Cryptography Fundamentals

LAB DIGITAL ASSIGNMENT -1

NAME: ANIRUDH KUMAR

REG NO: 19BCI0246

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

Caesar cipher:

Aim: To implement Caesar cipher to encrypt and decrypt a given text.

Procedure:

For encryption:

- Traverse the given text one character at a time.
- For each character ch: ch=(ch+3)%26.
- Return the new string generated.

For decryption:

- Traverse the given text one character at a time.
- For each character ch: ch=(ch-3)%26.
- Return the new string generated.

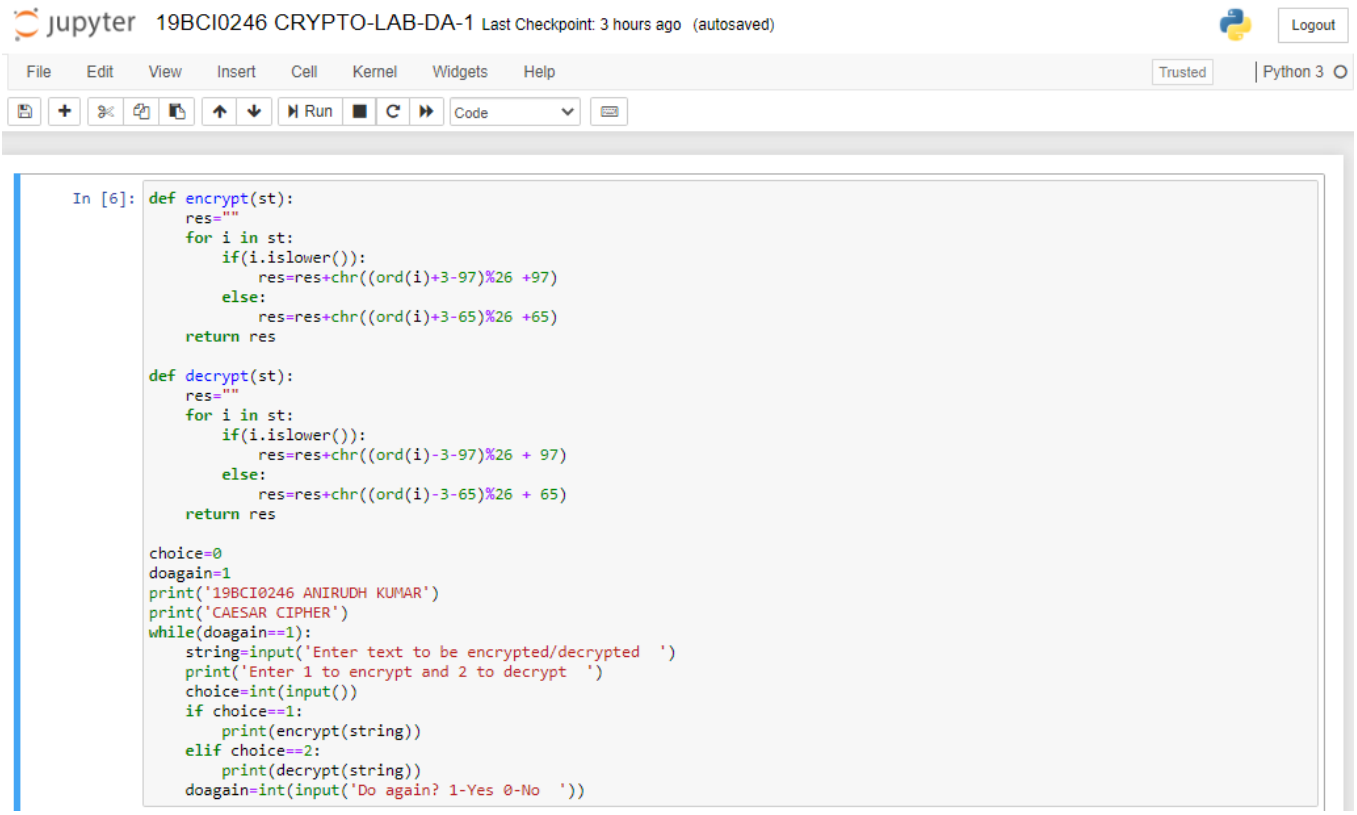
Code:

```
def encrypt(st):
    res=""
    for i in st:
        if(i.islower()):
            res=res+chr((ord(i)+3-97)%26 +97)
        else:
            res=res+chr((ord(i)+3-65)%26 +65)
    return res


def decrypt(st):
    res=""
    for i in st:
        if(i.islower()):
            res=res+chr((ord(i)-3-97)%26 + 97)
        else:
            res=res+chr((ord(i)-3-65)%26 + 65)
    return res
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
choice=0

doagain=1

print('19BCI0246 ANIRUDH KUMAR')

print('CAESAR CIPHER')

while(doagain==1):

    string=input('Enter text to be encrypted/decrypted  ')

    print('Enter 1 to encrypt and 2 to decrypt  ')

    choice=int(input())

    if choice==1:

        print(encrypt(string))

    elif choice==2:

        print(decrypt(string))

    doagain=int(input('Do again? 1-Yes 0-No  '))
```

Screenshot of code:



NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

OUTPUT:

```
19BCI0246 ANIRUDH KUMAR
CAESAR CIPHER
Enter text to be encrypted/decrypted  Caesar
Enter 1 to encrypt and 2 to decrypt
1
Fdhvdu
Do again? 1-Yes 0-No  1
Enter text to be encrypted/decrypted  Fdhvdu
Enter 1 to encrypt and 2 to decrypt
2
Caesar
Do again? 1-Yes 0-No  0
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

Playfair cipher:

Aim: To implement Playfair cipher to encrypt and decrypt a given text.

Procedure:

For encryption:

- Generate the key matrix: All letters to be placed in the 5x5 matrix. First, the letters in the key. Then, the other letters in ascending order
- If the plaintext contains J, replace with I.
- To encrypt the plaintext: The text is split into pairs of 2 letters. If there is an odd number of letters, a dummy letter x is added to the last letter.
- Rules for encryption:
    i. If letters in the same row: take letter to left of each letter.
    ii. If letters in same column, take letter below each letter.
    iii. Else: Form a rectangle with the 2 letters and take the letters on the horizontal opposite corner of the rectangle.
- Print the encrypted string.

For decryption:

- Generate the key matrix at the receiver's end: All letters to be placed in the 5x5 matrix. First, the letters in the key. Then, the other letters in ascending order. J is replaced by I.
- To decrypt the ciphertext: The text is split into pairs of 2 letters.
- Rules for decryption:
    i. If letters in same row: take letter to right of each letter.
    ii. If letters in same column: take letter above each letter.
    iii. Else: Form a rectangle with the two letters and take the letters on horizontal opposite corner of the rectangle.
- Print the decrypted string.

Code:

```
def matrix(x,y,initial):

    return [[initial for i in range(x)] for j in range(y)]


result=list()

for c in key:

    if c not in result:

        if c=='J':

            result.append('I')

        else:

            result.append(c)
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
flag=0

for i in range(65,91):

    if chr(i) not in result:

        if i==73 and chr(74) not in result:

            result.append("I")

            flag=1

        elif flag==0 and i==73 or i==74:

            pass

        else:

            result.append(chr(i))

k=0

my_matrix=matrix(5,5,0)

for i in range(0,5):

    for j in range(0,5):

        my_matrix[i][j]=result[k]

        k+=1


def locindex(c):

    loc=list()

    if c=='J':

        c='I'

    for i ,j in enumerate(my_matrix):

        for k,l in enumerate(j):

            if c==l:

                loc.append(i)

                loc.append(k)

                return loc


def encrypt():

    msg=str(input("ENTER MSG: "))

    msg=msg.upper()
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
    msg=msg.replace(" ", "")
    i=0
    for s in range(0,len(msg)+1,2):
        if s<len(msg)-1:
            if msg[s]==msg[s+1]:
                msg=msg[:s+1]+'X'+msg[s+1:]
    if len(msg)%2!=0:
        msg=msg[:]+'X'
    print("CIPHER TEXT: ",end=' ')
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:
            print("{}{}".format(my_matrix[(loc[0]+1)%5][loc[1]],my_matrix[(loc1[0]+1)%5][loc1[1]]),end='
')
        elif loc[0]==loc1[0]:
            print("{}{}".format(my_matrix[loc[0]][(loc[1]+1)%5],my_matrix[loc1[0]][(loc1[1]+1)%5]),end='
')
        else:
            print("{}{}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]]),end=' ')
        i=i+2
    print(" \n ")


def decrypt():
    msg=str(input("ENTER CIPHER TEXT: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    i=0
    res=''
    while i<len(msg):
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:
            res=res+my_matrix[(loc[0]-1)%5][loc[1]]+my_matrix[(loc1[0]-1)%5][loc1[1]]
        elif loc[0]==loc1[0]:
            res=res+my_matrix[loc[0]][(loc[1]-1)%5]+my_matrix[loc1[0]][(loc1[1]-1)%5]
        else:
            res=res+my_matrix[loc[0]][loc1[1]]+my_matrix[loc1[0]][loc[1]]
        i=i+2
    print(" \n ")
    print('Plaintext with X in it: \n',res.lower())
    res=list(res)
    for i in res:
        if i=='X':
            res.remove(i)
    result=''
    for i in res:
        result=result+i
    print('Plaintext without X in it: \n',result.lower())



print('19BCI0246 ANIRUDH KUMAR')
print('Play Fair Cipher')
key=input("Enter key: ")
key=key.replace(" ", "")
key=key.upper()
doagain=1
while(doagain==1):
    choice=int(input("Do you want to \n 1.Encrypt \n 2.Decrypt \n 3.EXIT \n"))
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
if choice==1:
    encrypt()
elif choice==2:
    decrypt()
elif choice==3:
    exit()
else:
    print("INVALID INPUT")
doagain=int(input('Do again?'))
```

Screenshot of code:

jupyter 19BCI0246 CRYPTO-LAB-DA-1 Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

In [9]:

```
def matrix(x,y,initial):
    return [[initial for i in range(x)] for j in range(y)]

result=list()
for c in key:
    if c not in result:
        if c=='J':
            result.append('I')
        else:
            result.append(c)
flag=0
for i in range(65,91):
    if chr(i) not in result:
        if i==73 and chr(74) not in result:
            result.append("I")
            flag=1
        elif flag==0 and i==73 or i==74:
            pass
        else:
            result.append(chr(i))
k=0
my_matrix=matrix(5,5,0)
for i in range(0,5):
    for j in range(0,5):
        my_matrix[i][j]=result[k]
        k+=1

def locindex(c):
    loc=list()
    if c=='J':
        c='I'
    for i ,j in enumerate(my_matrix):
        for k,l in enumerate(j):
            if c==l:
                loc.append(i)
                loc.append(k)
                return loc

def encrypt():
    msg=str(input("ENTER MSG: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    i=0
    for s in range(0,len(msg)+1,2):
        if s<len(msg)-1:
            if msg[s]==msg[s+1]:
                msg=msg[:s+1]+'X'+msg[s+1:]
    if len(msg)%2!=0:
        msg=msg[:]+'X'
    print("CIPHER TEXT: ",end=' ')
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:
            print("{}{}".format(my_matrix[(loc[0]+1)%5][loc[1]],my_matrix[(loc1[0]+1)%5][loc1[1]]),end=' ')
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
        if loc[1]==loc1[1]:
            print("{}{}".format(my_matrix[(loc[0]+1)%5][loc[1]],my_matrix[(loc1[0]+1)%5][loc1[1]]),end=' ')
        elif loc[0]==loc1[0]:
            print("{}{}".format(my_matrix[loc[0]][(loc[1]+1)%5],my_matrix[loc1[0]][(loc1[1]+1)%5]),end=' ')
        else:
            print("{}{}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]]),end=' ')
        i=i+2
    print(" \n ")

def decrypt():
    msg=str(input("ENTER CIPHER TEXT: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    i=0
    res=''
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:
            res=res+my_matrix[(loc[0]-1)%5][loc[1]]+my_matrix[(loc1[0]-1)%5][loc1[1]]
        elif loc[0]==loc1[0]:
            res=res+my_matrix[loc[0]][(loc[1]-1)%5]+my_matrix[loc1[0]][(loc1[1]-1)%5]
        else:
            res=res+my_matrix[loc[0]][loc1[1]]+my_matrix[loc1[0]][loc[1]]
        i=i+2
    print(" \n ")
    print('Plaintext with X in it: \n',res.lower())
    res=list(res)
    for i in res:
        if i=='X':
            res.remove(i)
    result=''
    for i in res:
        result=result+i
    print('Plaintext without X in it: \n',result.lower())


print('19BCI0246 ANIRUDH KUMAR')
print('Play Fair Cipher')
key=input("Enter key: ")
key=key.replace(" ", "")
key=key.upper()
doagain=1
while(doagain==1):
    choice=int(input("Do you want to \n 1.Encrypt \n 2.Decrypt \n 3.EXIT \n"))
    if choice==1:
        encrypt()
    elif choice==2:
        decrypt()
    elif choice==3:
        exit()
    else:
        print("INVALID INPUT")
    doagain=int(input('Do again?'))
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

OUTPUT:

```
19BCI0246 ANIRUDH KUMAR
Play Fair Cipher
Enter key: playfair
Do you want to
 1.Encrypt
 2.Decrypt
 3.EXIT
1
ENTER MSG: meet me at the school house
CIPHER TEXT:  EG MN EG FQ QM KN BK SV VR GQ XN KU

Do again?1
Do you want to
 1.Encrypt
 2.Decrypt
 3.EXIT
2
ENTER CIPHER TEXT: EG MN EG FQ QM KN BK SV VR GQ XN KU


Plaintext with X in it:
 meetmeattheschoxolhousex
Plaintext without X in it:
 meetmeattheschoolhouse
Do again?0
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

Hill cipher

Aim: To implement Hill cipher to encrypt a given text.

Procedure:

For encryption:

- The key is converted into a matrix which contains the number which that alphabet corresponds to: a=0 ….. z=25.
- The input is split into smaller pieces of size=3.
- If the input's length is not divisible by 3, additional letters are added: x and y depending on the remainder.
- The key matrix is multiplied with the input list's smaller matrix (which has only 3 elements). The resultant is the encrypted version of that part.
- The above step is repeated until the entire message gets encrypted.

Code:

```
print('19BCI0246 ANIRUDH KUMAR')

print('Hill Cipher')

keyMatrix = [[0] * 3 for i in range(3)]

messageMatrix = [[0] for i in range(3)]

resultMatrix = [[0] for i in range(3)]

result=''

def getKeyMatrix(key):

    k = 0

    for i in range(3):

        for j in range(3):

            keyMatrix[i][j] = key[k]

            k += 1


def encrypt(messageMatrix):

    for i in range(3):

        for j in range(1):
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
        resultMatrix[i][j] = 0
        for x in range(3):
            resultMatrix[i][j] += (keyMatrix[i][x] * messageMatrix[x][j])
        resultMatrix[i][j] = resultMatrix[i][j] % 26


def HillCipher(message, key):
    global result
    getKeyMatrix(key)

    for i in range(3):
        messageMatrix[i][0] = ord(message[i]) % 97

    encrypt(messageMatrix)

    CipherText = []
    for i in range(3):
        CipherText.append(chr(resultMatrix[i][0] + 97))
    for i in CipherText:
        result=result+i

message=input('Enter a message to be encrypted: ')
message.lower()
l=[]
if(len(message)%3==1):
    message=message+'x'+'y'
elif(len(message)%3==2):
    message=message+'x'
else:
    message=message
print(message)
for i in range(0,len(message),3):
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
    l.append(message[i:i+3])
print(l)
key = list(input('Please enter the key matrix a=0 - z=25: \n').split())
if len(key)==9:
    for i in range(len(key)):
        key[i]=int(key[i])
    print(key)
    for m in l:
        HillCipher(m, key)
    print('Resultant encrypted message:')
    print(result.upper())
else:
    print('Please enter key of valid length')
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

Screenshot of code:

jupyter 19BCI0246 CRYPTO-LAB-DA-1 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [10]:
```python
print('19BCI0246 ANIRUDH KUMAR')
print('Hill Cipher')
keyMatrix = [[0] * 3 for i in range(3)]
messageMatrix = [[0] for i in range(3)]
resultMatrix = [[0] for i in range(3)]
result=''
def getKeyMatrix(key):
    k = 0
    for i in range(3):
        for j in range(3):
            keyMatrix[i][j] = key[k]
            k += 1

def encrypt(messageMatrix):
    for i in range(3):
        for j in range(1):
            resultMatrix[i][j] = 0
            for x in range(3):
                resultMatrix[i][j] += (keyMatrix[i][x] * messageMatrix[x][j])
            resultMatrix[i][j] = resultMatrix[i][j] % 26

def HillCipher(message, key):
    global result
    getKeyMatrix(key)

    for i in range(3):
        messageMatrix[i][0] = ord(message[i]) % 97

    encrypt(messageMatrix)

    CipherText = []
    for i in range(3):
        CipherText.append(chr(resultMatrix[i][0] + 97))
    for i in CipherText:
        result=result+i

message=input('Enter a message to be encrypted: ')
message.lower()
l=[]
if(len(message)%3==1):
    message=message+'x'+'y'
elif(len(message)%3==2):
    message=message+'x'
else:
    message=message
print(message)
for i in range(0,len(message),3):
    l.append(message[i:i+3])
print(l)
key = list(input('Please enter the key matrix a=0 - z=25: \n').split())
if len(key)==9:
    for i in range(len(key)):
        key[i]=int(key[i])
    print(key)
    for m in l:
        HillCipher(m, key)
    print('Resultant encrypted message:')
```

```python
    print('Resultant encrypted message:')
    print(result.upper())
else:
    print('Please enter key of valid length')
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

OUTPUT:

```
19BCI0246 ANIRUDH KUMAR
Hill Cipher
Enter a message to be encrypted: blockcipher
blockcipherx
['blo', 'ckc', 'iph', 'erx']
Please enter the key matrix a=0 - z=25:
1 2 3 4 5 6 11 9 8
[1, 2, 3, 4, 5, 6, 11, 9, 8]
Resultant encrypted message:
NNOCSYHTTDFR
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

Vigenere cipher

Aim: To implement Vigenere cipher to encrypt and decrypt a given text.

Procedure:

For encryption:

- The first letter of the plaintext is paired with the first letter of the key.
- The sum of the numbers the ith letter of the key and the ith letter of the message correspond to mod 26 is the encrypted version of that letter.
- The key keeps getting repeated until the entire message gets encrypted.
- $CipherText_i = (PlainText_i + Key_i) \bmod 26$

For decryption:

- The first letter of the cipher text is paired with the first letter of the key.
- The difference of the Cipher text's ith letter and the Key's ith letter +26 mod 26 is the decrypted version of that letter.
- The key keeps getting repeated until the entire message gets decrypted.
- $PlainText_i = (CipherText_i - Key_i + 26) \bmod 26$.

Code:

```
print('19BCI0246 ANIRUDH KUMAR')

print('Vigenere Cipher')

enc=''

def encrypt(p,k):

    ct,i,j = "",0,0

    for _ in range(len(k),len(p)):

        k+=k[i%len(k)]

        i = i + 1

    for _ in p:

        ct+=alphabet[(alphabet.find(p[j])+alphabet.find(k[j]))%26]

        j = j + 1

    global enc

    enc=enc+ct
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
dec=''

def decrypt(c,k):

    nk,dt,i,j = [],"",0,0

    for _ in range(len(c)):

        nk+=(k[i%len(k)])

        i = i + 1

    for _ in c:

        dt+=alphabet[(alphabet.find(c[j])-alphabet.find(nk[j]))%26]

        j = j + 1

    global dec

    dec=dec+dt



alphabet = "abcdefghijklmnopqrstuvwxyz"

doagain=1

while doagain==1:

    choice=int(input('Enter 1 to encrypt and 2 to decrypt the text '))

    if choice==1:

        key = input("Enter the key: ").lower()

        pt1 = input("Enter the plaintext: ").lower()

        l=list(pt1.split())

        pt=''

        for i in l:

            pt=pt+i

        encrypt(pt,key)

        print(enc)

    elif choice==2:

        key = input("Enter the key: ").lower()

        cip1=input('Enter cipher text to be decrypted: ').lower()

        l=list(cip1.split())

        cip=''
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

```
    for i in l:
        cip=cip+i
    decrypt(cip,key)
    print('Decrypted message is: ',dec)
doagain=int(input('Do again? 1 to do again; 0 to stop '))
```

Screenshot of code:

jupyter 19BCI0246 CRYPTO-LAB-DA-1 Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

In [9]:
```
print('19BCI0246 ANIRUDH KUMAR')
print('Vigenere Cipher')
enc=''
def encrypt(p,k):
    ct,i,j = "",0,0
    for _ in range(len(k),len(p)):
        k+=k[i%len(k)]
        i = i + 1
    for _ in p:
        ct+=alphabet[(alphabet.find(p[j])+alphabet.find(k[j]))%26]
        j = j + 1
    global enc
    enc=enc+ct

dec=''
def decrypt(c,k):
    nk,dt,i,j = [],"",0,0
    for _ in range(len(c)):
        nk+=(k[i%len(k)])
        i = i + 1
    for _ in c:
        dt+=alphabet[(alphabet.find(c[j])-alphabet.find(nk[j]))%26]
        j = j + 1
    global dec
    dec=dec+dt


alphabet = "abcdefghijklmnopqrstuvwxyz"
doagain=1
while doagain==1:
    choice=int(input('Enter 1 to encrypt and 2 to decrypt the text '))
    if choice==1:
        key = input("Enter the key: ").lower()
        pt1 = input("Enter the plaintext: ").lower()
        l=list(pt1.split())
        pt=''
        for i in l:
            pt=pt+i
        encrypt(pt,key)
        print(enc)
    elif choice==2:
        key = input("Enter the key: ").lower()
        cip1=input('Enter cipher text to be decrypted: ').lower()
        l=list(cip1.split())
        cip=''
        for i in l:
            cip=cip+i
        decrypt(cip,key)
        print('Decrypted message is: ',dec)
    doagain=int(input('Do again? 1 to do again; 0 to stop '))
```

NAME: ANIRUDH KUMAR
REG NO: 19BCI0246

OUTPUT:

```
19BCI0246 ANIRUDH KUMAR
Vigenere Cipher
Enter 1 to encrypt and 2 to decrypt the text 1
Enter the key: vigenere
Enter the plaintext: i am anirudh
diseamiyyp
Do again? 1 to do again; 0 to stop 1
Enter 1 to encrypt and 2 to decrypt the text 2
Enter the key: vigenere
Enter cipher text to be decrypted: diseamiyyp
Decrypted message is:  iamanirudh
Do again? 1 to do again; 0 to stop 0
```