Exp4) Design and Implement a product cipher using Substitution ciphers and Transposition ciphers.

**Code:**

text = input("Enter the Plaintext: ")

x= [chr(ord(i)+1) for i in text]

for i in range(len(x)):

if x[i] =='{':

x[i]='a'

elif x[i] =='|':

x[i]='b'

elif x[i] =='}':

x[i]='c'

else:

continue

print("Ciphertext: ", end="")

for i in x:

print(i, end="")

key = input("\nEnter the key: ")

no\_of\_columns = len(key)

no\_of\_rows = int(len(x)/len(key))

matrix = []

k = 0

for i in range(0, no\_of\_rows):

a =[]

for j in range(0, no\_of\_columns):

a.append(x[k])

k += 1

matrix.append(a)

def order(s):

ordered = sorted(s)

position = {c: i for i, c in enumerate(ordered)}

return [position[c]+1 for c in s]

key\_no=order(key)

print("")

for i in key:

print(" ",i," ",end="")

print("")

for i in key\_no:

print(" ",i," ",end="")

print("")

for i in matrix:

print(i)

swap = []

def bubbleSort(ar):

n = len(ar)

swapped = False

for i in range(n-1):

for j in range(0, n-i-1):

if ar[j] > ar[j + 1]:

swapped = True

ar[j], ar[j + 1] = ar[j + 1], ar[j]

swap.append([j, j+1])

if not swapped:

return

bubbleSort(key\_no)

for i in swap:

j,k=i

for i in matrix:

i[j],i[k]=i[k],i[j]

print("\nTransposition Cipher: ",end="")

result= [[matrix[j][i] for j in range(len(matrix))] for i in

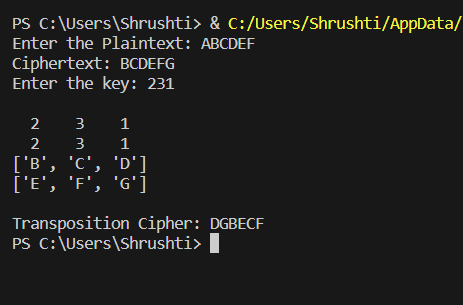
range(len(matrix[0]))]

for i in result:

for j in i:

print(j,end="")

**OUTPUT:**



**Conclusion:**

Hence implemented a product cipher using Substitution ciphers and Transposition ciphers.