Name: Aditya Kulkarni	Name: Devansh Kahndor
Roll Number: 16010121093	Roll Number: 16010121085

INFORMATION SECURITY IA-1

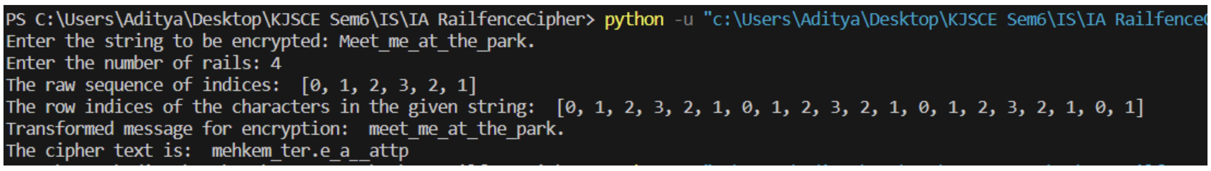
Functionality Selected:

1. Encryption using RailFence technique.
2. Decryption using RailFence technique.
3. User defined rails encryption and decryption.

Github Repo: https://github.com/AdityaK-1302/Rail-Fence-Cipher

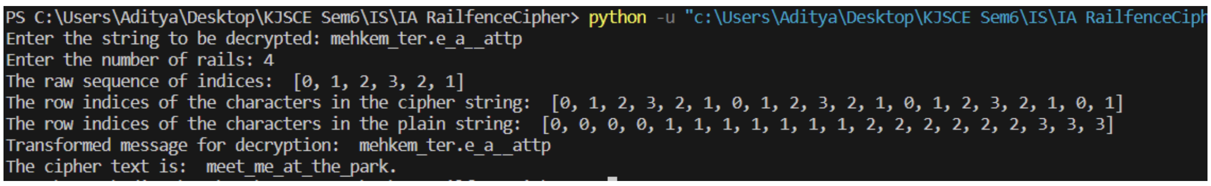
Implementation Snapshots:

1. Encryption:



```
PS C:\Users\Aditya\Desktop\KJSCE Sem6\IS\IA RailfenceCipher> python -u "c:\Users\Aditya\Desktop\KJSCE Sem6\IS\IA Railfence
Enter the string to be encrypted: Meet_me_at_the_park.
Enter the number of rails: 4
The raw sequence of indices:  [0, 1, 2, 3, 2, 1]
The row indices of the characters in the given string:  [0, 1, 2, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, 1]
Transformed message for encryption:  meet_me_at_the_park.
The cipher text is:  mehkem_ter.e_a__attp
```

2. Decryption:

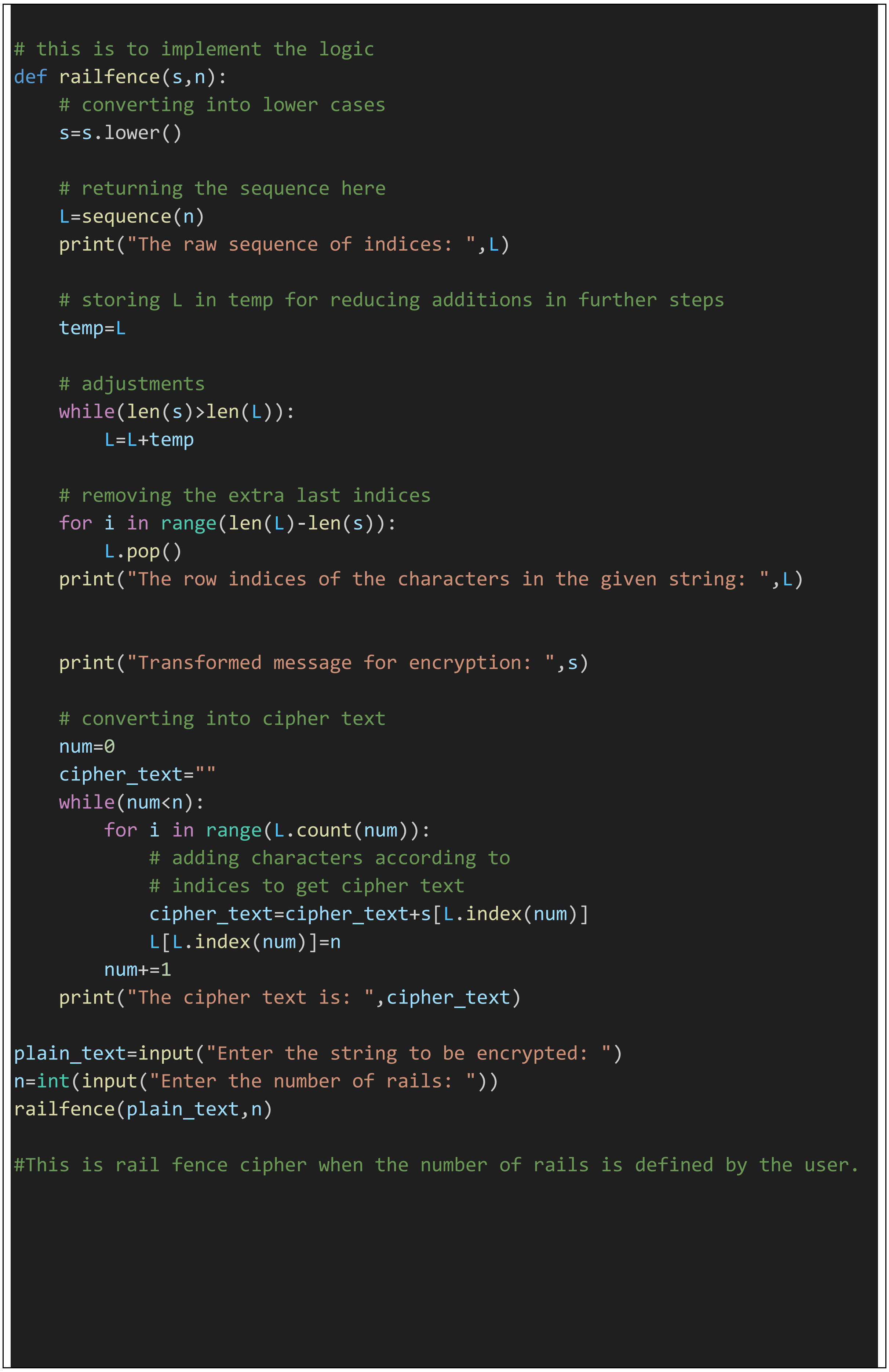


```
PS C:\Users\Aditya\Desktop\KJSCE Sem6\IS\IA RailfenceCipher> python -u "c:\Users\Aditya\Desktop\KJSCE Sem6\IS\IA RailfenceCiph
Enter the string to be decrypted: mehkem_ter.e_a__attp
Enter the number of rails: 4
The raw sequence of indices:  [0, 1, 2, 3, 2, 1]
The row indices of the characters in the cipher string:  [0, 1, 2, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, 1]
The row indices of the characters in the plain string:  [0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3]
Transformed message for decryption:  mehkem_ter.e_a__attp
The cipher text is:  meet_me_at_the_park.
```

Code:

```
ENCRYPTION


# this function is to get the desired sequence
def sequence(n):
    arr=[]
    i=0
    # creating the sequence required for
    # implementing railfence cipher
    # the sequence is stored in array
    while(i<n-1):
        arr.append(i)
        i+=1
    while(i>0):
        arr.append(i)
        i-=1
    return(arr)
```



```python
# this is to implement the logic
def railfence(s,n):
    # converting into lower cases
    s=s.lower()

    # returning the sequence here
    L=sequence(n)
    print("The raw sequence of indices: ",L)

    # storing L in temp for reducing additions in further steps
    temp=L

    # adjustments
    while(len(s)>len(L)):
        L=L+temp

    # removing the extra last indices
    for i in range(len(L)-len(s)):
        L.pop()
    print("The row indices of the characters in the given string: ",L)


    print("Transformed message for encryption: ",s)

    # converting into cipher text
    num=0
    cipher_text=""
    while(num<n):
        for i in range(L.count(num)):
            # adding characters according to
            # indices to get cipher text
            cipher_text=cipher_text+s[L.index(num)]
            L[L.index(num)]=n
        num+=1
    print("The cipher text is: ",cipher_text)

plain_text=input("Enter the string to be encrypted: ")
n=int(input("Enter the number of rails: "))
railfence(plain_text,n)

#This is rail fence cipher when the number of rails is defined by the user.
```

Decryption

```python
# this function is to get the desired sequence
def sequence(n):
    arr=[]
    i=0
    # creating the sequence required for
    # implementing railfence cipher
    # the sequence is stored in array
    while(i<n-1):
        arr.append(i)
        i+=1
    while(i>0):
        arr.append(i)
        i-=1
    return(arr)

# this is to implement the logic
def railfence(cipher_text,n):
    # converting into lower cases
    cipher_text=cipher_text.lower()

    # If you want to remove spaces,
    # you can uncomment this
    # s=s.replace(" ","")

    # returning the sequence here
    L=sequence(n)
    print("The raw sequence of indices: ",L)

    # storing L in temp for reducing additions in further steps
    # if not stored and used as below, the while loop
    # will create L of excess length
    temp=L

    # adjustments
    while(len(cipher_text)>len(L)):
        L=L+temp

    # removing the extra last indices
    for i in range(len(L)-len(cipher_text)):
        L.pop()

    # storing L.sort() in temp1
    temp1=sorted(L)
    print("The row indices of the characters in the cipher string: ",L)

    print("The row indices of the characters in the plain string: ",temp1)

    print("Transformed message for decryption: ",cipher_text)

    # converting into plain text
    plain_text=""
    for i in L:
        # k is index of particular character in the cipher text
        # k's value changes in such a way that the order of change
        # in k's value is same as plaintext order
        k=temp1.index(i)
        temp1[k]=n
        plain_text+=cipher_text[k]

    print("The cipher text is: ",plain_text)


cipher_text=input("Enter the string to be decrypted: ")
n=int(input("Enter the number of rails: "))
railfence(cipher_text,n)
```