|  |  |
| --- | --- |
| Ex.No.4 | **IMPLEMENTATION OF RSA** |

|  |
| --- |
| **AIM:** |

To simulate the working of RSA in Virtual lab environment and to implement the same in Java/Python

|  |
| --- |
| **THEORY:** |

**One-way function**

A Trapdoor one-way function is a one-way function which might be described as a function for which evaluation in one direction is straightforward, while computation in the reverse direction is far more difficult.

**Key generation**

The difficulty of determining a private key from an RSA public key is equivalent to factoring the modulus n. An attacker thus cannot use knowledge of an RSA public key to determine an RSA private key unless he can factor n. It is also a one-way function, going from p & q values to modulus n is easy but reverse is not possible.

Select two large primes, p and q.

Calculate n=p\*q.

Select number e which must be greater than 1 and less than (p − 1)(q − 1) and such that e and (p – 1)(q – 1) are coprime.

(n, e) - RSA public key

d ≡ e-1 mod (p − 1)(q − 1) – RSA private key

**RSA Encryption**

The sender whose public key is (n, e) represents the plaintext as a series of numbers less than n.

To encrypt the first plaintext P, which is a number modulo n. The encryption process is simple mathematical step as

**c = pe mod n**

**RSA Decryption**

The receiver of public-key pair (n, e) with a ciphertext c raises c to the power of his private key d. The result modulo n will be the plaintext P

**p = cd mod n**

|  |
| --- |
| **ALGORITHM:** |

**Key generation**

1. Select two large primes, p and q.
2. Calculate n=p\*q.
3. Select number e which must be greater than 1 and less than (p − 1)(q − 1) and such that e and (p – 1)(q – 1) are coprime.
4. (n, e) gives the public key pair.
5. d ≡ e-1 mod (p − 1)(q − 1) gives the private key of the recipient.

**RSA Encryption**

1. Make the plain text as p.
2. With the public key pair (n,e) calculate cipher text where c is

**c = pe mod n**

1. Send the cipher text to the receiver.

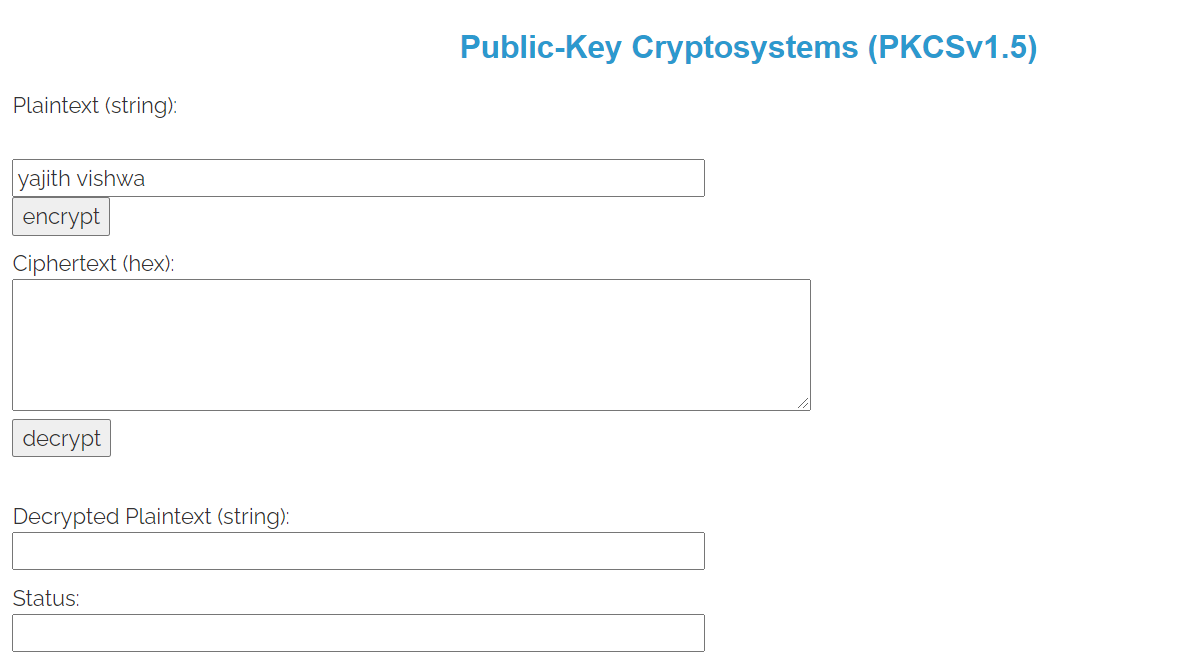
**RSA Decryption**

1. Receive the cipher text as c.
2. With the public key pair (n,e) and with private key d calculate p

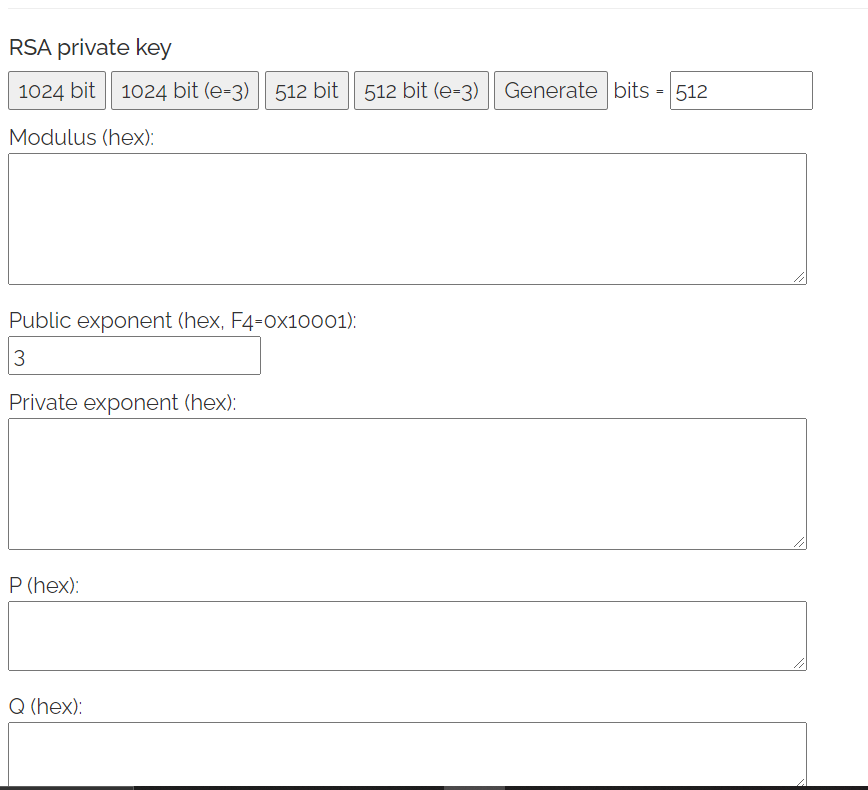
**p = cd mod n**

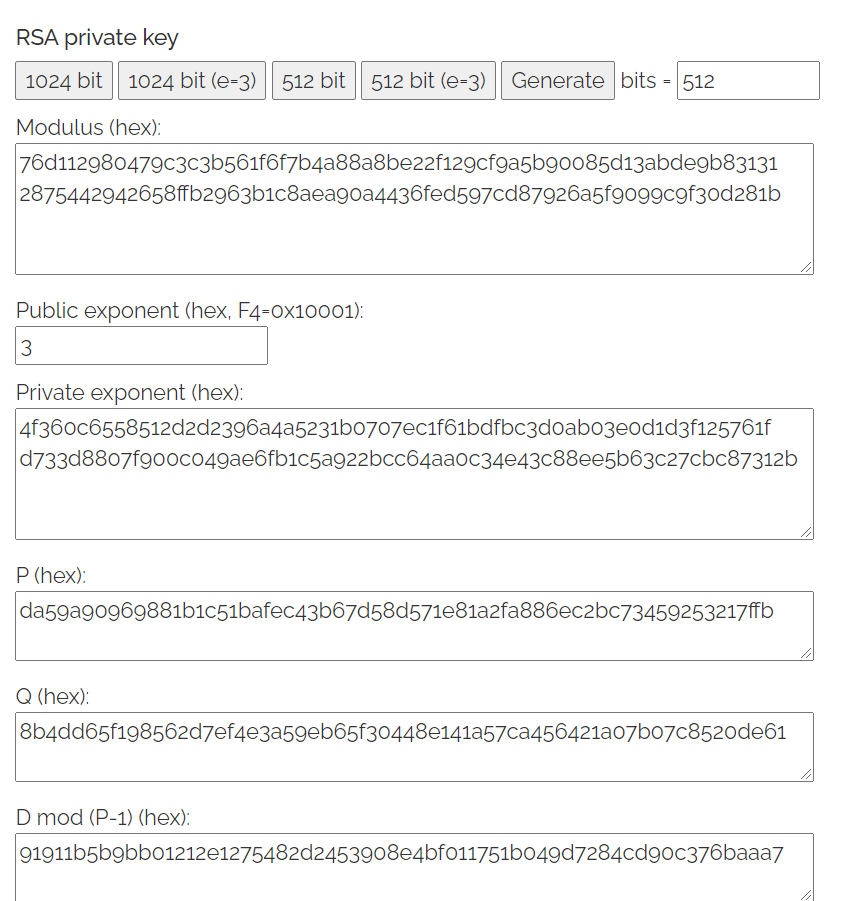
|  |
| --- |
| **Screen Shots of simulation in Virtual labs** |

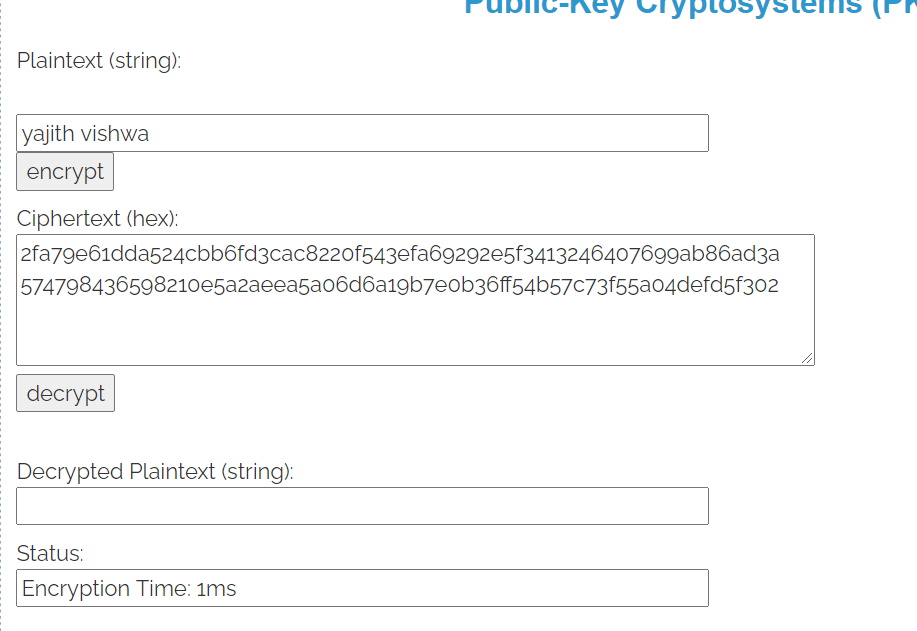
1. Enter the plain text .

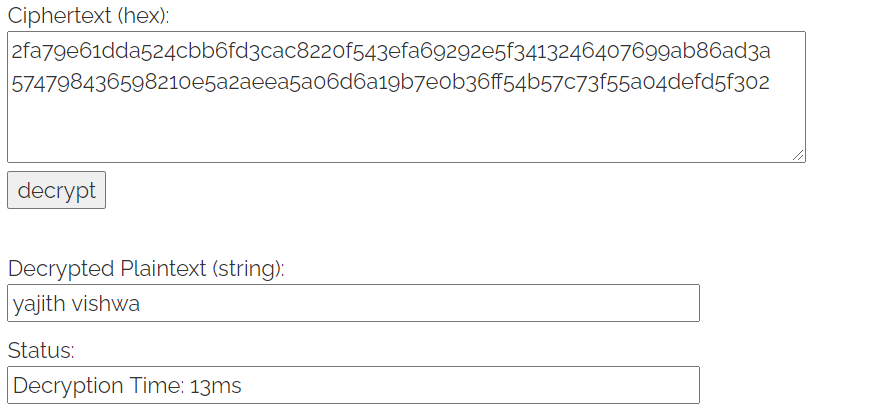


1. Generate RSA private key by clicking any one bit key



3. Generate private key.

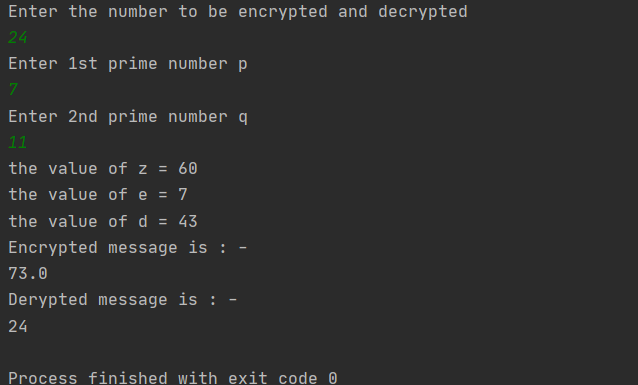
4. Click Encrypt and will get cipher text.

5. Click Decrypt to get back the plain text.

|  |
| --- |
| **Coding** |

package com.yajith.hillcipher;  
import java.math.BigDecimal;  
import java.math.BigInteger;  
import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 int p,q,n,z,d=0,e,i;  
 System.*out*.println("Enter the number to be encrypted and decrypted");  
 int msg=sc.nextInt();  
 double c;  
 BigInteger msgback;  
 System.*out*.println("Enter 1st prime number p");  
 p=sc.nextInt();  
 System.*out*.println("Enter 2nd prime number q");  
 q=sc.nextInt();  
  
 n=p\*q;  
 z=(p-1)\*(q-1);  
 System.*out*.println("the value of z = "+z);  
 for(e=2;e<z;e++)  
 {  
 if(*gcd*(e,z)==1) // e is from public key pair  
 {  
 break;  
 }  
 }  
 System.*out*.println("the value of e = "+e);  
 for(i=0;i<=9;i++)  
 {  
 int x=1+(i\*z);  
 if(x%e==0) //d is for private key  
 {  
 d=x/e;  
 break;  
 }  
 }  
 System.*out*.println("the value of d = "+d);  
 c=(Math.*pow*(msg,e))%n;  
 System.*out*.println("Encrypted message is : -");  
 System.*out*.println(c);  
 BigInteger N = BigInteger.*valueOf*(n);  
 BigInteger C = BigDecimal.*valueOf*(c).toBigInteger();  
 msgback = (C.pow(d)).mod(N);  
 System.*out*.println("Derypted message is : -");  
 System.*out*.println(msgback);  
  
 }  
 static int gcd(int e, int z)  
 {  
 if(e==0)  
 return z;  
 else  
 return *gcd*(z%e,e);  
 }  
}

|  |
| --- |
| **SCREEN SHOTS:** |



|  |
| --- |
| **RESULT:** |

Thus, the simulation of RSA algorithm in done in virtual lab and the Java program for the algorithm is also implemented and the results are obtained.

|  |
| --- |
| **Evaluation** |

|  |  |  |
| --- | --- | --- |
| Parameter | Max Marks | Marks Obtained |
| Uniqueness of the Code | 15 |  |
| Completion of experiment on time | 5 |  |
| Documentation | 5 |  |
| Simulation in Vlabs | 5 |  |
| Total | 30 |  |
| Signature of the faculty with Date |  |  |