Ceasar

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template

\*/

package atbm\_th1;

import java.util.Scanner;

/\*\*

\*

\* @author ASUS

\*/

public class Ceasar {

static int n;

public static void maHoa(){

Scanner s = new Scanner(System.in);

System.out.println("Nhập thông điệp cần mã hóa: ");

String plain = s.nextLine();

System.out.println("Nhập độ dịch chuyển: ");

int k = s.nextInt();

String cipher = "";

char alpha;

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

alpha = plain.charAt(i);

if(alpha >= 'a' && alpha <= 'z' || alpha >= 'A' && alpha <= 'Z'){

alpha = (char)(alpha + k);

}else if(alpha > 'z'){

alpha = (char)(alpha + 'a' - 'z' - 1);

}else if(alpha > 'Z'){

alpha = (char)(alpha + 'A' - 'Z' - 1);

}

cipher += alpha;

}

System.out.println("Thông điệp sau khi mã hóa: " + cipher);

}

public static void giaiMa(){

Scanner s = new Scanner(System.in);

System.out.println("Nhập thông điệp cần giải mã: ");

String cipher = s.nextLine();

System.out.println("Nhập độ dịch chuyển: ");

int k = s.nextInt();

String decrypt = "";

char alpha;

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

alpha = cipher.charAt(i);

if(alpha >= 'a' && alpha <= 'z' || alpha >= 'A' && alpha <= 'Z'){

alpha = (char)(alpha - k);

}else if(alpha < 'a'){

alpha = (char)(alpha - 'a' + 'z' + 1);

}else if(alpha > 'Z'){

alpha = (char)(alpha - 'A' + 'Z' + 1);

}

decrypt += alpha;

}

System.out.println("Thông điệp sau khi giải mã: " + decrypt);

}

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

do{

System.out.println("====Menu====");

System.out.println("1.Mã hóa");

System.out.println("2.Giải mã");

System.out.println("Mời nhập lựa chọn: ");

n = s.nextInt();

switch(n){

case 1:

maHoa();

break;

case 2:

giaiMa();

break;

}

}while(n != 0);

}

}

Hill

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package atbm\_th1;

/\*\*

\*

\* @author ASUS

\*/

import java.util.Scanner;

public class HillCipher {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Get user-defined key matrix

System.out.println("Enter key matrix (2x2):");

int[][] key = new int[2][2];

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

key[i][j] = scanner.nextInt();

}

}

// Get plaintext message

System.out.print("Enter plaintext message: ");

String plaintext = scanner.next().toUpperCase().replaceAll("\\s", "");

// Encrypt plaintext message

String ciphertext = encrypt(plaintext, key);

System.out.println("Ciphertext: " + ciphertext);

// Decrypt ciphertext message

String decryptedText = decrypt(ciphertext, key);

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

}

private static String encrypt(String plaintext, int[][] key) {

// Step 1: Convert plaintext message to matrix of integers

int[][] plaintextMatrix = stringToMatrix(plaintext);

// Step 2: Multiply plaintext matrix with key matrix to obtain ciphertext matrix

int[][] ciphertextMatrix = matrixMultiply(key, plaintextMatrix);

// Step 3: Convert ciphertext matrix to string of characters

return matrixToString(ciphertextMatrix);

}

private static String decrypt(String ciphertext, int[][] key) {

// Step 1: Convert ciphertext message to matrix of integers

int[][] ciphertextMatrix = stringToMatrix(ciphertext);

// Step 2: Find inverse of key matrix

int[][] invKey = getInverse(key);

// Step 3: Multiply ciphertext matrix with inverse of key matrix to obtain plaintext matrix

int[][] plaintextMatrix = matrixMultiply(invKey, ciphertextMatrix);

// Step 4: Convert plaintext matrix to string of characters

return matrixToString(plaintextMatrix);

}

private static int[][] stringToMatrix(String str) {

int[][] matrix = new int[2][str.length() / 2];

int index = 0;

for (int i = 0; i < matrix[0].length; i++) {

for (int j = 0; j < matrix.length; j++) {

matrix[j][i] = str.charAt(index) - 'A';

index++;

}

}

return matrix;

}

private static String matrixToString(int[][] matrix) {

StringBuilder str = new StringBuilder();

for (int i = 0; i < matrix[0].length; i++) {

for (int j = 0; j < matrix.length; j++) {

str.append((char) (matrix[j][i] + 'A'));

}

}

return str.toString();

}

private static int[][] matrixMultiply(int[][] a, int[][] b) {

int rowsA = a.length;

int colsA = a[0].length;

int rowsB = b.length;

int colsB = b[0].length;

int[][] c = new int[rowsA][colsB];

if (colsA != rowsB) {

throw new IllegalArgumentException("Matrix sizes are not compatible for multiplication");

}

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

for (int j = 0; j < colsB; j++) {

int sum = 0;

for (int k = 0; k < colsA; k++) {

sum += a[i][k] \* b[k][j];

}

c[i][j] = sum % 26;

}

}

return c;

}

private static int[][] getInverse(int[][] key) {

int det = key[0][0] \* key[1][1] - key[0][1] \* key[1][0];

int detInv = 0;

for (int i = 0; i < 26; i++) {

if ((det \* i) % 26 == 1) {

detInv = i;

break;

}

}

int[][] adj = {{key[1][1], -key[0][1]}, {-key[1][0], key[0][0]}};

int[][] invKey = new int[2][2];

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

invKey[i][j] = ((adj[i][j] \* detInv) % 26 + 26) % 26;

}

}

return invKey;

}

}

Afine

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package atbm\_th1;

import java.util.Scanner;

/\*\*

\*

\* @author ASUS

\*/

public class Affine {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

int key1, key2, key11 = 0, ch, ct, pt, l;

char t;

int fact26[] = {1, 2, 13};

int co = 0;

char c[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', ' '};

do {

System.out.println("1. Encryption");

System.out.println("2. Decryption");

System.out.print("Enter your choice :");

ch = sc.nextInt();

switch (ch) {

case 1:

Scanner dc = new Scanner(System.in);

System.out.print("Enter PlainText : ");

String num = dc.nextLine();

num = num.toUpperCase();

l = num.length();

int k = 0;

int a[] = new int[l];

char cte[] = new char[l];

char[] num1 = num.toCharArray();

for (int j = 0; j < num1.length; j++) {

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

if (num1[j] == c[i]) {

if (num1[j] != ' ') {

a[k] = i;

} else {

a[k] = 32;

}

k = k + 1;

}

}

}

System.out.print("Enter Key 1 :");

key1 = sc.nextInt();

for (int i = 0; i < 3; i++) {

if (key1 % fact26[i] == 0) {

co++;

}

}

if (co > 1) {

System.out.println("Key is inappropriate i.e. " + key1 + " and 26 is not coprime");

break;

}

System.out.print("Enter Key 2 :");

key2 = sc.nextInt();

System.out.print("Encrypted message :");

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

if (a[i] != 32) {

ct = (a[i] \* key1 + key2) % 26;

cte[i] = c[ct];

} else {

cte[i] = (char) a[i];

}

System.out.print(cte[i]);

}

break;

case 2:

Scanner ac = new Scanner(System.in);

System.out.print("Enter CipherText : ");

num = ac.nextLine();

num = num.toUpperCase();

l = num.length();

int k1 = 0;

int a1[] = new int[l];

char cte1[] = new char[l];

char[] num2 = num.toCharArray();

for (int j = 0; j < num2.length; j++) {

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

if (num2[j] == c[i]) {

if (num2[j] != ' ') {

a1[k1] = i;

} else {

a1[k1] = 32;

}

k1 = k1 + 1;

}

}

}

System.out.print("Enter Key 1 :");

key1 = sc.nextInt();

for (int i = 0; i < 3; i++) {

if (key1 % fact26[i] == 0) {

co++;

}

}

if (co > 1) {

System.out.println("Key is inappropriate i.e. " + key1 + " and 26 is not coprime");

break;

}

System.out.print("Enter Key 2 :");

key2 = sc.nextInt();

for (int i = 0; i < 26; i++) {

key11 = i;

if ((key1 \* i) % 26 == 1) {

break;

}

}

System.out.print("Decrypted message :");

for (int i = 0; i < a1.length; i++) {

if (a1[i] != 32) {

ct = ((a1[i] - key2) \* key11) % 26;

if (ct < 0) {

ct = ct + 26;

}

cte1[i] = c[ct];

} else {

cte1[i] = (char) a1[i];

}

System.out.print(cte1[i]);

}

break;

default:

System.out.println("Wrong Choice");

}

} while (ch != 0);

}

}