NAME - HARSH BALYAN   Date - 15/06/21

Course - BCA (6th)   Subject - information Security

Section - A

Uni Roll No - 1121056   Harsh

Roll No - 51

Question-3

```
import random as r.
def otpgen ():
otp = " "
for i in range(4):
otp + = str(r. randint (1,9))
print (" Your One Time Password is ")
print(otp)
```

Question -3

```
# Vigenere Cipher

def generateKey(string, key):
    key = list(key)
    if len(string) == len(key):
        return(key)
    else:
        for i in range(len(string) - len(key)):

            key.append(key[i % len(key)])

    return ("".join(key))
# Encryption
def CipherText(string, key):

    Cipher_Text = []

    for i in range(len(string)):

        x = (ord(string[i]) + ord(key[i]))%26

        x+ = ord('A')
        Cipher_text.append(chr(x))
    return ("". join(cipher_text))

# Function for decrypting
def originalText(cipher_Text, key):
    orig_text = []
    for i in range(len(cipher_Text)):
        x = (ord(cipher_text[i]) - ord(key[i]) + 26)
                                                    %26
        x += ord('A')
        orig_text.append(chr(x))
    return ("". join(orig_text))

# Driver Code
if _name_ == "_main_":
    string = "Cryptography"
    keyword = "Monarchy"

    key = generateKey(string, keyword)
    print ("CipherText :", cipher_text)
    print ("Original/DecryptedText :", originalText(
                                       cipher_text, key))
```

Name - HARSH BALYAN Date - 15/06/2021

Course - BCA (6th sem)

ROLL NO - 1121056

Subject - Information Security and Cyber laws.

Question - 5

Implementation of Encryption and Decryption using Caeser Cipher

```
def encryption (plain-text, key):
    encrypted = " "
    for c in plain-text:
        if c.isupper():
            c-index = ord(c)-ord('A')
            c-shifted = (c-index + key)%26+ord('A')
            c-new = chr(c-shifted)
            encrypted+ = c-new
    elif c.islower():
            c-index = ord(c)_ord('a')
            c-shifted = (c-index + key)%26+ord('a')
            c-new = chr(c-shifted)
            encrypted + = c-new
```

Narsp

```
    elif c.isdigit():
        c-new=(int(c)+key)%10
        encrypted+ = str(c-new)
    else :
        encrypted + = c
return encrypted
# Decryption
def cipher-decrypt(ciphertext, key):
    decrypted = ""
    for c in ciphertext :
        if c.isupper():
            c-index= ord(c)-ord('A')
            c-og-pos=(c-index-key)%26+ord('A')
            c-og = chr(c-og-pos)
            decrypted + = c-og
        elif c.islower():
            c-index =ord(c)-ord('a')
            c-og-pos=(c-index-key)%26+ord('a')
            c-og = chr(c-og-pos)
            decrypted+ = c-og
        elif c.isdigit ():
            c-og = (int (c) - key ) % 10
            decrypted + = str (c-og)
        else :
            decrypted + = c
    return decrypted
plain-text = "Attack from North"
Ciphertext = encryption (plain-text, 4)
print(" plain text ", plaintext)
print (" encrypted Ciphertext : \n", CipherText)
decryptedmsg = decryption( ciphertext, 4)
print(" The decrypted message is : \n", decryptedmsg)
```