**SCHOOL OF COMPUTER SCIENCE ENGINEERING AND APPLICATION**

**BCA TY SEM VI**

**SUBJECT NAME: INFORMATION SECURITY**

**LAB- ASSIGNMENT NO 2**

**AIM: IMPLEMENT RAIL FENCE ALGORITHM. (ENCRYPTION AND DECRYPTION)**

**RAIL FENCE:**

The Rail Fence Cipher is a type of transposition cipher, where the plaintext is written in a zigzag pattern across a number of "rails" or "lines" (hence the name), and then read off as ciphertext. It's a simple and ancient method of encryption.

Here's how it works:

* Choose the number of rails you want to use. Let's call this number "n".
* Write your message diagonally across "n" rows, starting from the top left and moving downward and diagonally to the right.
* Once you reach the bottom rail, reverse the direction and move diagonally upwards and to the right until you reach the top rail again. Continue this zigzag pattern until you've written out the entire message.
* Read off the message row by row to obtain the ciphertext.

**ALGORITHM:**

The Rail Fence Cipher, also known as the Zigzag Cipher, is a simple form of transposition cipher. In this algorithm, the plaintext is written diagonally and read off horizontally in a zigzag pattern.

Here's a step-by-step explanation of the Rail Fence Algorithm

1. **\*\*Encryption:\*\***

a. Write down the plaintext in a zigzag pattern along a number of "rails" or "waves".

b. For encryption, the number of rails determines the key. For example, if the key is 3, the plaintext is written down in a zigzag pattern across 3 rails.

c. After writing the plaintext along the rails, read off the characters row by row to create the ciphertext.

2. \***\*Decryption:\*\***

a. To decrypt, follow a similar process. First, determine the number of rails based on the key.

b. Write down the ciphertext in a zigzag pattern along the determined number of rails.

c. Then, read off the characters row by row to reveal the original plaintext.

Here's a simple example:

Suppose we have the plaintext "HELLO WORLD" and a key of 3.

**Encryption:**

H . . . O . . . R

. E . L . W . L .

. . L . . . D . .

The ciphertext would be "HOR ELLWL LD".

**Decryption:**

H . . . O . . . R

. E . L . W . L .

. . L . . . D . .

Reading row by row, we get the plaintext "HELLO WORLD" again.

**PSUEDO CODE IN PYTHON FOR RAIL FENCE ENCRYPTION AND DECRYPTION WITH OUTPUT:**

# Railfence Cipher

def Railfence(key, textsize):

    counter = 0

    sign = "+"

    railfence = ""

    for i in range(textsize):

        counter = eval(f"{counter}{sign}1")

        railfence+=str(counter)

        if counter==key:

            sign='-'

        elif counter==1:

            sign='+'

    return railfence

def encrypt(plaintext,key):

    textsize = len(plaintext)

    ciphertext = ""

    railfence = Railfence(key=key,textsize=textsize)

    for i in range(1,key+1):

        for char\_i in range(textsize):

            if railfence[char\_i]==f"{i}":

                ciphertext=ciphertext+plaintext[char\_i]

    return ciphertext

def decrypt(ciphertext,key):

    textsize = len(ciphertext)

    plaintext = ""

    railfence = Railfence(key=key,textsize=textsize)

    antirailfence = sorted([index for index in railfence])

    for char\_i in range(textsize):

        for j in range(textsize):

            if railfence[char\_i]==antirailfence[j]:

                ch = ciphertext[j]

                antirailfence[j]=0

                break

        plaintext+=ch

    return plaintext

plaintext = input("Enter a Plaintext: ")

key = int(input("Enter Key: "))

ciphertext = encrypt(plaintext=plaintext,key=key)

print("encrypted text:",ciphertext)

print("decrypted text:",decrypt(ciphertext=ciphertext,key=key))

PS C:\Users\T480> & C:/Users/T480/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/T480/Desktop/sem\_6/IS/assignments/lab assignments/RAILFENCE.PY"

Enter a Plaintext: HELLO WORLD

Enter Key: 3

encrypted text: HOREL OLLWD

decrypted text: HELLO WORLD