PART A

1. Caser cipher

```
def CaeserCipher(text,key):
    cipherText=""
    for ch in text:
        if(ch.isupper()):
            cipherText+=chr((ord(ch)-65+key)%26+65)
        elif(ch.islower()):
            cipherText+=chr((ord(ch)-97+key)%26+97)
        else:
            cipherText+=" "
    return cipherText

s=input("Enter string: ")
key=int(input("Enter key: "))
ctext=CaeserCipher(s,key)
print("Encrpted Text is: "+ctext)
key=key*-1
dtext=CaeserCipher(ctext,key)
print("Decrpted Text is: "+dtext)
```

2. Monoalphabetic cipher

```
UL=['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q',
'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
MU=['G', 'X', 'T', 'Z', 'A', 'K', 'F', 'H', 'I', 'W', 'L', 'Q', 'P', 'D', 'R', 'Y', 'U',
'O', 'C', 'E', 'V', 'S', 'B', 'N', 'J', 'M']

s=input("Enter the String: ")
cipherText=""
for ch in s:
    index=UL.index(ch)
    cipherText+=MU[index]
print("Encypted msg is: "+cipherText)
msg=""
for ch in cipherText:
    index=MU.index(ch)
    msg+=UL[index]
print("Decypted msg is: "+msg)
```

3. Diffie Hellman

```
#q=71 a=7 xa=12 xb=5

q = int(input("Enter a prime number : "))
a = int(input("Enter a primitive root :"))

Xa = int(input("Enter the private key of A :"))
Xb = int(input("Enter the private key of B : "))
```

```
Ya = pow(a, Xa) % q
Yb = pow(a, Xb) % q

print("Public key of  A  : ",Ya)
print("Public key of B : ",Yb)

Ka = pow(Yb,Xa)%q
Kb = pow(Ya,Xb)%q

print("Shared key for A : ",Ka)
print("Shared key for B : ",Kb)
```

4. ECC

```
import tinyec
from tinyec import registry
import secrets

curve = registry.get_curve('brainpoolP256r1')

def compress_point(point):
    return hex(point.x) + hex(point.y % 2)[2:]

def getEnKey(pubKey):
    ciPrivKey = secrets.randbelow(curve.field.n)
    ciPubKey = ciPrivKey * curve.g
    enKey = ciPrivKey * pubKey
    return (enKey, ciPubKey)

senderPriKey = secrets.randbelow(curve.field.n)
senderPubKey = senderPriKey * curve.g

print("Sender's Private Key: " + hex(senderPriKey))
print("Sender's Public Key: " + compress_point(senderPubKey))

(enKeySender, ciPubKeySender) = getEnKey(senderPubKey)

print("\nSender's Ciphertext Public Key: " + compress_point(ciPubKeySender))
print("Sender's Encryption Key: " + compress_point(enKeySender))

receiverPriKey = secrets.randbelow(curve.field.n)
receiverPubKey = receiverPriKey * curve.g

print("\nReceiver's Private Key: " + hex(receiverPriKey))
print("Receiver's Public Key: " + compress_point(receiverPubKey))

(enKeyReceiver, ciPubKeyReceiver) = getEnKey(receiverPubKey)

print("\nReceiver's Ciphertext Public Key: " + compress_point(ciPubKeyReceiver))
print("Receiver's Encryption Key: " + compress_point(enKeyReceiver))
```

5. Vigenère Cipher

```
s=input("Enter String: ") #uppercase
key=input("Enter key: ")

i=0
while(len(s)!=len(key)):
    key+=key[i]
    i+=1
#print(key)

i=0
cipherText=""
for ch in s:
    cipherText+=chr(((ord(ch)-65)+(ord(key[i])-65))%26+65)
    i+=1
print(cipherText)
i=0
msg=""
for ch in cipherText:
    msg+=chr(((ord(ch)-65)-(ord(key[i])-65))%26+65)
    i+=1
print(msg)
```

All the Best 😊
-Nelson D Cunha