Blowfish Algorithm:

import javax.Crypto.Cipher;

import javax.Crypto.keyGenerator;

import javax.Crypto.SecretKey;

import javax.swing.Joptoionpane;

public class Blowfishcipher

{

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

{

keyGenerator keygen=keyGenerator.getInstance("blowfish");

SecretKey seckey=keygen.generatekey();

cipher cipher=cipher.getInstance("blowfish");

cipher.init(cipher.ENCRYPT-MODE,seckey);

String inputtext=Joptionpane.showinputDialog("Input your message");

byte[] encrypted=Cipher.doFinal(inputtext.getBytes());

Cipher.init(Cipher.DECRYPT-MODE,seckey);

byte[] decrypted=Cipher.doFinal(encypted);

Joptionpane.showmessageDialog(Joptionpane.getRootFrame(),"encrypted text:"+new String(encrypted)+"decrypted text:"+new String(decrypted));

System.exit(0);

}

}

RC4 Algorithm:

import javax.crypto.Cipher;

import javax.crypto.KeyGnerator;

import javax.crypto.SecretKey;

import javax.swing.JOptionPane;

public class RC4 {

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

KeyGenerator keygen=KeyGenerator.getInstance("blowfish");

SecretKey seckey=keygen.generateKey();

Cipher c=Cipher.getInstance("blowfish");

c.init(c.ENCRYPT\_MODE,seckey);

String text="Hello World";

byte[] encrypted=c.doFinal(text.getBytes());

c.init(c.DECRYPT\_MODE,seckey);

byte[] decrypted=c.doFinal(encrypted);

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

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

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

}

}

RSA Algorithm:

import java.math.\*;

import java.util.Random;

import java.util.Scanner;

public class RSA{

static Scanner sc=new Scanner(System.in);

public static void main(String[] args)

{

System.out.print("Enter a prime number:");

BigInteger p=sc.nextBigInteger();

System.out.print("Enter another prime number:");

BigInteger q=sc.nextBigInteger();

BigInteger n=p.multiply(q);

BigInteger n2=q.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));

BigInteger e=generate(n2);

BigInteger d=e.modInverse(n2);

System.out.println("Encryption keys are:" +e+ "," +n);

System.out.println("Decryption keys are:" +d+ "," +n);

}

private static BigInteger generate(BigInteger FioFn)

{

int y,int GCD;

Random x=new Random();

BigInteger e;

do{

y=x.nextInt(FioFn,intvalue()-1);

String l=Integer.toString(y);

e=new BigInteger(l);

BigInteger gcd=FioFn.gcd(e);

int GCD=gcd.intvalue();

}while(y<=2 || GCD==1)

return e;

}

}

SHA1 Algorithm:

import java.security.\*;

public class SHA1 {

public static void main(String[] args) {

try

{

MessageDigest md=MessageDigest.getInstance("SHA1");

System.out.println("Message digest object into:");

System.out.println("Algorithm="+md.getAlgorithm());

System.out.println("Provider="+md.getProvider());

System.out.println("to string="+md.toString());

String input="";

md.update(input.getBytes());

byte[] output=md.digest();

System.out.println(md.getAlgorithm()+"("+input+")");

System.out.println(" "+bytesToHex(output));

input="abcdefghijklmnopqrstuvwxyz";

md.update(input.getBytes());

output=md.digest();

System.out.println();

System.out.println(md.getAlgorithm()+"("+input+")");

System.out.println(" "+bytesToHex(output));

}

catch(Exception e){

System.out.println("Exception:"+e);

}

}

public static String bytesToHex(byte[] b){

char hexdigit[]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};

StringBuffer buf=new StringBuffer();

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

buf.append(hexdigit[(b[j]>>4)&0x0f]);

buf.append(hexdigit[b[j]&0x0f]);

}

return buf.toString();

}

}

MD5 Algorithm:

import java.security.\*;

public class MD5 {

public static void main(String[] args) {

try

{

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

System.out.println("Message digest object into:");

System.out.println("Algorithm="+md.getAlgorithm());

System.out.println("Provider="+md.getProvider());

System.out.println("to string="+md.toString());

String input="";

md.update(input.getBytes());

byte[] output=md.digest();

System.out.println(md.getAlgorithm()+"("+input+")");

System.out.println(" "+bytesToHex(output));

input="abcdefghijklmnopqrstuvwxyz";

md.update(input.getBytes());

output=md.digest();

System.out.println();

System.out.println(md.getAlgorithm()+"("+input+")");

System.out.println(" "+bytesToHex(output));

}

catch(Exception e){

System.out.println("Exception:"+e);

}

}

public static String bytesToHex(byte[] b){

char hexdigit[]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};

StringBuffer buf=new StringBuffer();

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

buf.append(hexdigit[(b[j]>>4)&0x0f]);

buf.append(hexdigit[b[j]&0x0f]);

}

return buf.toString();

}

}

Diffie Hellman

import java.util.\*;

class diff

{

public static void main(String[] args)

{

Scanner sc= new Scanner(System.in);

System.out.println("Enter modulo(p):");

int p=sc.nextInt();

System.out.println("Enter 1st secret number(1st person):");

int a=sc.nextInt();

System.out.println("Enter primitive root of "+p);

int g=sc.nextInt();

System.out.println("Enter 2nd secret number(2nd person):");

int b=sc.nextInt();

int A=(int)Math.pow(g,a)%p;

int B=(int)Math.pow(g,b)%p;

int S\_A=(int)Math.pow(B,a)%p;

int S\_B=(int)Math.pow(A,b)%p;

if(S\_A==S\_B)

{

System.out.println("1st person and 2nd person can communicate with each other.");

System.out.println("They share a secret number:"+S\_A);

}

else

{

System.out.println("1st person and 2nd person cannot communicate with each other."); } } }