

EXP NO:3

DATE:

RAIL FENCE CIPHER

Aim: To implement an encryption algorithm using Rail Fence Cipher technique.

Algorithm:

- Step 1: Declare msg and key, initializing msg with the original message, and set key to the desired rail fence key.
- Step 2: Create railMatrix with dimensions [key][msgLen], initializing elements with newline characters.
- Step 3: Iterate through msg, placing characters in railMatrix based on the Rail Fence Cipher pattern, updating row and col.
- Step 4: Print the encrypted message by traversing railMatrix, excluding newline characters.
- Step 5: Return 0 for successful execution and program termination.

Program:

```
#include<stdio.h>
```

```
#include<string.h>
```

```
void encryptMsg(char msg[], int key){
```

```
    int msgLen = strlen(msg), i, j, k = -1, row = 0, col = 0;
```

```
    char railMatrix[key][msgLen];
```

```
    for(i = 0; i < key; ++i)
```

```
        for(j = 0; j < msgLen; ++j)
```

```
            railMatrix[i][j] = '\n';
```

```
    for(i = 0; i < msgLen; ++i){
```

```
        railMatrix[row][col++] = msg[i];
```

```
        if(row == 0 || row == key-1)
```

```

        k= k * (-1);
        row = row + k;
    }

    printf("\nEncrypted Message: ");

    for(i = 0; i < key; ++i)
    for(j = 0; j < msgLen; ++j)
    if(railMatrix[i][j] != '\n')
    printf("%c", railMatrix[i][j]);
    }
int main(){
    char msg[] = "This is Thrisha";
    int key = 3;
    printf("Original Message: %s", msg);
    encryptMsg(msg, key);    return 0;
}

```

Output:

```

/tmp/FHeyuCLAX8.o
Original Message: I am Viniiii
Encrypted Message: I ii mVniiaii

=== Code Execution Successful ===

```

Result: