")

for plaintext in plaintexts:

    print(plaintext)

**output :-**

**A screenshot of a computer

Description automatically generated**

**Provide Security Mono-alphabetic Cipher Algorithm**

def encrypt(plain\_text, key):

    cipher\_text = ""

    for char in plain\_text:

        if char.isalpha():

            if char.isupper():

                index = ord(char) - ord('A')

                cipher\_char = key[index].upper()

            else:

                index = ord(char) - ord('a')

                cipher\_char = key[index].lower()

            cipher\_text += cipher\_char

        else:

            cipher\_text += char

    return cipher\_text

def decrypt(cipher\_text, key):

    plain\_text = ""

    for char in cipher\_text:

        if char.isalpha():

            if char.isupper():

                index = ord(char) - ord('A')

                plain\_char = chr(key[index].upper())

            else:

                index = ord(char) - ord('a')

                plain\_char = chr(key[index].lower())

            plain\_text += plain\_char

        else:

            plain\_text += char

    return plain\_text

def main():

    key = []

    for i in range(26):

        key.append(input(f"Enter the key for '{chr(ord('a')+i)}': "))

    plain\_text = input("Enter the plain text: ")

    cipher\_text = encrypt(plain\_text, key)

    print("Encrypted text:", cipher\_text)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**output: -**

**A screen shot of a computer

Description automatically generated**