NAME - HARSH BALYAN

Course - BCA

Uni Roll No - 1121056

Date - 15/06/2021

Subject - Information Security

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 - 15/06/21

Course - BCA (6th) Subject - Information Security

Section - A

Uni Roll No - 1121056

Roll No - 51

Harsh

Question - 4

```
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)
```

Name - HARSH BALYAN Date - 15/06/2021

Course - BCA (6th Sem)

ROLL No - 1121056 Harsh

Subject - Information Security and Cyber Laws.

Question -5

#Implementation of Encryption and Decryption using Caesar 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
```

Naresh

```
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
```