Practical – 3

Aim : Explain Encryption & Decryption and implement different types of ciphers.

Sol.

Encryption : Encryption is a way of scrambling data so that only authorized parties can understand the information. In technical terms, it is the process of converting plaintext to ciphertext. In simpler terms, encryption takes readable data and alters it so that it appears random.

Decryption : The conversion of encrypted data into its original form is called Decryption. It is generally a reverse process of encryption. It decodes the encrypted information so that an authorized user can only decrypt the data because decryption requires a secret key or password.

- ADFGVX Cipher :
  Code :

```
from pycipher import ADFGVX

def func():
        adfgvx = ADFGVX(key='PH0QG64MEA1YL2NOFDXKR3CVS5ZW7BJ9UTI8',
keyword='GERMAN')
        enc = adfgvx.encipher("Hello world!")
        dec = adfgvx.decipher(enc)
        print (dec)

func()
```

```
J:\Desktop>py cipher.py
HELLOWORLD
```

- Shift Cipher :
  Code :

```
def encrypt(text,s):
result = ""
for i in range(len(text)):
    char = text[i]
    if (char.isupper()):
```

```
            result += chr((ord(char) + s-65) % 26 + 65)
        else:
            result += chr((ord(char) + s - 97) % 26 + 97)
return result

text = "CEASER CIPHER DEMO"
s = 4
print ("Plain Text : " + text)
print ("Cipher: " + encrypt(text,s))
```

```
J:\Desktop>py cipher.py
Plain Text : CEASER CIPHER DEMO
Cipher: GIEWIVrGMTLIVrHIQS
```

- Substitution Cipher :

  Code :

```
import string

all_letters= string.ascii_letters

dict1 = {}
key = 4

for i in range(len(all_letters)):
    dict1[all_letters[i]] = all_letters[(i+key)%len(all_letters)]

plain_txt= "I am studying Data Encryption"
cipher_txt=[]

for char in plain_txt:
    if char in all_letters:
        temp = dict1[char]
        cipher_txt.append(temp)
    else:
        temp =char
```

```
        cipher_txt.append(temp)

cipher_txt= "".join(cipher_txt)
print("Cipher Text is: ",cipher_txt)



dict2 = {}
for i in range(len(all_letters)):
    dict2[all_letters[i]] = all_letters[(i-key)%(len(all_letters))]

decrypt_txt = []

for char in cipher_txt:
    if char in all_letters:
        temp = dict2[char]
        decrypt_txt.append(temp)
    else:
        temp = char
        decrypt_txt.append(temp)

decrypt_txt = "".join(decrypt_txt)
print("Recovered plain text :", decrypt_txt)
```

```
J:\Desktop>py cipher.py
Cipher Text is:  M eq wxyhCmrk Hexe IrgvCtxmsr
Recovered plain text : I am studying Data Encryption
```

- Vigenere Cipher :

  Code :

  ```
  from pycipher import Vigenere

  def encrypt():
      text = "Hello world!"
      key = "keykey"
      enc_text = Vigenere(key).encipher(text)
      print (enc_text)
  ```

```
    dec_text = Vigenere(key).decipher(enc_text)
    print (dec_text)

encrypt()
```

```
J:\Desktop>py cipher.py
RIJVSUYVJN
HELLOWORLD
```

- *Affine Cipher :*

  Code :

```
from pycipher import Affine

def encrypt():
    text = "Hello world!"
    enc_text = Affine(a=5,b=9).encipher(text)
    dec_text = Affine(a=5,b=9).decipher(enc_text)

    print (enc_text)
    print (dec_text)

encrypt()
```

```
J:\Desktop>py cipher.py
SDMMBPBQMY
HELLOWORLD
```

- Permutation cipher :

  Code

```
def split_len(seq, length):
    return [seq[i:i + length] for i in range(0, len(seq), length)]

def encode(key, plaintext):
    order = {
    int(val): num for num, val in enumerate(key)
    }
```

```
    ciphertext = ''

    for index in sorted(order.keys()):
    for part in split_len(plaintext, len(key)):
            try:ciphertext += part[order[index]]
            except IndexError:
                        continue
    return ciphertext

print(encode('3214', 'HELLO World'))
```

```
J:\Desktop>py cipher.py
LWdE lHOrLo
```

- *Playfair cipher :*

  Code :

```
from pycipher import Playfair

def encrypt():
    text = "Hello world"
    key = "zgptfoihmuwdrcnykeqaxvsbl"
    enc_text = Playfair(key).encipher(text)
    dec_text = Playfair(key).decipher(enc_text)

    print (enc_text)
    print (dec_text)

encrypt()
```

```
J:\Desktop>py cipher.py
RSXVWYHWVN
HELXOWORLD
```

- *Enigma Cipher :*

  Code :

```
from pycipher import Enigma

def encrypt():
    text = "Hello world!"
    enc_text =
Enigma(settings=('A','A','A'),rotors=(1,2,3),reflector='B',ringstellung=('F','V','N'),stecke
rs=[('P','O'),('M','L'),('I','U'),('K','J'),('N','H'),('Y','T'),('G','B'),('V','F'),('R','E'),('D','C')]).enci
pher(text)

   dec_text =
Enigma(settings=('A','A','A'),rotors=(1,2,3),reflector='B',ringstellung=('F','V','N'),stecke
rs=[('P','O'),('M','L'),('I','U'),('K','J'),('N','H'),('Y','T'),('G','B'),('V','F'),('R','E'),('D','C')]).deci
pher(enc_text)
   print (enc_text)
   print (dec_text)

encrypt()
```

```
J:\Desktop>py cipher.py
VQQEFFGBHU
HELLOWORLD
```