**Source Code:**

import java.lang.Integer.\*;

import java.math.BigInteger;

import java.util.Random;

import java.util.Scanner;

import java.lang.String.\*;

public class RsaAlgorithm {

static int arrayposition=0;

static String Dbase27string="";

static long d=0;//Private Key

public static boolean extendedEuclidGCD(long a, long b)

{

long x=0,y = 1, lastx = 1, lasty = 0, temp;

while (b != 0)

{

long q = a / b;

long r = a % b;

a = b;

b = r;

temp = x;

x = lastx - q \* x;

lastx = temp;

temp = y;

y = lasty - q \* y;

lasty = temp;

}

if (a==1)

{

d=(lasty);

// System.out.println("value od decry key is : "+d);

return true;

}

else

{

return false;

}

}

//method to efficiently calculate powers

public static double powers(double b,double ew)

{

//System.out.println("base b:"+b);

//System.out.println("Exponent ew:"+ew);

double r=1;

while(ew>0){

if(ew%2==0){

b=b\*b;

//System.out.println("inside if : "+b);

ew=ew/2;

}

else {

r=r\*b;

//System.out.println("inside else : "+r);

ew=ew-1;

}

}

//System.out.println("power is :"+r);

return r;

}

//method to convert Decrypted message to base 27

public static void base27Conversion(long decimal)

{

Dbase27string=decimal%27+" "+Dbase27string;

//System.out.println("enters into base27conversion method:"+Dbase27string);

arrayposition++;

decimal=decimal/27;

if(decimal<27)

{

Dbase27string=Long.toString(decimal)+" "+Dbase27string;

}

else

{

base27Conversion(decimal);

}

Dbase27string= Dbase27string.trim();

}

public static boolean isPrime(long n, int iteration)

{

/\*\* base case \*\*/

if (n == 0 || n == 1)

return false;

/\*\* base case - 2 is prime \*\*/

if (n == 2)

return true;

/\*\* an even number other than 2 is composite \*\*/

if (n % 2 == 0)

return false;

long s = n - 1;

while (s % 2 == 0)

s /= 2;

Random rand = new Random();

for (int i = 0; i < iteration; i++)

{

long r = Math.abs(rand.nextLong());

long a = r % (n - 1) + 1, temp = s;

long mod = modPow(a, temp, n);

while (temp != n - 1 && mod != 1 && mod != n - 1)

{

mod = mulMod(mod, mod, n);

temp \*= 2;

}

if (mod != n - 1 && temp % 2 == 0)

return false;

}

return true;

}

/\*\* Function to calculate (a ^ b) % c \*\*/

public static long modPow(long a, long b, long c)

{

long res = 1;

for (int i = 0; i < b; i++)

{

res \*= a;

res %= c;

}

return res % c;

}

/\*\* Function to calculate (a \* b) % c \*\*/

public static long mulMod(long a, long b, long mod)

{

return BigInteger.valueOf(a).multiply(BigInteger.valueOf(b)).mod(BigInteger.valueOf(mod)).longValue();

}

public static int[] phi12(String a1){

//String inpText= a1.getTEXT();

char[] alpha=a1.toCharArray();

int b;

int[] num= new int[a1.length()];

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

int a = Character.getNumericValue(alpha[i]);

if(a==-1)

{

b=0;

//System.out.println("the bearcatii num for null is : "+b);

}

else

{

b=a-9;

//System.out.println("the bearcatii num for "+ alpha[i]+" is "+b);

}

if(i<=a1.length())

{

num[i]=b;

}

}

return num;

}

public static int toDecimal(int[] numb){

int t=0;

int sum=0;

int initiatepower=0;

for(int i=numb.length-1;i>=0;i--)

{

sum+=numb[i]\*powers(27,initiatepower);

initiatepower++;

//System.out.println("print each addition term"+numb[i]\*(powers(27,i)));

}

/\*for (int j = 0; j <numb.length; j++) {

//System.out.println("the value of j and numb[j] "+j+" and "+numb[j]);

t=(int) (Math.pow(27,numb.length-1-j )\*numb[j]);

sum=sum+t;

//System.out.println("T is : "+t);

}\*/

return sum;

}

static long Modkey(long base,long exp, long modN) {

long[] c=new long[100];

long count=1,next\_val=base;

int a=0,j=0;

int[] binry = BinaryFormat((int) exp);

long[] new\_var=new long[100];

while(count<exp){

c[j]=next\_val%modN;

next\_val=c[j]\*c[j];

j=j+1;

count=count\*2;

}

for (int l=0;l<binry.length;l++){

if(binry[l]==1){

new\_var[a]=c[l];

a++;

}

}

for(int s=1;s<=99;s=s+1){

if(new\_var[s]>0){

new\_var[0] =(new\_var[0]\*new\_var[s])%modN;

}

}

return new\_var[0]%modN;

}

static int[] BinaryFormat(int number){

int binary[] = new int[100];

int index = 0;

while(number > 0){

binary[index++] = number%2;

number = number/2;

}

return binary;

}

public static void main(String[] args) {

long p=5279; //prime1 inti

long q=22691; //prime2 init

long n; // n=p\*q

long phi; //

long e;

//int bitlength = 1024;

boolean reinput=false;

long C;//cipher text

long decrypted=0;

Scanner sc = new Scanner(System.in);

System.out.println("RSA Algorithm Implementation in JAVA");

Random rand = new Random();

boolean a=true;

while(a){

p =rand.nextInt(100000);

q = rand.nextInt(100000);

if(isPrime(p, 5) && isPrime(q, 5)) {

a=false; }

}

System.out.println("Randomly generated primes values are((p & q)) : "+p+" & "+q);

n=p\*q;

phi=(p-1)\*(q-1);

System.out.println("The value of n is (n) : "+n);

System.out.println("The value of tortion function phi(n) is : "+phi);

//Taking e input from user

do

{

System.out.println("Please Input Public Key (e) which is co-prime to \""+phi+"\" and in limit 1<e<phi : ");

e = sc.nextLong();

boolean isgcd1=extendedEuclidGCD(phi,e);

reinput=false;

//Check if gcd(e,phi)is 1,otherwise ask for input again

if(!isgcd1)

{

reinput=true;

System.out.println("Entered input is not relatively prime to n");

}

}

while(reinput);

Scanner s1 = new Scanner(System.in );

System.out.println("Enter the TEXT message to be Encrypted (M):");

String a1=s1.nextLine();

//Rsapojo rs1=new Rsapojo();

//rs1.setTEXT(a1);

int[] number=phi12(a1);

int decim=toDecimal(number);

System.out.println("M in BEARCATII : "+decim);

C=Modkey(decim,e ,n);

System.out.println("ENCRYPTED MESSAGE (C) : "+C);

//Decrypted message

if(d<0){

d=phi+d;

}

decrypted=Modkey(C,d,n);

System.out.println("DECRYPTED MESSAGE (D) : "+decrypted);

//Converting Decrypted message back to base27

base27Conversion(decrypted);

System.out.println("after decrypted : "+Dbase27string);

String[] splitbase27D=Dbase27string.split("\\s");

String plaintext="";

for(String s:splitbase27D)

{

if(s.equals("0"))

{

plaintext+=" ";

}

else

{

double tempint= (Integer.valueOf(s))+64.0;

char tempch=(char) tempint;

plaintext+=tempch;

}

}System.out.println("The Decrypted TEXT (P) : "+plaintext);

}

}

**OUTPUT1:**

RSA Algorithm Implementation in JAVA

Randomly generated primes values are((p & q)) : 4679 & 35339

The value of n is (n) : 165351181

The value of tortion function phi(n) is : 165311164

Please Input Public Key (e) which is co-prime to "165311164" and in limit 1<e<phi :

165311163

Enter the TEXT message to be Encrypted (M):

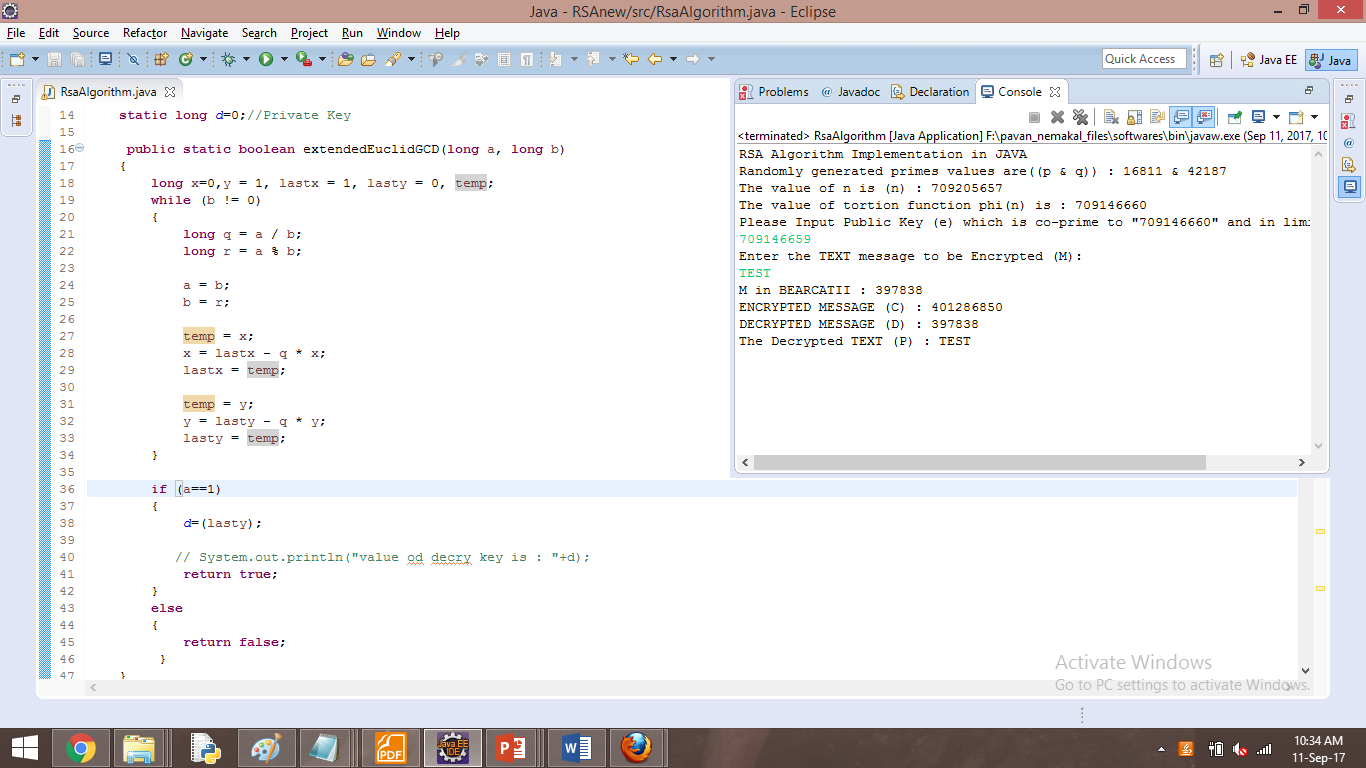
TEST

M in BEARCATII : 397838

ENCRYPTED MESSAGE (C) : 64071411

DECRYPTED MESSAGE (D) : 397838

The Decrypted TEXT (P) : TEST



**OUTPUT2:**

RSA Algorithm Implementation in JAVA

Randomly generated primes values are((p & q)): 5279 & 22691

The value of n is (n): 119785789

The value of tortion function phi (n) is: 119757820

Please Input Public Key (e) which is co-prime to "119757820" and in limit 1<e<phi:

119757819

Enter the TEXT message to be Encrypted (M):

H A I

M in BEARCATII: 4252266

ENCRYPTED MESSAGE (C): 97376853

DECRYPTED MESSAGE (D): 4252266

The Decrypted TEXT (P): H A I

