Shankar Narayanan
312217104146
17-9-2020

SSN College of Engineering,

Department of Computer Science and Engineering

CS6711 Security Laboratory

Exercise 3

Rail fence cipher :

```
import java.io.*;
import java.lang.*;
import java.util.*;

class RailFenceCipher{

    public String ptext;
    public int key;
    public char[][] mat;
    public StringBuffer ctext;

    public RailFenceCipher(String p, int k){
        this.ptext = p;
        this.key = k;
        mat = new char[1000][1000];
        ctext = new StringBuffer();

        for(int i=0;i<1000;i++){
            for(int j=0;j<1000;j++){
                mat[i][j]='0';
            }
        }
    }
```

```
public void encrypt(){
	int row=0, col=0;
	boolean down_dir=false;
	int n,i,j;
	n = ptext.length();

	for(i=0;i<n;i++){

		if(row==0 || row==key-1) down_dir=!down_dir;

		char ch = ptext.charAt(i);
		mat[row][col]= ch;

		if(down_dir) row++;
		else row--;
		col++;
	}

	for(i=0;i<key;i++){
		for(j=0;j<n+1;j++){
			if(mat[i][j] != '0')
				ctext.append(mat[i][j]);
		}
	}
}

public void displayRail(){
	System.out.println("\n\n");
	int i,j,n;
	n=ptext.length();
	for(i=0;i<key;i++){
		for(j=0;j<n+1;j++){
			if(mat[i][j] != '0')
				System.out.print(mat[i][j]+" ");
			else
				System.out.print("_ ");
		}
		System.out.print("\n");
	}
```

```
        System.out.println("\n\n");
    }

    public void decrypt(){
        int n,i,j;
        n = ptext.length();

        for(i=0;i<1000;i++){
                for(j=0;j<1000;j++){
                        mat[i][j]='0';
                }
        }

        int row=0, col=0;
        boolean down_dir=false;
        for(i=0;i<n;i++){

                if(row==0 || row==key-1) down_dir=!down_dir;

                mat[row][col] = '*';

                if(down_dir) row++;
                else row--;
                col++;
        }

        int k=0;
        for(i=0;i<key;i++){
                for(j=0;j<=n;j++){
                        if(mat[i][j]=='*'){
                                char ch = ctext.charAt(k);
                                mat[i][j] = ch;
                                k++;
                        }
                }
        }

        down_dir=false;
        StringBuffer decr = new StringBuffer();
        row=0;
```

```
        col=0;
        for(i=0;i<n;i++){

                if(row==0 || row==key-1) down_dir=!down_dir;

                decr.append(mat[row][col]);

                if(down_dir) row++;
                else row--;
                col++;
        }

        System.out.println("The decrypted plaintext : "+decr);
    }

}

public class RFDriver{

    public static void main(String[] args){
        Scanner in = new Scanner(System.in);

        String ptext;
        int key;

        System.out.println("Enter the plaintext string (small letters) : ");
        ptext = in.nextLine();

        System.out.println("Enter the key value : ");
        key = in.nextInt();

        RailFenceCipher rfc = new RailFenceCipher(ptext, key);

        rfc.encrypt();

        rfc.displayRail();

        System.out.println("The cipher text : "+rfc.ctext);

        rfc.decrypt();
```

```
   }
}
```

OUTPUT:

```
(base) Shankars-MacBook-Pro:Ex14 shankar99$ javac RFDriver.java
(base) Shankars-MacBook-Pro:Ex14 shankar99$ java RFDriver
Enter the plaintext string (small letters) :
the vault is small
Enter the key value :
3


t _ _ _ v _ _ _ t _ _ _  _ _ _ l _ _
_ h _  _ a _ l _  _ s _ s _ a _ l _
_ _ e _ _ _ u _ _ _ i _ _ _ m _ _ _ _


The cipher text : tvt lh al ssaleuim
The decrypted plaintext : the vault is small
```

RowColCipher:

```
import java.io.*;
import java.lang.*;
import java.util.*;

class RowColCipher{

    String ptext;
    int[] key;
    StringBuffer ctext;
    char[][] mat;
    int m,row, col;

    public RowColCipher(String p, int[] k,int l){
        System.out.println("constructor");
        this.ptext = p;
        this.key = k;
        this.m = l;
        ctext = new StringBuffer();
        mat = new char[1000][1000];
        row=0;
        col=0;

        for(int i=0;i<1000;i++){
            for(int j=0;j<1000;j++){
                mat[i][j]='0';
            }
        }

    }


    public void encrypt(){
        System.out.println("encrypt");
        int i,j,n;
```

```
n = ptext.length();

int k=0;
i=0;
j=0;
while(k<n){
	mat[i][j] = ptext.charAt(k);
	k++;
	j++;

	if(j>=m){
		j=0;
		i++;
	}
}

while(j<m){
	mat[i][j]=' ';
	j++;
}

row=i+1;
col=m;

//displaying the row col table
for(i=0;i<m;i++){
	System.out.print(key[i]+" ");
}
System.out.println("\n");

for(i=0;i<row;i++){
	for(j=0;j<col;j++){
		System.out.print(mat[i][j]+" ");
	} System.out.print("\n");
}System.out.print("\n");

//getting cipher text from row col table

for(i=1;i<=m;i++){
	int ind=-1;
```

```
            for(j=0;j<m;j++){
                if(key[j]==i) {
                    ind=j;
                    break;
                }
            }

            for(j=0;j<row;j++){
                char ch = mat[j][ind];
                ctext.append(ch);
            }
        }



        System.out.println("The encrypted ciphertext : "+ctext);
}

public void decrypt(){
        int i,j,n;
        n = ctext.length();

        //clearing the matrix mat[][] for decryption

        int k=0;

        for(i=1;i<=m;i++){
            int ind=-1;

            for(j=0;j<m;j++){
                if(key[j]==i) ind = j;
            }

            for(j=0;j<row;j++){
                mat[j][ind]=ctext.charAt(k);
                k++;
            }
        }

        //reading the matrix row-wise to get the plaintext back
```

```
        StringBuffer decr = new StringBuffer();
        for(i=0;i<row;i++){
                for(j=0;j<col;j++){
                        decr.append(mat[i][j]);
                }
        }

        System.out.println("\n\nThe decrypted message : "+decr);
    }


}

public class RowColDriver{

    public static void main(String[] args){
        Scanner in = new Scanner(System.in);

        String ptext;
        int i,m;
        int[] key = new int[1000];

        System.out.println("Enter the plaintext : ");
        ptext = in.nextLine();

        System.out.println("Enter the key length  : ");
        m = in.nextInt();
        //System.out.println(m);

        System.out.println("Enter the key  : ");
        for(i=0;i<m;i++){
                int a;
                a = in.nextInt();
                key[i] = a;
                //System.out.println(a);
        }

        //check for key validity here
```

```
        //--------------------

        RowColCipher rcc = new RowColCipher(ptext, key, m);

        rcc.encrypt();

        rcc.decrypt();

    }
}
```

OUTPUT :

```
(base) Shankars-MacBook-Pro:Ex14 shankar99$ java RowColDriver
Enter the plaintext :
The vault is small
Enter the key length  :
4
Enter the key  :
3 1 2 4

encrypt
3 1 2 4

T h e
v a u l
t  i s
 s m a
l l
```

The encrypted ciphertext : ha sleuim Tvt l lsa

The decrypted message : The vault is small
(base) Shankars-MacBook-Pro:Ex14 shankar99$

Result : Implemented the transposition cipher methods RailFence Cipher and RowColCipher and understood the working of it.