Pandit Deendayal Energy University

School of Technology

Information Security Lab

B.Tech-Computer Science & Engineering (Sem-V)

PATEL VEDANT H.

19BCP138

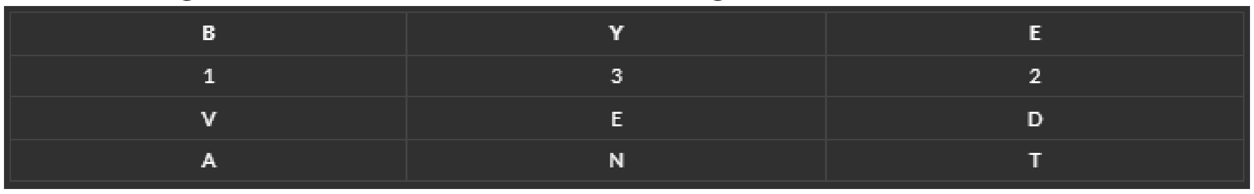
DIVISION – 2

Lab 4 Assignment

- **Aim:** Study and Implement program for Transposition Cipher.
- **Introduction:**

In cryptography, a transposition cipher is a method of encryption by which the positions held by units of plaintext are shifted according to a regular system, so that the cipher text constitutes a permutation of the plaintext. That is, the order of the units is changed. Mathematically a bijective function is used on the characters' positions to encrypt and an inverse function to decrypt.

In a columnar transposition, the message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order. Both the width of the rows and the permutation of the columns are usually defined by a keyword. For example, keyword **BYE** is of length 3 (so rows are of length 3), and the permutation is defined by the alphabetical order of the letters in the keyword. In this case, the order would be **"1 3 2"** and the message **VEDANT** then, we write this into the grid as follows:

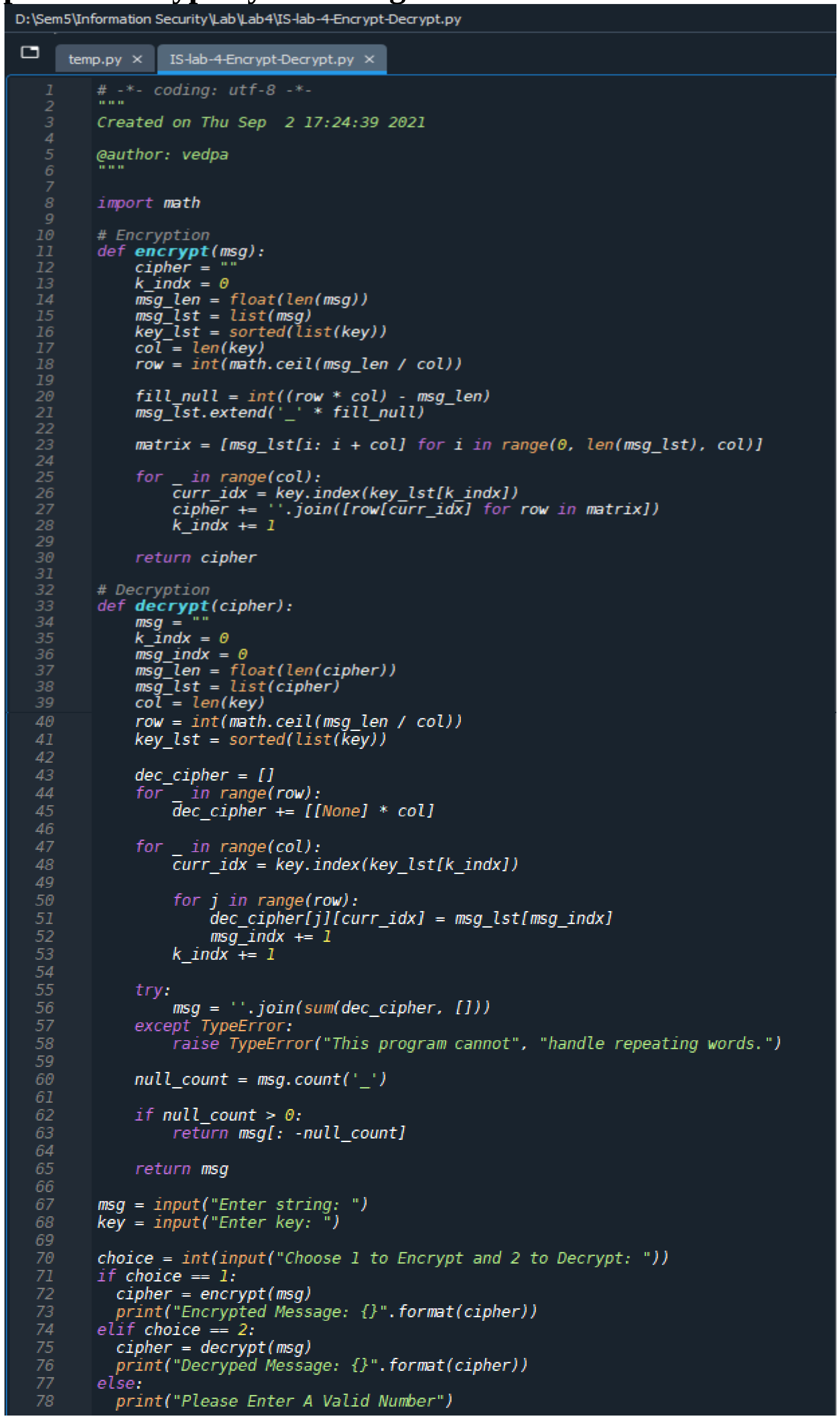


B	Y	E
1	3	2
V	E	D
A	N	T

Now, the message is read off in columns, in the order specified by the keyword and the answer here is **VADTEN** .In a regular columnar transposition cipher, any spare spaces are filled with nulls; in an irregular columnar transposition cipher, the spaces are left blank.

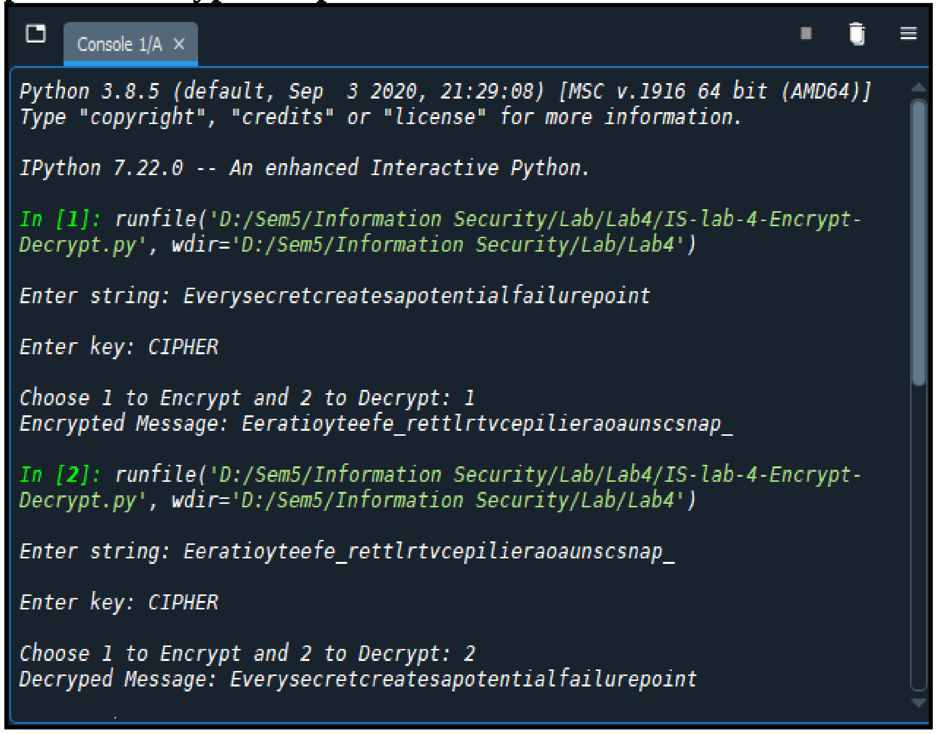
❖ Program:

➢ Encrypt and Decrypt Python Program:


D:\Sem5\Information Security\Lab\Lab4\IS-lab-4-Encrypt-Decrypt.py
temp.py
IS-lab-4-Encrypt-Decrypt.py

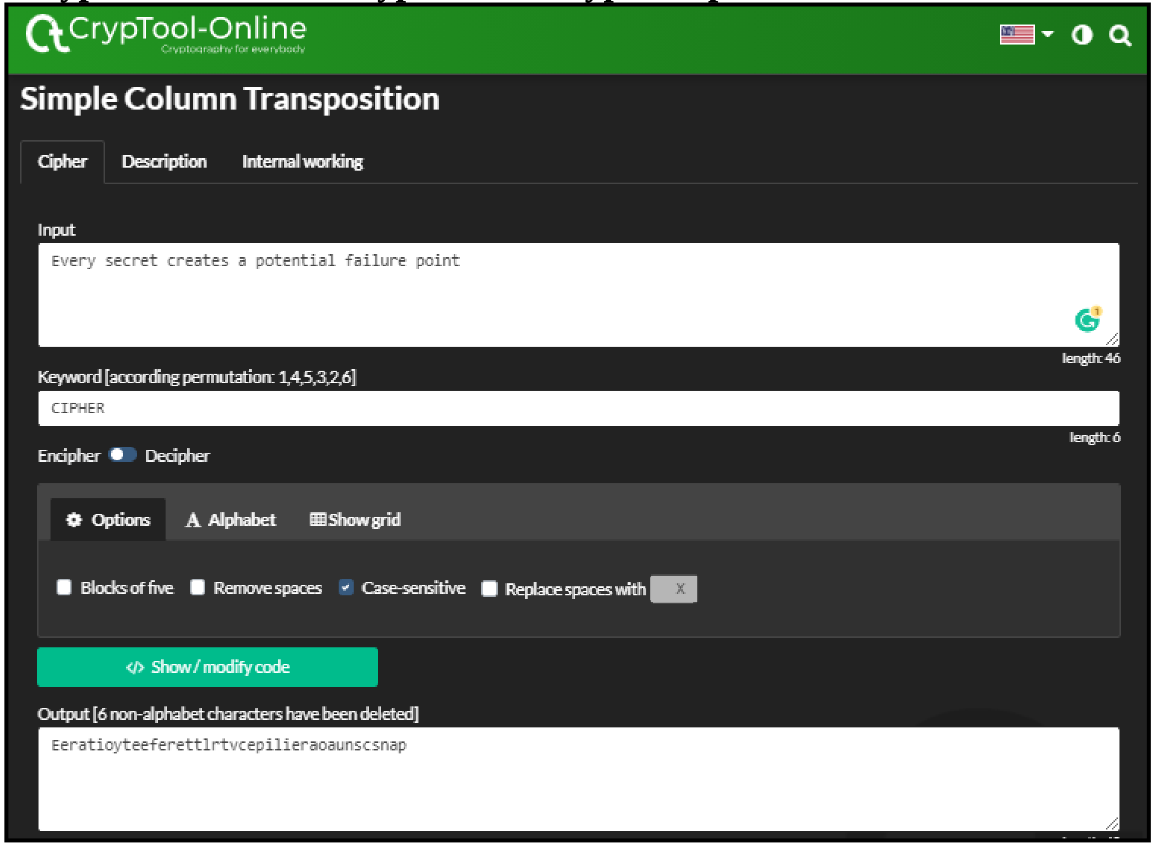
```python
# -*- coding: utf-8 -*-
"""
Created on Thu Sep  2 17:24:39 2021

@author: vedpa
"""

import math

# Encryption
def encrypt(msg):
    cipher = ""
    k_indx = 0
    msg_len = float(len(msg))
    msg_lst = list(msg)
    key_lst = sorted(list(key))
    col = len(key)
    row = int(math.ceil(msg_len / col))

    fill_null = int((row * col) - msg_len)
    msg_lst.extend('_' * fill_null)

    matrix = [msg_lst[i: i + col] for i in range(0, len(msg_lst), col)]

    for _ in range(col):
        curr_idx = key.index(key_lst[k_indx])
        cipher += ''.join([row[curr_idx] for row in matrix])
        k_indx += 1

    return cipher

# Decryption
def decrypt(cipher):
    msg = ""
    k_indx = 0
    msg_indx = 0
    msg_len = float(len(cipher))
    msg_lst = list(cipher)
    col = len(key)
    row = int(math.ceil(msg_len / col))
    key_lst = sorted(list(key))

    dec_cipher = []
    for _ in range(row):
        dec_cipher += [[None] * col]

    for _ in range(col):
        curr_idx = key.index(key_lst[k_indx])

        for j in range(row):
            dec_cipher[j][curr_idx] = msg_lst[msg_indx]
            msg_indx += 1
        k_indx += 1

    try:
        msg = ''.join(sum(dec_cipher, []))
    except TypeError:
        raise TypeError("This program cannot", "handle repeating words.")

    null_count = msg.count('_')

    if null_count > 0:
        return msg[: -null_count]

    return msg

msg = input("Enter string: ")
key = input("Enter key: ")

choice = int(input("Choose 1 to Encrypt and 2 to Decrypt: "))
if choice == 1:
  cipher = encrypt(msg)
  print("Encrypted Message: {}".format(cipher))
elif choice == 2:
  cipher = decrypt(msg)
  print("Decryped Message: {}".format(cipher))
else:
  print("Please Enter A Valid Number")
```

➢ Encrypt and Decrypt Output:



```
Python 3.8.5 (default, Sep  3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.22.0 -- An enhanced Interactive Python.

In [1]: runfile('D:/Sem5/Information Security/Lab/Lab4/IS-lab-4-Encrypt-
Decrypt.py', wdir='D:/Sem5/Information Security/Lab/Lab4')

Enter string: Everysecretcreatesapotentialfailurepoint

Enter key: CIPHER

Choose 1 to Encrypt and 2 to Decrypt: 1
Encrypted Message: Eeratioyteefe_rettlrtvcepilieraoaunscsnap_

In [2]: runfile('D:/Sem5/Information Security/Lab/Lab4/IS-lab-4-Encrypt-
Decrypt.py', wdir='D:/Sem5/Information Security/Lab/Lab4')

Enter string: Eeratioyteefe_rettlrtvcepilieraoaunscsnap_

Enter key: CIPHER

Choose 1 to Encrypt and 2 to Decrypt: 2
Decryped Message: Everysecretcreatesapotentialfailurepoint
```

➢ CrypTool Online Encrypt and Decrypt Output:



Simple Column Transposition

Input

```
Every secret creates a potential failure point
```

length: 46

Keyword [according permutation: 1,4,5,3,2,6]

```
CIPHER
```

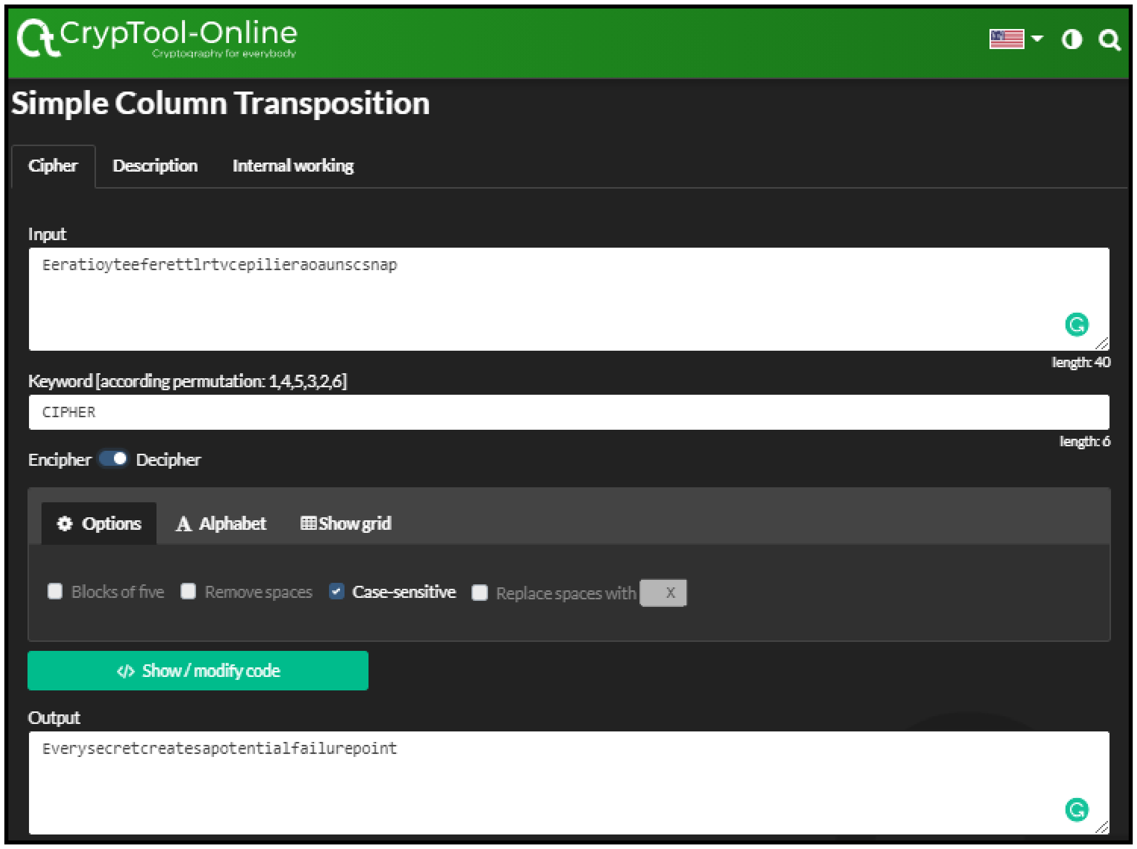
length: 6

Encipher ● Decipher

☐ Blocks of five ☐ Remove spaces ☑ Case-sensitive ☐ Replace spaces with X

Output [6 non-alphabet characters have been deleted]

```
Eeratioyteeferettlrtvcepilieraoaunscsnap
```



Simple Column Transposition

Input
```
Eeratioyteeferettlrtvcepilieraoaunscsnap
```
length: 40

Keyword [according permutation: 1,4,5,3,2,6]
```
CIPHER
```
length: 6

Encipher ● Decipher

☐ Blocks of five ☐ Remove spaces ✓ Case-sensitive ☐ Replace spaces with X

Output
```
Everysecretcreatesapotentialfailurepoint
```

❖ Cryptanalysis:

The key to the transposition cipher is simply the **permutation P**. So, the transposition cipher has the property that the encrypted message contains all the characters that were in the plaintext message. In other words, the unigram statistics for the message are unchanged by the encryption process.

❖ Applications:

- Probably one of the oldest known implementations of the transposition cipher was the Spartan Scytale (also commonly spelled as Skytale). In ancient Greece (around 475 B.C.), the Spartan army commanders created a Scytale, a device they designed for sending secret messages.
- One important strength of transposition ciphers is that they are not susceptible to frequency analysis, since we have not changed the symbols for each letter. If there are 14 "e" in the plaintext, then there will be 14 "E" in the cipher text, just in different positions.

❖ Reference:

- https://en.wikipedia.org/wiki/Transposition_cipher
- https://www.geeksforgeeks.org/columnar-transposition-cipher/
- https://www.cryptool.org/en/cto/transposition