Perform encryption and decryption using following transposition techniques [Rail fence]

Ex. No : 2a

Date :

Aim:

To perform encryption and decryption using Rail fence.

Algorithm:

Step 1: Obtain the text for encryption /decryption

Step 2: Get input from the user to Encrypt/Decrypt

Step 3: Get the key from the user.

Step 4: Perform an encryption/decryption using key.

Step 5: Output the corresponding Plaintext/Cipher Text.

Source code:

```
def encrypt(text,depth):
    n=len(text)
    row=depth
    text+= 'x'*( (-(n%row))%row)
    n=len(text)
    col= (n//row)
    cmat = [['' for j in range(col)]for i in range(row)]
    k=0
    for i in range(col):
        for j in range(row):
            cmat[j][i]=text[k]
            k+=1
    ctext=""
    for i in range(row):
        for j in range(col):
        ctext+=cmat[i][j]
    print("Encrypted Text ",ctext)
def decrypt(text,depth):
    n=len(text)
    row=depth
    col= (n//row)
    cmat = [['' for j in range(col)]for i in range(row)]
    k=0
    for i in range(row):
        for j in range(col):
            cmat[i][j]=text[k]
            k+=1
    ptext=""
    for i in range(col):
        for j in range(row):
            ptext+=cmat[j][i]
    print("Decrypted Text ",ptext)
def main():
    text = input("Enter Text to encrypt/decrypt : ")
    depth=int(input("Enter key : ")) #depth can be inferred as key
    choice = int(input("Enter 1.Encrypt 2.Decrypt : "))
    if(1==choice):
        encrypt(text,depth)
    else:
        decrypt(text,depth)
main()
```

Output:

```
Enter the text for encrypt/decrypt : happy
Enter the key : 3
1.Encrypt 2.Decrypt : 1
Encrypted text : hpaypx
```

```
Enter the text for encrypt/decrypt : hpaypx
Enter the key : 3
1.Encrypt 2.Decrypt : 2
Decrypted text : happyx
```

Result:

The Rail fence encryption and decryption technique was executed successfully and output was verified.

Perform encryption and decryption using following transposition techniques [row & Column Transformation]

Ex. No : 2b

Date :

Aim:

To perform encryption and decryption using row & Column Transformation.

Algorithm:

```
Step 1: Obtain the text for encryption /decryption
Step 2: Get input from the user to Encrypt/Decrypt
Step 3: Get the key from the user.
Step 4: Perform an encryption/decryption using key.
Step 5: Output the corresponding Plaintext/Cipher Text.
```

Source code:

```
def encrypt(text,key):
        n=len(text)
        col=len(key)
        text+= 'x'*( (-(n%col))%col)
        n=len(text)
        row= (n//col)
        cmat = [['' for j in range(col)]for i in range(row)]
        k=0
        for i in range(row):
                for j in range(col):
                        cmat[i][j]=text[k]
                        k+=1
        sort_key=sorted(list(key))
        ctext=""
```

```
    for i in range(col):
        curr_col = key.find(sort_key[i])
        ctext+= ''.join([cmat[i][curr_col] for i in range(row)])
    print("Encrypted Text ",ctext)
def decrypt(text,key):
    n=len(text)
    col=len(key)
    row= (n//col)
    cmat = [['' for j in range(col)]for i in range(row)]
    k=0
    sort_key=sorted(list(key))
    for i in range(col):
        curr_col= key.find(sort_key[i])
        for j in range(row):
            cmat[j][curr_col]=text[k]
            k+=1
    ptext=""
    for i in range(row):
        for j in range(col):
            ptext+=cmat[i][j]
    print("Decrypted Text ",ptext)
def main():
    text = input("Enter Text to encrypt/decrypt : ")
    key = input("Enter the key : ")
    choice = int(input("Enter 1.Encrypt 2.Decrypt : "))
    if(1==choice):
        encrypt(text,key)
    else:
        decrypt(text,key)
main()
```

Output:

```
Enter Text to encrypt/decrypt : happy

Enter key : god

Enter 1.Encrypt 2.Decrypt :  1

Encrypted Text :  pxhpay
```

```
Enter Text to encrypt/decrypt : pxhpay

Enter key : god

Enter 1.Encrypt 2.Decrypt :  2

Decrypted Text :  happyx
```

Result:

The row & Column Transformation encryption and decryption technique was executed successfully and output was verified.