D12B\_01\_SANJANA ASRANI\_CSS-02 24/01/23

1. Caesar Cipher / Additive Shift Cipher :

CODE :

import java.util.Scanner;

public class trial {

public static final String Alphabet = "abcdefghijklmnopqrstuvwxyz";

//ENCRYPTION

public static String encryptData(String input, int shift){

input = input.toLowerCase();

System.out.print(" P| ");

for(int i = 0; i < input.length(); i++){

System.out.print(input.charAt(i) + " | "); }

String encryptStr = "";

if(input==" "){

encryptStr+=" ";

}

for(int i = 0;i<input.length();i++){

int position = Alphabet.indexOf(input.charAt(i));

int encryptPos = (shift+position)%26;

char encryptChar = Alphabet.charAt(encryptPos);

encryptStr += encryptChar;

}

System.out.println();

System.out.print(" V(P) | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.print(Alphabet.indexOf(input.charAt(i)) + " | "); }

System.out.println();

System.out.print(" P+K | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.print((shift+Alphabet.indexOf(input.charAt(i)) + " | ")); }

System.out.println();

System.out.print(" V(C) | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.print((shift+Alphabet.indexOf(input.charAt(i)))%26 + " | "); }

System.out.println();

System.out.print(" C | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.print(encryptStr.charAt(i) + " | "); }

System.out.println();

return encryptStr;

}

// DECRYPTION

public static String decryptData(String input, int shift){

input = input.toLowerCase();

System.out.println();

System.out.print(" C | ");

for(int i = 0; i < input.length(); i++){

System.out.print(input.charAt(i) + " | "); }

// HERE INPUT WILL BE THE ALREADY ENCRYPTED STRING

String decryptStr = "";

for(int i = 0;i<input.length();i++){

int position = Alphabet.indexOf(input.charAt(i));

int decryptPos = (position-shift)%26;

if(decryptPos<0){

decryptPos = Alphabet.length() + decryptPos;

}

char decryptChar = Alphabet.charAt(decryptPos);

decryptStr += decryptChar;

}

System.out.println();

System.out.print("V(C) | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.print(Alphabet.indexOf(input.charAt(i)) + " | "); }

System.out.println();

System.out.print("C-K | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.print((Alphabet.indexOf(input.charAt(i))-shift) + " | "); }

System.out.println();

System.out.print("V(P) | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.print((Alphabet.indexOf(input.charAt(i))-shift)%26 + " | "); }

System.out.println();

System.out.print("P | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.print(decryptStr.charAt(i) + " | "); }

System.out.println();

return decryptStr;

}

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

System.out.println("Enter a string for encryption using Caesar Cipher: ");

String input = sc.nextLine();

System.out.println("Enter shift: ");

int shift = Integer.valueOf(sc.nextLine());

System.out.println();

System.out.println();

System.out.println("Encrypted Data ===> "+encryptData(input, shift));

System.out.println();

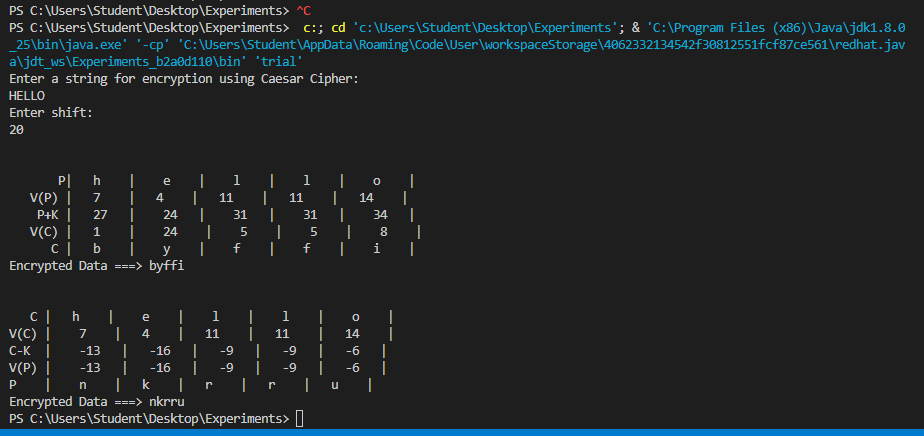
// System.out.println("Decrypted Data ===> "+decryptData(encryptData(input, shift), shift));

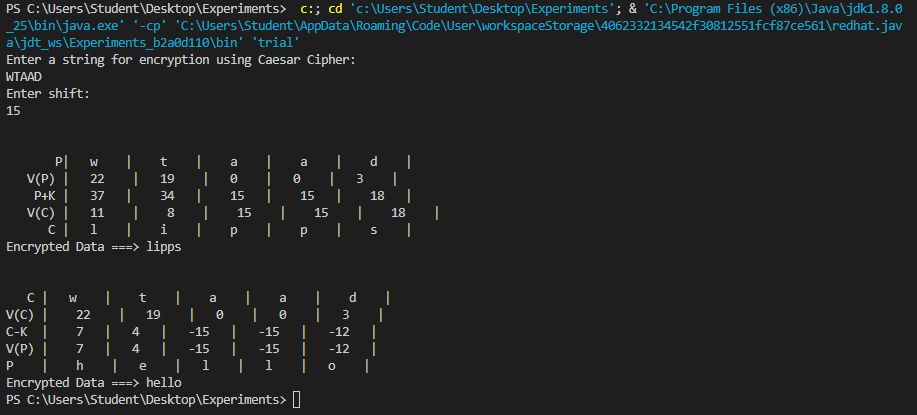
System.out.println("Encrypted Data ===> "+decryptData(input, shift));

sc.close();

}

}





Transposition Cipher : Rail fence Technique 31/01/23

code

import java.util.\*;

class RailFenceBasic{

int depth;

String Encryption(String plainText,int depth)throws Exception

{

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)throws Exception

{

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[])throws Exception

{

RailFenceBasic rf=new RailFenceBasic();

Scanner scn=new Scanner(System.in);

int depth;

String plainText,cipherText,decryptedText;

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

plainText=scn.nextLine();

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

depth=scn.nextInt();

cipherText=rf.Encryption(plainText,depth);

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

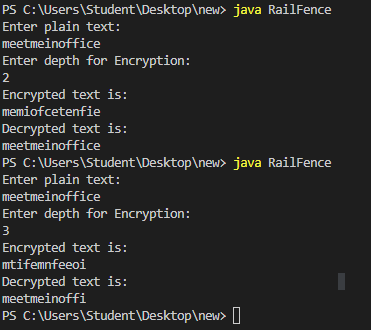
decryptedText=rf.Decryption(cipherText, depth);

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

}

}

Output :



/////////////////////////////////////////////////////////////////////////////////////////////////////////////

Railfence ke baad additive 👍

import java.util.\*;

class temp{

int depth;

public static final String Alphabet = "abcdefghijklmnopqrstuvwxyz";

// EncryptData : Additive Encryption

// Encryption : Rail Fence Encryption

public static String encryptData(String input, int shift){

input = input.toLowerCase();

System.out.print(" P | ");

for(int i = 0; i < input.length(); i++){

System.out.printf("%-15s\t",input.charAt(i)) ; }

String encryptStr = "";

if(input==" "){

encryptStr+=" ";

}

for(int i = 0;i<input.length();i++){

int position = Alphabet.indexOf(input.charAt(i));

int encryptPos = (shift+position)%26;

char encryptChar = Alphabet.charAt(encryptPos);

encryptStr += encryptChar;

}

System.out.println();

// %32s%10d%16s"

System.out.print(" V(P) | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.printf("%-15s\t",Alphabet.indexOf(input.charAt(i)) ); }

System.out.println();

System.out.print(" P+K | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.printf("%-15s\t",(shift+Alphabet.indexOf(input.charAt(i)) )); }

System.out.println();

System.out.print(" V(C) | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.printf("%-15s\t",(shift+Alphabet.indexOf(input.charAt(i)))%26 ); }

System.out.println();

System.out.print(" C | ");

for(int i = 0; i < encryptStr.length(); i++){

System.out.printf("%-15s\t",encryptStr.charAt(i) ); }

System.out.println();

return encryptStr;

}

// DECRYPTION

public static String decryptData(String input, int shift){

input = input.toLowerCase();

System.out.println();

System.out.print(" C | ");

for(int i = 0; i < input.length(); i++){

System.out.printf("%-15s\t",input.charAt(i) ); }

// HERE INPUT WILL BE THE ALREADY ENCRYPTED STRING

String decryptStr = "";

for(int i = 0;i<input.length();i++){

int position = Alphabet.indexOf(input.charAt(i));

int decryptPos = (position-shift)%26;

if(decryptPos<0){

decryptPos = Alphabet.length() + decryptPos;

}

char decryptChar = Alphabet.charAt(decryptPos);

decryptStr += decryptChar;

}

System.out.println();

System.out.print("V(C) | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.printf("%-15s\t",Alphabet.indexOf(input.charAt(i)) ); }

System.out.println();

System.out.print("C-K | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.printf("%-15s\t",(Alphabet.indexOf(input.charAt(i))-shift) ); }

System.out.println();

System.out.print("V(P) | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.printf("%-15s\t",(Alphabet.indexOf(input.charAt(i))-shift)%26 ); }

System.out.println();

System.out.print("P | ");

for(int i = 0; i < decryptStr.length(); i++){

System.out.printf("%-15s\t",decryptStr.charAt(i) ); }

System.out.println();

return decryptStr;

}

String Encryption(String plainText,int depth)throws Exception

{

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)throws Exception

{

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 finalMix{

public static void main(String args[])throws Exception

{

temp rf=new temp();

Scanner scn=new Scanner(System.in);

int depth;

String plainText,cipherText,decryptedText;

System.out.println(" RAILFENCE :\n\n\n");

System.out.println("Enter P : ");

plainText=scn.nextLine();

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

depth=scn.nextInt();

cipherText=rf.Encryption(plainText,depth);

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

System.out.println("Enter shift For Additive Cipher: ");

int shift = scn.nextInt();

cipherText=rf.encryptData(cipherText,shift);

System.out.println("final Encrypted Data using Additive ===> "+cipherText);

System.out.println("\n\n\nDECRYPTION NOW\n\n\n");

decryptedText=rf.decryptData(cipherText,shift);

System.out.println(" cipher text after decryption of additive ===> "+decryptedText);

decryptedText=rf.Decryption(decryptedText, depth);

System.out.println(" final cipher text after decryption of railfence ===> "+decryptedText);

}

}