**Final Practical**

1. **AIM**

To Encrypt the message into a cipher text with different encoding steps.

1. **Contribution**

The writing and debugging of code are done by me with the help of Python 3.7 IDLE

* The following code we have used in our program

from random import choice

import onetimepad,string,math,numpy,random,pyotp

def decrypt(m):

    even\_letters = get\_even\_letters(m)

    new\_m = ''.join(even\_letters)

    return new\_m

def get\_t():

    t = input('\nENTER THE OPTION to Encrypted(e) or Decrypted(d) or Quit(q) : ')

    return t

def get\_m():

    m = input('\nENTER MESSAGE : ')

    return m

def is\_even(number):

    return number % 2 == 0

def get\_even\_letters(m):

    even\_letters = []

    for counter in range(0, len(m)):

        if is\_even(counter):

            even\_letters.append(m[counter])

    return even\_letters

def get\_odd\_letters(m):

    odd\_letters = []

    for counter in range(0, len(m)):

        if not is\_even(counter):

            odd\_letters.append(m[counter])

    return odd\_letters

def swap\_letters(m):

    letter\_list = []

    if not is\_even(len(m)):

        # printing letters

        letters = string.ascii\_letters

        f\_l= ''.join(random.choice(letters) for i in range(3))

        # printing digits

        letters = string.digits

        f\_d = ''.join(random.choice(letters) for i in range(2))

        m = m + f\_d + f\_l

    else:

        # printing letters

        letters = string.ascii\_letters

        f\_l= ''.join(random.choice(letters) for i in range(2))

        # printing digits

        letters = string.digits

        f\_d = ''.join(random.choice(letters) for i in range(2))

        m = m + f\_d + f\_l

    even\_letters = get\_even\_letters(m)

    odd\_letters = get\_odd\_letters(m)

    for counter in range(0, int(len(m)/2)):

        letter\_list.append(odd\_letters[counter])

        letter\_list.append(even\_letters[counter])

        new\_m = ''.join(letter\_list)

    return new\_m

def encrypt(m):

    swapped\_m = swap\_letters(m)

    encrypted\_m = ''.join(reversed(swapped\_m))

    return encrypted\_m

def decrypt(m):

    unreversed\_m = ''.join(reversed(m))

    decrypted\_m = swap\_letters(unreversed\_m)

    return decrypted\_m

while True:

    t = get\_t()

    totp = pyotp.TOTP("JBSWY3DPEHPK3PXP")

    if t == 'e':

        m = get\_m()

        encrypted = encrypt(m)

        cipher = onetimepad.encrypt(encrypt(m), 'random')

        print('\nCiphertext of the secret message is:', cipher)

        print("\nCurrent OTP : ", totp.now())

        pin = totp.now()

        print('\n')

    elif t == 'd':

        x = input('\nENTER OTP : ')

        if x == pin:

            m = get\_m()

            msg = onetimepad.decrypt(cipher, 'random')

            m = msg

            decrypted = decrypt(m)

            print('\nPlaintext of the secret message is:', decrypted)

        else:

            print('\n\*\*\*\*\*\*INVALID OTP\*\*\*\*\*\*\*\n')

    elif t == 'q':

        break

* This code is written with many modifications like include of onetimepad, fake letters addition and many more.
* This kind of code is not available on internet as it is a mixture of different concepts.