**IT8761 – Security Laboratory**

**Reshma Ramesh Babu**

**312217104129**

**Aim:** To implement the transposition techniques: RailFence Cipher and Row & Column Cipher

**RailFence Cipher:**

**Code:**

import java.util.\*;

class RailFenceBasic{

int depth;

String Encryption(String plainText,int depth)

{

int r=depth,len=plainText.length();

int c=len/depth;

char mat[][]=new char[r][c];

int k=0;

String cipherText="";

for(int i=0;i< c;i++)

{

for(int j=0;j< r;j++)

{

if(k!=len)

mat[j][i]=plainText.charAt(k++);

else

mat[j][i]='X';

}

}

for(int i=0;i< r;i++)

{

for(int j=0;j< c;j++)

{

cipherText+=mat[i][j];

}

}

return cipherText;

}

String Decryption(String cipherText,int depth)

{

int r=depth,len=cipherText.length();

int c=len/depth;

char mat[][]=new char[r][c];

int k=0;

String plainText="";

for(int i=0;i< r;i++)

{

for(int j=0;j< c;j++)

{

mat[i][j]=cipherText.charAt(k++);

}

}

for(int i=0;i< c;i++)

{

for(int j=0;j< r;j++)

{

plainText+=mat[j][i];

}

}

return plainText;

}

}

class RailFence{

public static void main(String args[])

{

RailFenceBasic rf=new RailFenceBasic();

Scanner scn=new Scanner(System.in);

int depth;

String plainText,cipherText,decryptedText;

char ch;

int choice;

do{

System.out.println("Menu:\n1) Encryption\n2) Decryption");

choice=scn.nextInt();

switch(choice)

{

case 1: System.out.println("Enter plain text:");

plainText=scn.next();

System.out.println("Enter depth for Encryption:");

depth=scn.nextInt();

cipherText=rf.Encryption(plainText,depth);

System.out.println("Encrypted text is:\n"+cipherText);

break;

case 2: System.out.println("Enter cipher text:");

cipherText=scn.next();

System.out.println("Enter depth for Decryption:");

depth=scn.nextInt();

decryptedText=rf.Decryption(cipherText, depth);

System.out.println("Decrypted text is:\n"+decryptedText);

break;

}

System.out.println("\nDo you want to continue? y/n");

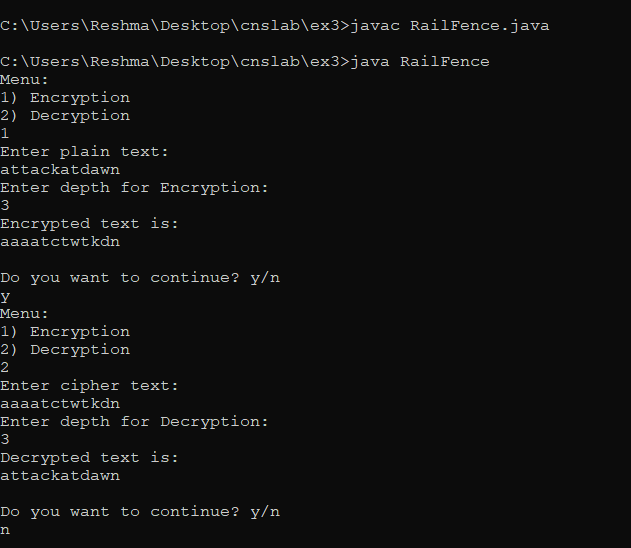
ch = scn.next().charAt(0);

}while(ch!='n');

}

}

**Output:**



**Row & Column Transposition Cipher:**

**Code:**

import java.util.\*;

public class RowColumn{

char arr[][],encrypt[][],decrypt[][],keya[],keytemp[];

public void creatematrixE(String s,String key,int row,int column){

arr=new char[row][column];

int k=0;

keya=key.toCharArray();

for(int i=0;i<row;i++){

for(int j=0;j<column;j++)

{

if(k<s.length())

{

arr[i][j]=s.charAt(k);

k++;

}

else

{

arr[i][j]=' ';

}

}

}

}

public void createkey(String key,int column){

keytemp=key.toCharArray();

for(int i=0;i<column-1;i++){

for(int j=i+1;j<column;j++)

{

if(keytemp[i]>keytemp[j])

{

char temp=keytemp[i];

keytemp[i]=keytemp[j];

keytemp[j]=temp;

}

}

}

}

public void creatematrixD(String s,String key,int row,int column){

arr=new char[row][column];

int k=0;

keya=key.toCharArray();

for(int i=0;i<column;i++)

{

for(int j=0;j<row;j++)

{

if(k<s.length())

{

arr[j][i]=s.charAt(k);

k++;

}

else

{

arr[j][i]=' ';

}

}

}

}

public void encrypt(int row,int column){

encrypt=new char[row][column];

for(int i=0;i<column;i++)

{

for(int j=0;j<column;j++)

{

if(keya[i]==keytemp[j])

{

for(int k=0;k<row;k++)

{

encrypt[k][j]=arr[k][i];

}

keytemp[j]='?';

break;

}

}

}

}

public void decrypt(int row,int column){

decrypt=new char[row][column];

for(int i=0;i<column;i++)

{

for(int j=0;j<column;j++)

{

if(keya[j]==keytemp[i])

{

for(int k=0;k<row;k++)

{

decrypt[k][j]=arr[k][i];

}

keya[j]='?';

break;

}

}

}

}

public void resultE(int row,int column,char arr[][]){

System.out.println("Encrypted text:");

for(int i=0;i<column;i++)

{

for(int j=0;j<row;j++)

{

System.out.print(arr[j][i]);

}

}

}

public void resultD(int row,int column,char arr[][]) {

System.out.println("Decrypted text:");

for(int i=0;i<row;i++)

{

for(int j=0;j<column;j++)

{

System.out.print(arr[i][j]);

}

}

}

public static void main(String args[]){

int row,column,choice;

char ch;

RowColumn obj=new RowColumn();

Scanner in = new Scanner(System.in);

do{

System.out.println("Menu:\n1) Encryption\n2) Decryption");

choice=in.nextInt();

System.out.println("Enter the string:");

String s=in.next();

System.out.println("Enter the key:");

String key=in.next();

row=s.length()/key.length();

if(s.length()%key.length()!=0)

row++;

column=key.length();

switch(choice)

{

case 1: obj.creatematrixE(s,key,row,column);

obj.createkey(key,column);

obj.encrypt(row,column);

obj.resultE(row,column,obj.encrypt);

break;

case 2: obj.creatematrixD(s,key,row,column);

obj.createkey(key,column);

obj.decrypt(row,column);

obj.resultD(row,column,obj.decrypt);

break;

}

System.out.println("\nDo you want to continue? y/n");

ch = in.next().charAt(0);

}while(ch!='n');

}

}

**Output:**

