VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

APPLIED CRYPTOGRAPHY AND NETWORK SECURITY

NAME : MOTHISHWARAN C.

REG No : 19MID0017

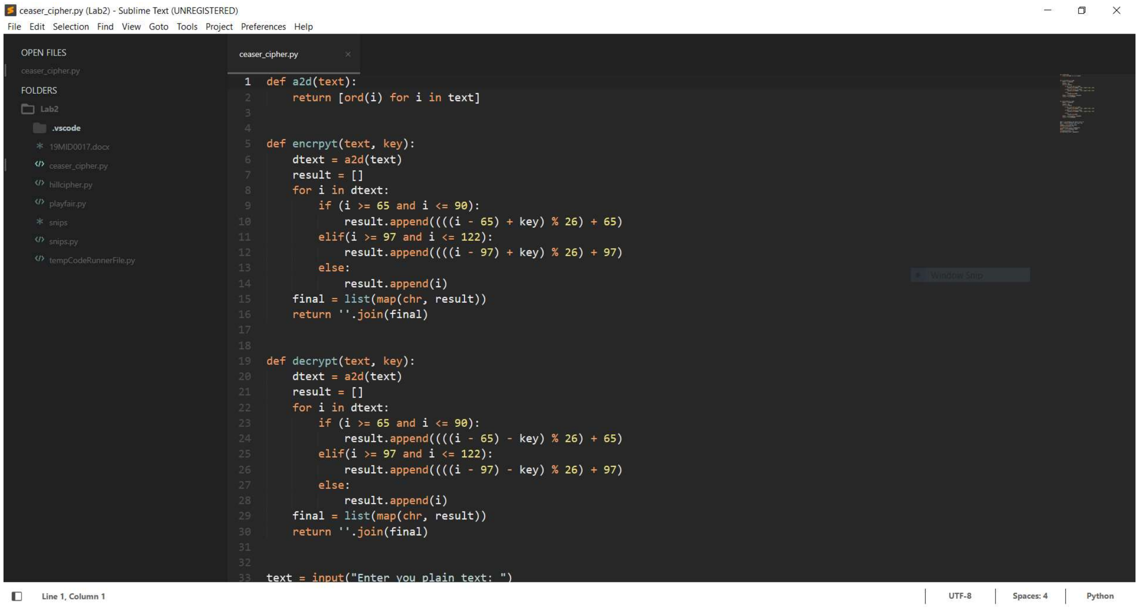
COURSE CODE : CSI3002

ASSESSMENT NO : 01

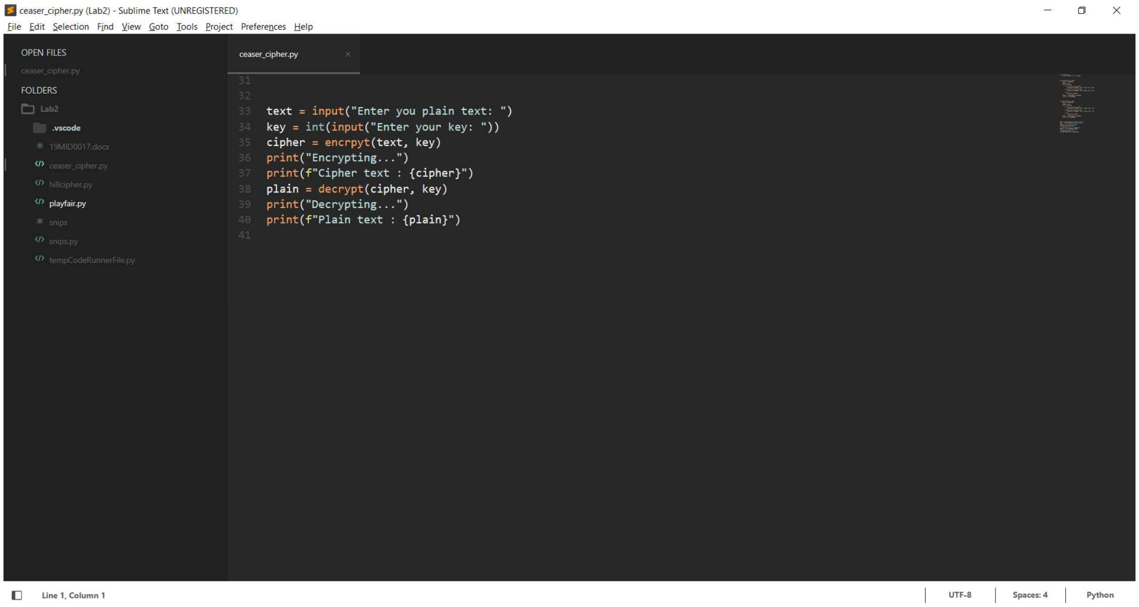
FACULTY : DR. A. MARY MEKALA

CIPHER ALGORITHMS

1) CEASER CIPHER:

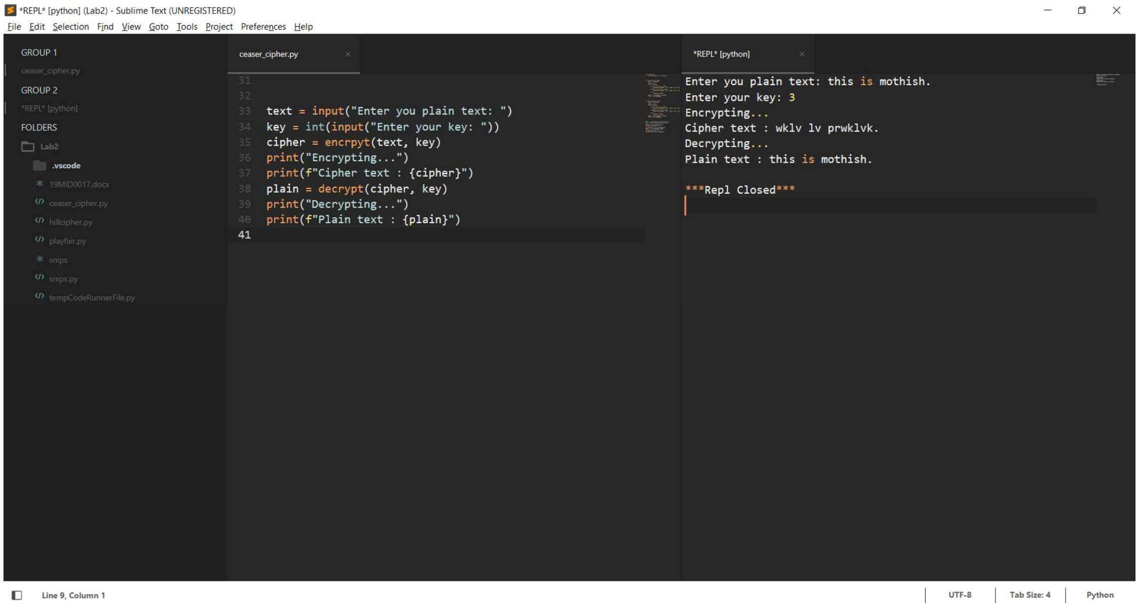


```python
def a2d(text):
    return [ord(i) for i in text]


def encrpyt(text, key):
    dtext = a2d(text)
    result = []
    for i in dtext:
        if (i >= 65 and i <= 90):
            result.append((((i - 65) + key) % 26) + 65)
        elif(i >= 97 and i <= 122):
            result.append((((i - 97) + key) % 26) + 97)
        else:
            result.append(i)
    final = list(map(chr, result))
    return ''.join(final)


def decrypt(text, key):
    dtext = a2d(text)
    result = []
    for i in dtext:
        if (i >= 65 and i <= 90):
            result.append((((i - 65) - key) % 26) + 65)
        elif(i >= 97 and i <= 122):
            result.append((((i - 97) - key) % 26) + 97)
        else:
            result.append(i)
    final = list(map(chr, result))
    return ''.join(final)


text = input("Enter you plain text: ")
```



```python


text = input("Enter you plain text: ")
key = int(input("Enter your key: "))
cipher = encrpyt(text, key)
print("Encrypting...")
print(f"Cipher text : {cipher}")
plain = decrypt(cipher, key)
print("Decrypting...")
print(f"Plain text : {plain}")

```

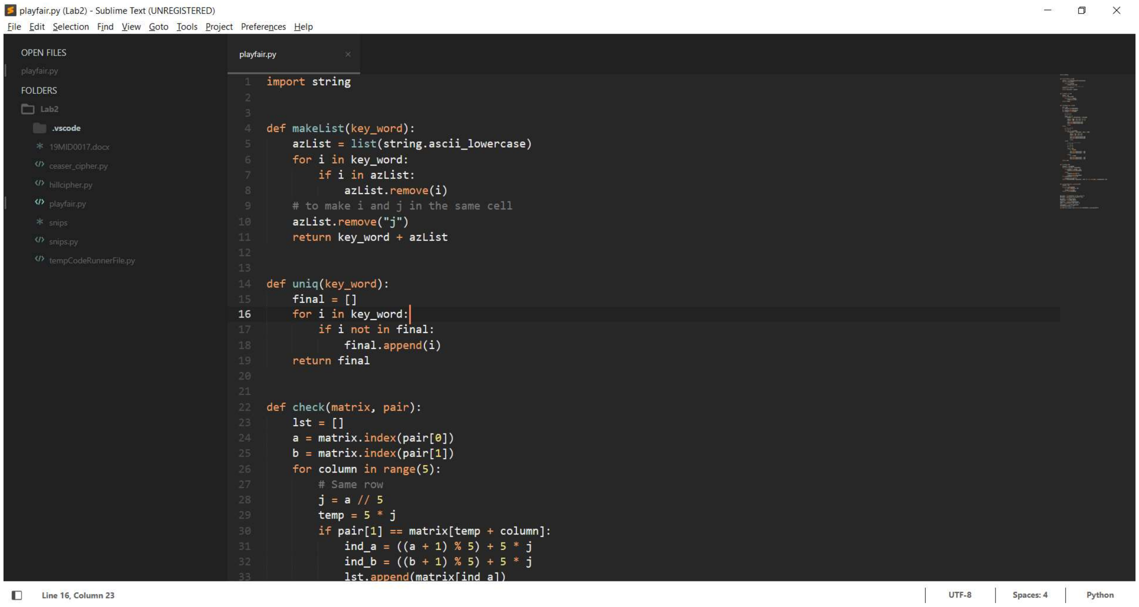
Output:



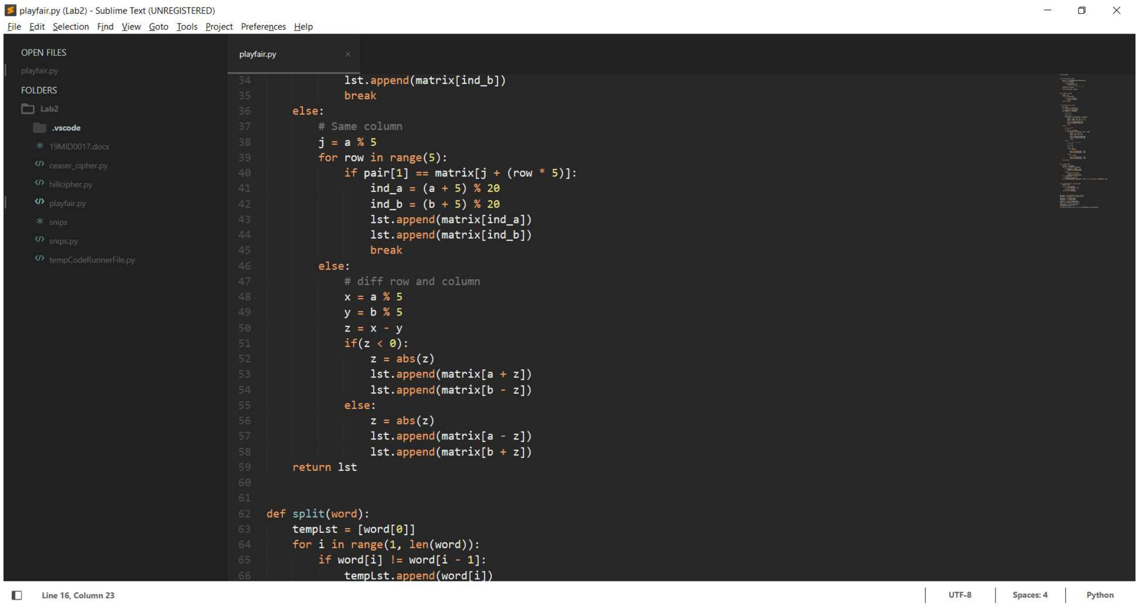
```python


text = input("Enter you plain text: ")
key = int(input("Enter your key: "))
cipher = encrpyt(text, key)
print("Encrypting...")
print(f"Cipher text : {cipher}")
plain = decrypt(cipher, key)
print("Decrypting...")
print(f"Plain text : {plain}")

```

```
Enter you plain text: this is mothish.
Enter your key: 3
Encrypting...
Cipher text : wklv lv prwklvk.
Decrypting...
Plain text : this is mothish.

***Repl Closed***
```

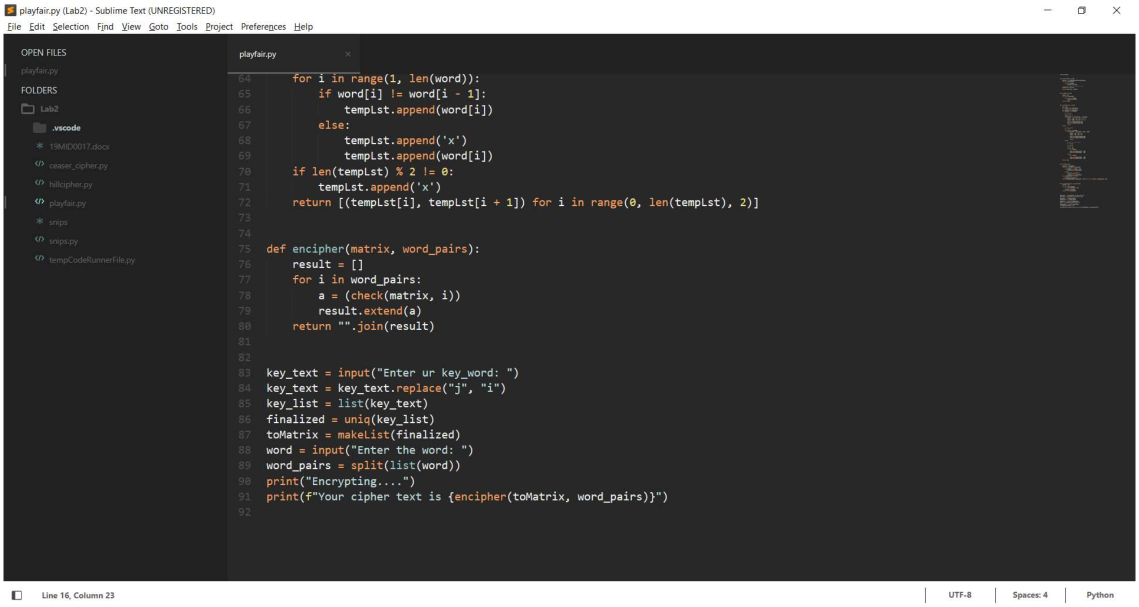
2) PLAYFAIR CIPHER:



```python
import string


def makeList(key_word):
    azList = list(string.ascii_lowercase)
    for i in key_word:
        if i in azList:
            azList.remove(i)
    # to make i and j in the same cell
    azList.remove("j")
    return key_word + azList


def uniq(key_word):
    final = []
    for i in key_word:
        if i not in final:
            final.append(i)
    return final


def check(matrix, pair):
    lst = []
    a = matrix.index(pair[0])
    b = matrix.index(pair[1])
    for column in range(5):
        # Same row
        j = a // 5
        temp = 5 * j
        if pair[1] == matrix[temp + column]:
            ind_a = ((a + 1) % 5) + 5 * j
            ind_b = ((b + 1) % 5) + 5 * j
            lst.append(matrix[ind_a])
```

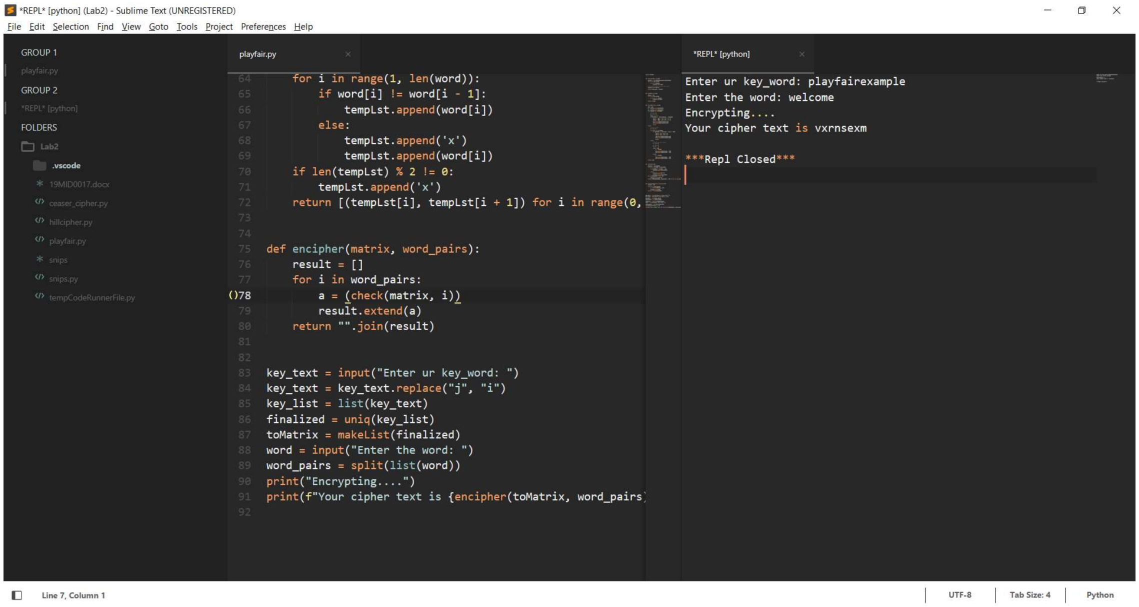


```python
                lst.append(matrix[ind_b])
                break
        else:
            # Same column
            j = a % 5
            for row in range(5):
                if pair[1] == matrix[j + (row * 5)]:
                    ind_a = (a + 5) % 20
                    ind_b = (b + 5) % 20
                    lst.append(matrix[ind_a])
                    lst.append(matrix[ind_b])
                    break
            else:
                # diff row and column
                x = a % 5
                y = b % 5
                z = x - y
                if(z < 0):
                    z = abs(z)
                    lst.append(matrix[a + z])
                    lst.append(matrix[b - z])
                else:
                    z = abs(z)
                    lst.append(matrix[a - z])
                    lst.append(matrix[b + z])
    return lst


def split(word):
    tempLst = [word[0]]
    for i in range(1, len(word)):
        if word[i] != word[i - 1]:
            tempLst.append(word[i])
```



```python
    for i in range(1, len(word)):
        if word[i] != word[i - 1]:
            tempLst.append(word[i])
        else:
            tempLst.append('x')
            tempLst.append(word[i])
    if len(tempLst) % 2 != 0:
        tempLst.append('x')
    return [(tempLst[i], tempLst[i + 1]) for i in range(0, len(tempLst), 2)]


def encipher(matrix, word_pairs):
    result = []
    for i in word_pairs:
        a = (check(matrix, i))
        result.extend(a)
    return "".join(result)


key_text = input("Enter ur key_word: ")
key_text = key_text.replace("j", "i")
key_list = list(key_text)
finalized = uniq(key_list)
toMatrix = makeList(finalized)
word = input("Enter the word: ")
word_pairs = split(list(word))
print("Encrypting....")
print(f"Your cipher text is {encipher(toMatrix, word_pairs)}")
```

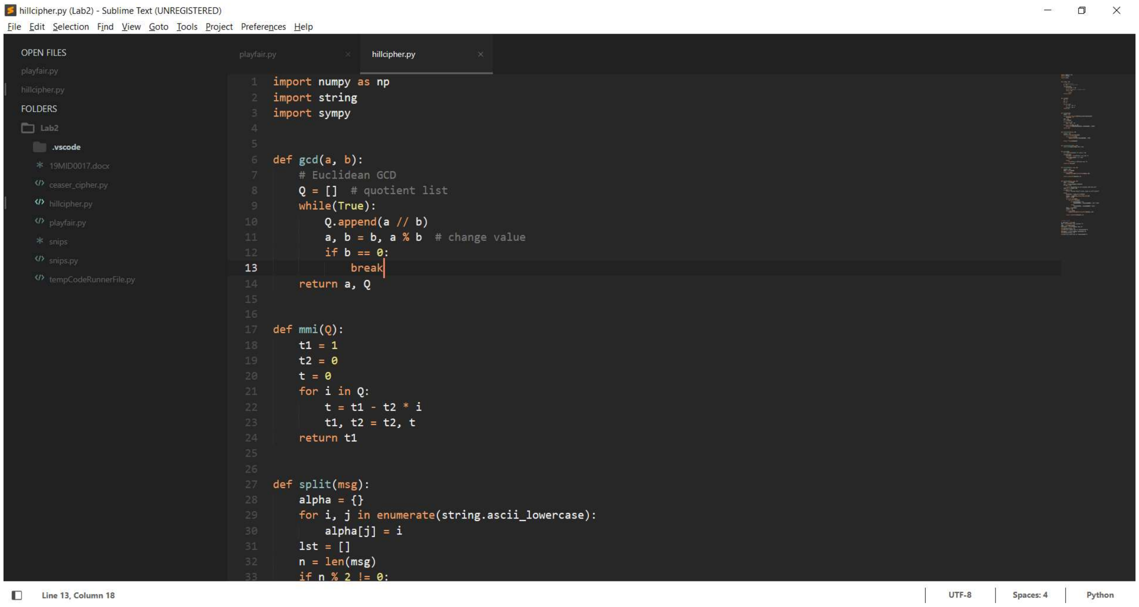
Output:



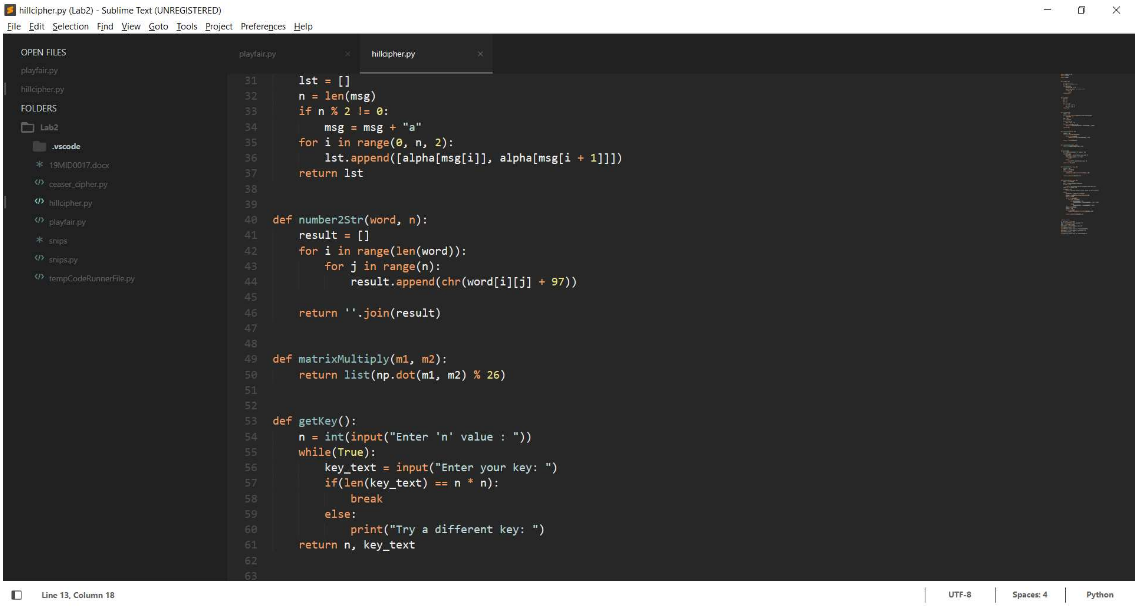
```python
    for i in range(1, len(word)):
        if word[i] != word[i - 1]:
            tempLst.append(word[i])
        else:
            tempLst.append('x')
            tempLst.append(word[i])
    if len(tempLst) % 2 != 0:
        tempLst.append('x')
    return [(tempLst[i], tempLst[i + 1]) for i in range(0,


def encipher(matrix, word_pairs):
    result = []
    for i in word_pairs:
        a = (check(matrix, i))
        result.extend(a)
    return "".join(result)


key_text = input("Enter ur key_word: ")
key_text = key_text.replace("j", "i")
key_list = list(key_text)
finalized = uniq(key_list)
toMatrix = makeList(finalized)
word = input("Enter the word: ")
word_pairs = split(list(word))
print("Encrypting....")
print(f"Your cipher text is {encipher(toMatrix, word_pairs
```

```
Enter ur key_word: playfairexample
Enter the word: welcome
Encrypting....
Your cipher text is vxrnsexm

***Repl Closed***
```

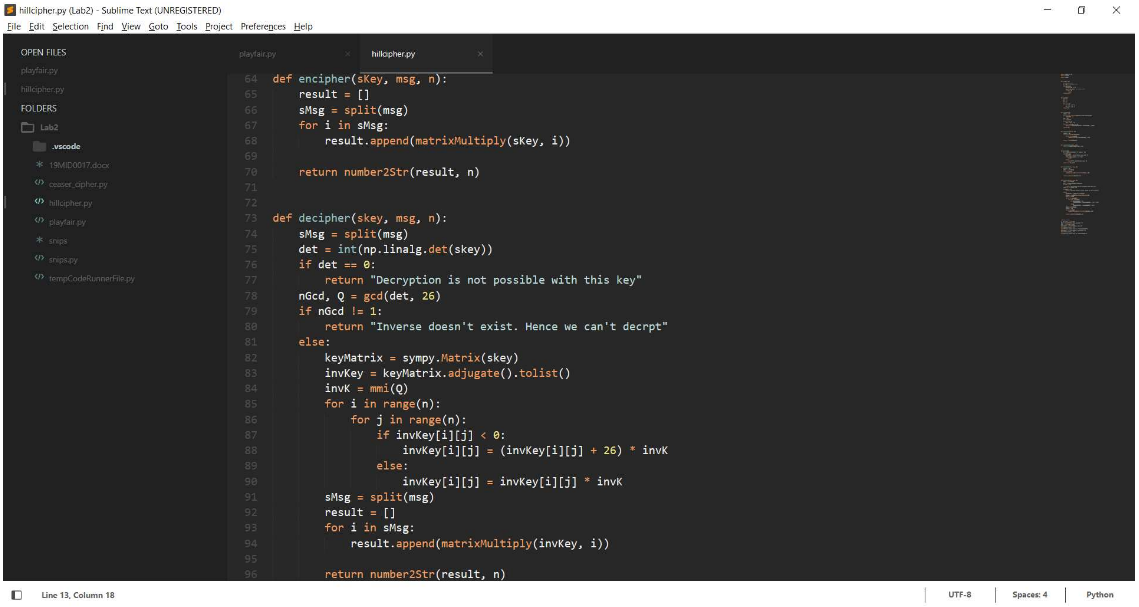
3) HILL CIPHER:



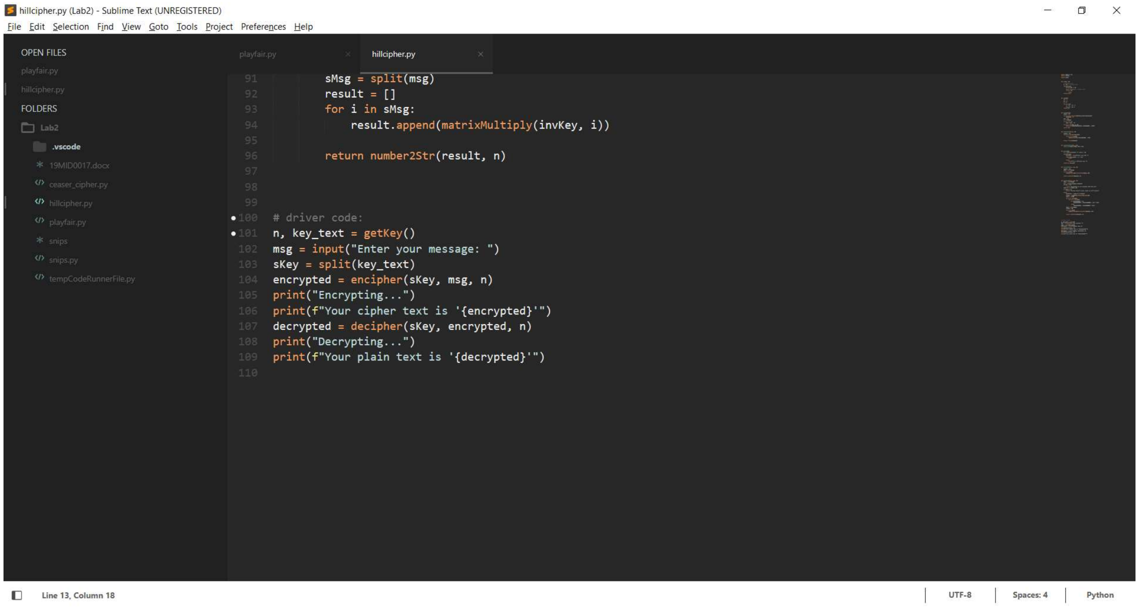
```python
import numpy as np
import string
import sympy


def gcd(a, b):
    # Euclidean GCD
    Q = []  # quotient list
    while(True):
        Q.append(a // b)
        a, b = b, a % b  # change value
        if b == 0:
            break
    return a, Q


def mmi(Q):
    t1 = 1
    t2 = 0
    t = 0
    for i in Q:
        t = t1 - t2 * i
        t1, t2 = t2, t
    return t1


def split(msg):
    alpha = {}
    for i, j in enumerate(string.ascii_lowercase):
        alpha[j] = i
    lst = []
    n = len(msg)
    if n % 2 != 0:
```



```python
    lst = []
    n = len(msg)
    if n % 2 != 0:
        msg = msg + "a"
    for i in range(0, n, 2):
        lst.append([alpha[msg[i]], alpha[msg[i + 1]]])
    return lst


def number2Str(word, n):
    result = []
    for i in range(len(word)):
        for j in range(n):
            result.append(chr(word[i][j] + 97))

    return ''.join(result)


def matrixMultiply(m1, m2):
    return list(np.dot(m1, m2) % 26)


def getKey():
    n = int(input("Enter 'n' value : "))
    while(True):
        key_text = input("Enter your key: ")
        if(len(key_text) == n * n):
            break
        else:
            print("Try a different key: ")
    return n, key_text

```

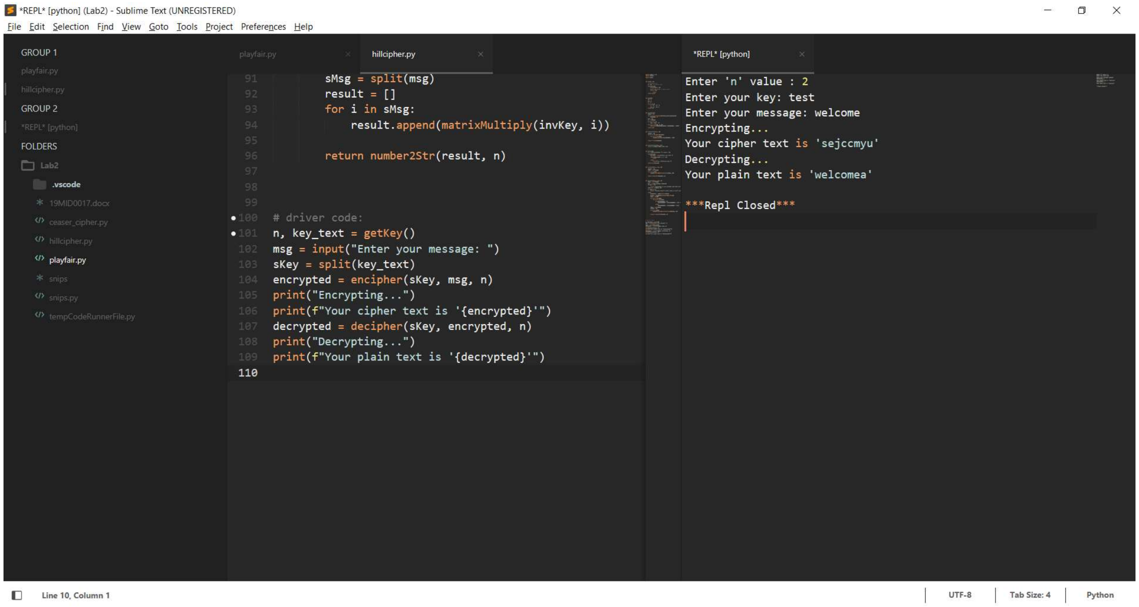


```python
def encipher(sKey, msg, n):
    result = []
    sMsg = split(msg)
    for i in sMsg:
        result.append(matrixMultiply(sKey, i))

    return number2Str(result, n)


def decipher(skey, msg, n):
    sMsg = split(msg)
    det = int(np.linalg.det(skey))
    if det == 0:
        return "Decryption is not possible with this key"
    nGcd, Q = gcd(det, 26)
    if nGcd != 1:
        return "Inverse doesn't exist. Hence we can't decrpt"
    else:
        keyMatrix = sympy.Matrix(skey)
        invKey = keyMatrix.adjugate().tolist()
        invK = mmi(Q)
        for i in range(n):
            for j in range(n):
                if invKey[i][j] < 0:
                    invKey[i][j] = (invKey[i][j] + 26) * invK
                else:
                    invKey[i][j] = invKey[i][j] * invK
        sMsg = split(msg)
        result = []
        for i in sMsg:
            result.append(matrixMultiply(invKey, i))

        return number2Str(result, n)
```



```python
        sMsg = split(msg)
        result = []
        for i in sMsg:
            result.append(matrixMultiply(invKey, i))

        return number2Str(result, n)



# driver code:
n, key_text = getKey()
msg = input("Enter your message: ")
sKey = split(key_text)
encrypted = encipher(sKey, msg, n)
print("Encrypting...")
print(f"Your cipher text is '{encrypted}'")
decrypted = decipher(sKey, encrypted, n)
print("Decrypting...")
print(f"Your plain text is '{decrypted}'")

```

Output:



```python
        sMsg = split(msg)
        result = []
        for i in sMsg:
            result.append(matrixMultiply(invKey, i))

        return number2Str(result, n)



# driver code:
n, key_text = getKey()
msg = input("Enter your message: ")
sKey = split(key_text)
encrypted = encipher(sKey, msg, n)
print("Encrypting...")
print(f"Your cipher text is '{encrypted}'")
decrypted = decipher(sKey, encrypted, n)
print("Decrypting...")
print(f"Your plain text is '{decrypted}'")
```

```
Enter 'n' value : 2
Enter your key: test
Enter your message: welcome
Encrypting...
Your cipher text is 'sejccmyu'
Decrypting...
Your plain text is 'welcomea'

***Repl Closed***
```