ins **p1**

import java.util.Scanner;

import java.util.\*;

public class insp1{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

System.out.print("Input Text to Encrypt: ");

String pt=sc.nextLine();

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

int key=sc.nextInt();

String Encrypted=encrypt(pt,key);

System.out.println("Encrypted Text is: "+Encrypted);

String decrypted=decrypt(Encrypted,key);

System.out.println("Decrypted Text is: "+decrypted);

}

public static String encrypt(String pt,int key){

String ct="";

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

int c=pt.charAt(i);

if(Character.isUpperCase(c)){

c=(c+key % 26);

if(c>'Z'){

c=c-26;

}

}

else if(Character.isLowerCase(c)){

c=(c+key % 26);

if(c>'z'){

c=c-26;

}

}

ct+=(char)c;

}

return ct;

}

public static String decrypt(String pt,int key){

String ct="";

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

int c=pt.charAt(i);

if(Character.isUpperCase(c)){

c=(c-key % 26);

if(c<'A'){

c=c+26;

}

}

else if(Character.isLowerCase(c)){

c=(c-key % 26);

if(c<'a'){

c=c+26;

}

}

ct+=(char)c;

}

return ct;

}

}

**ins 1.2**

import java.util.Scanner;

import java.util.\*;

public class insp2{

public static void main(String[] args){

Scanner sc =new Scanner(System.in);

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

String pt=sc.nextLine();

String lower="abcdefghijklmnopqrstuvwxyz";

String upper="ABCDEFGHIJKLMNOPQRSTUVWXYZ";

ArrayList<Integer> p=new ArrayList<>();

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

p.add(i);

}

Collections.shuffle(p);

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

System.out.print(p.get(i)+" ");

}

String key="",KEY="";

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

key+=lower.charAt(p.get(i));

KEY+=upper.charAt(p.get(i));

}

String ct="";

int i,j;

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

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

if(pt.charAt(i)==lower.charAt(j)){

ct+=key.charAt(j);

break;

}

if(pt.charAt(i)==upper.charAt(j)){

ct+=KEY.charAt(j);

break;

}

}

if(j==upper.length()){

ct+=pt.charAt(i);

}

}

String dt="";

i=0;

j=0;

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

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

if(ct.charAt(i)==key.charAt(j)){

dt+=lower.charAt(j);

break;

}

if(ct.charAt(i)==KEY.charAt(j)){

dt+=upper.charAt(j);

break;

}

}

if(j==key.length()){

dt+=upper.charAt(i);

}

}

System.out.println("\n Monoalphabetic Cipher ");

System.out.println("key: "+key);

System.out.println("KEY: "+KEY);

System.out.println("PlainText: "+pt);

System.out.println("CipherText: "+ct);

System.out.println("DecryptedText: "+dt);

}

}

**ins 2.1**

import java.util.Scanner;

public class insp3{

public static void main(String[] args){

String text,key,output="",dec="";

char t,k;

int x;

Scanner sc=new Scanner(System.in);

System.out.println("Enter Text To Encrypt/Decrypt: ");

text=sc.nextLine();

System.out.println("Enter any Key of Length "+text.length()+" ");

key=sc.nextLine();

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

t=text.charAt(i);

k=key.charAt(i);

x=t^k;

output+=(char)(x+96);

}

System.out.println("Encrypted Text is: "+output);

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

t=output.charAt(i);

k=key.charAt(i);

x=t^k;

dec+=(char)(x+96);

}

System.out.println("Decrypted Text is: "+dec);

}

}

**ins 2.2**

import java.util.Scanner;

public class insp3{

public static void main(String[] args){

String text,key,output="",dec="";

char t,k;

int x;

Scanner sc=new Scanner(System.in);

System.out.println("Enter Text To Encrypt/Decrypt: ");

text=sc.nextLine();

System.out.println("Enter any Key of Length "+text.length()+" ");

key=sc.nextLine();

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

t=text.charAt(i);

k=key.charAt(i);

x=t^k;

output+=(char)(x+96);

}

System.out.println("Encrypted Text is: "+output);

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

t=output.charAt(i);

k=key.charAt(i);

x=t^k;

dec+=(char)(x+96);

}

System.out.println("Decrypted Text is: "+dec);

}

}

**ins 3.1**

import javax.crypto.\*;

import java.util.\*;

public class insprac3DES{

Cipher ecpiher,dcipher;

public static void main(String args[]){

System.out.println("Enter any String: ");

Scanner sc = new Scanner(System.in);

String input =sc.nextLine();

try{

KeyGenerator k=KeyGenerator.getInstance("DES");

SecretKey key=k.generateKey();

insprac3DES en=new insprac3DES(key);

String ct=en.encrypt(input);

String decrypted=en.decrypt(ct);

System.out.println("Original String: "+input);

System.out.println("Encrypted String: "+ct);

System.out.println("Decrypted String: "+decrypted);

}

catch(Exception e){

System.out.println(e);

}

}

public insprac3DES(SecretKey key){

try{

ecpiher=Cipher.getInstance("DES");

ecpiher.init(Cipher.ENCRYPT\_MODE,key);

dcipher=Cipher.getInstance("DES");

dcipher.init(Cipher.DECRYPT\_MODE,key);

}

catch(Exception e){

System.out.println(e);

}

}

public String encrypt(String str){

Base64.Encoder encoder=Base64.getEncoder();

try{

byte[] b=str.getBytes("UTF8");

byte[] enc=ecpiher.doFinal(b);

return new String(encoder.encode(enc),"UTF8");

}

catch(Exception e){

System.out.println(e);

}

return null;

}

public String decrypt(String str){

Base64.Decoder decoder=Base64.getDecoder();

try{

byte[] b=decoder.decode(str);

byte[] dec=dcipher.doFinal(b);

return new String(dec,"UTF8");

}

catch(Exception e){

System.out.println(e);

}

return null;

}

}

**ins 3.2**

import javax.crypto.\*;

import java.util.\*;

public class insprac3AES{

Cipher ecpiher,dcipher;

public static void main(String args[]){

System.out.println("Enter any String: ");

Scanner sc = new Scanner(System.in);

String input =sc.nextLine();

try{

KeyGenerator k=KeyGenerator.getInstance("AES");

SecretKey key=k.generateKey();

insprac3AES en=new insprac3AES(key);

String ct=en.encrypt(input);

String decrypted=en.decrypt(ct);

System.out.println("Original String: "+input);

System.out.println("Encrypted String: "+ct);

System.out.println("Decrypted String: "+decrypted);

}

catch(Exception e){

System.out.println(e);

}

}

public insprac3AES(SecretKey key){

try{

ecpiher=Cipher.getInstance("AES");

ecpiher.init(Cipher.ENCRYPT\_MODE,key);

dcipher=Cipher.getInstance("AES");

dcipher.init(Cipher.DECRYPT\_MODE,key);

}

catch(Exception e){

System.out.println(e);

}

}

public String encrypt(String str){

Base64.Encoder encoder=Base64.getEncoder();

try{

byte[] b=str.getBytes("UTF8");

byte[] enc=ecpiher.doFinal(b);

return new String(encoder.encode(enc),"UTF8");

}

catch(Exception e){

System.out.println(e);

}

return null;

}

public String decrypt(String str){

Base64.Decoder decoder=Base64.getDecoder();

try{

byte[] b=decoder.decode(str);

byte[] dec=dcipher.doFinal(b);

return new String(dec,"UTF8");

}

catch(Exception e){

System.out.println(e);

}

return null;

}

}

**ins 4**

import java.math.\*;

import java.security.\*;

public class InsP4 {

SecureRandom r;

BigInteger p,q,p1,q1,n,phi,e,d,msg,ct,pt;

public InsP4(){

r=new SecureRandom();

p=new BigInteger(512,100,r);

q=new BigInteger(512,100,r);

n=p.multiply(q);

System.out.println("Prime No P is: "+p.intValue());

System.out.println("Prime No Q is: "+q.intValue());

System.out.println("N=P\*Q is: "+n.intValue());

p1=p.subtract(new BigInteger("1"));

q1=q.subtract(new BigInteger("1"));

phi=p1.multiply(q1);

e=new BigInteger("2");

while(phi.gcd(e).intValue()>1 || e.compareTo(p1)!=-1){

e=e.add(new BigInteger("1"));

}

System.out.println("Public key is ("+n.intValue()+", "+e.intValue()+")");

d=e.modInverse(phi);

System.out.println("Private key is ("+n.intValue()+", "+d.intValue()+")");

msg=new BigInteger("3");

ct=msg.modPow(e, n);

System.out.println("Encrypted text is: "+ct.intValue());

pt=ct.modPow(d, n);

System.out.println("Decrypted text is: "+pt.intValue());

}

public static void main(String args[]){

new InsP4();

}

}

**ins 5**

import java.util.\*;

public class InsP5 {

public static void main(String args[]){

Scanner sc=new Scanner(System.in);

System.out.println("Enter a prime no q: ");

int q=sc.nextInt();

System.out.println("Enter Primitive Root alpha such that alpha<q ");

int alpha=sc.nextInt();

System.out.println("Enter value of Xa ");

int Xa=sc.nextInt();

System.out.println("Enter value of Xb ");

int Xb=sc.nextInt();

int Ya=(int) ((Math.pow(alpha,Xa))%q);

int Yb=(int) ((Math.pow(alpha,Xb))%q);

int Ka=(int) ((Math.pow(Yb,Xa))%q);

int Kb=(int) ((Math.pow(Ya,Xb))%q);

if(Ka==Kb){

System.out.println("Keys Matched!!!");

}

else{

System.out.println("Keys Dont Match!!!");

}

}

}

**ins 6**

send.java:

import java.io.FileOutputStream;

import java.io.ObjectOutputStream;

import java.security.MessageDigest;

import java.util.Scanner;

public class InsP8Send {

public static void main(String[] args) {

System.out.println("Enter the Message to be Sent: ");

try {

Scanner sc=new Scanner(System.in);

String input=sc.nextLine();

byte buffer[]= new byte[input.length()];

FileOutputStream fos = new FileOutputStream("abc.txt");

ObjectOutputStream oos=new ObjectOutputStream(fos);

MessageDigest md=MessageDigest.getInstance("MD5");

buffer=input.getBytes();

md.update(buffer);

oos.writeObject(input);

oos.writeObject(md.digest());

System.out.println("Message Sent Successfully!!");

}

catch(Exception e) {

e.printStackTrace();

}

}

}

receive.java:

import java.io.\*;

import java.security.\*;

public class InsP8Receive {

public static void main(String[] args) {

byte dig[]=new byte[1024];

try {

FileInputStream fis=new FileInputStream("abc.txt");

ObjectInputStream ois=new ObjectInputStream(fis);

Object obj=ois.readObject();

String data=(String)obj;

System.out.println("Received Data: "+data);

obj=ois.readObject();

dig=(byte[])obj;

MessageDigest md=MessageDigest.getInstance("MD5");

md.update(data.getBytes());

if(MessageDigest.isEqual(md.digest(), dig)) {

System.out.println(data+" Retrieved Successfully !!");

}

ois.close();

}

catch(Exception e) {

System.out.println("Message is Corrupted !!");

}

}

}

**ins 7**

import java.util.Scanner;

import java.security.SecureRandom;

import javax.crypto.Mac;

import javax.crypto.spec.SecretKeySpec;

public class InsP7 {

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

Scanner sc=new Scanner(System.in);

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

String data=sc.nextLine();

SecureRandom rnd=new SecureRandom();

byte[] k=new byte[100];

rnd.nextBytes(k);

SecretKeySpec key =new SecretKeySpec(k,"HMACSHA1");

System.out.println("Key: "+key.getEncoded());

Mac m= Mac.getInstance("HmacSHA1");

m.init(key);

m.update(data.getBytes());

byte[] result=m.doFinal();

System.out.println("Signature using HMACSHA1: "+new String(result));

}

}