**Aim:**

Cryptography is playing a major role in data protection in applications running in a network environment. It allows people to do business electronically without worries of deceit and deception in addition to ensuring the integrity of the message and authenticity of the sender. It has become more critical to our day-to-day life because thousands of people interact electronically every day; through e-mail, e-commerce, ATM machines, cellular phones, etc. This geometric increase of information transmitted electronically has made increased reliance on cryptography and authentication by users.Despite the fact that secured communication has existed for centuries, the key management problem has prevented it from commonplace application. The development of public-key cryptography has enabled large-scale network of users that can communicate securely with one another even if they had never communicated before [6-8]. This paper considers a Public Key encryption method using RSA algorithm that will convert the information to a form not understandable by the intruder therefore protecting unauthorized users from having access to the information even if they are able to break into the system.

**Algorithm:**

Begin

1. Choose two prime numbers p and q.

2. Compute n = p\*q.

3. Calculate phi = (p-1) \* (q-1).

4. Choose an integer e such that 1 < e < phi(n) and gcd(e, phi(n)) = 1; i.e., e and phi(n) are coprime.

5. Calculate d as d ≡ e−1 (mod phi(n)); here, d is the modular multiplicative inverse of e modulo phi(n).

6. For encryption, c = me mod n, where m = original message.

7. For decryption, m = c d mod n.

End

**Program:**

#include<iostream>

#include<math.h>

using namespace std;

// find gcd

int gcd(int a, int b) {

int t;

while(1) {

t= a%b;

if(t==0)

return b;

a = b;

b= t;

}

}

int main() {

//2 random prime numbers

double p ;

double q ;

cout<<"Enter prime p->";

cin>>p;

cout<<"Enter prime q->";

cin>>q;

double n=p\*q;//calculate n

double track;

double phi= (p-1)\*(q-1);//calculate phi

//public key

//e stands for encrypt

double e=7;

//for checking that 1 < e < phi(n) and gcd(e, phi(n)) = 1; i.e., e and phi(n) are coprime.

while(e<phi) {

track = gcd(e,phi);

if(track==1)

break;

else

e++;

}//private key

//d stands for decrypt

//choosing d such that it satisfies d\*e = 1 mod phi

double d1=1/e;

double d=fmod(d1,phi);

double message = 9;

double c = pow(message,e); //encrypt the message

double m = pow(c,d);

c=fmod(c,n);

m=fmod(m,n);

cout<<"Original Message = "<<message;

cout<<"\n"<<"p ="<<p;

cout<<"\n"<<"q ="<<q;

cout<<"\n"<<"n = pq = "<<n;

cout<<"\n"<<"phi = "<<phi;

cout<<"\n"<<"e = "<<e;

cout<<"\n"<<"d = "<<d;

cout<<"\n"<<"Encrypted message = "<<c;

cout<<"\n"<<"Decrypted message = "<<m;

return 0;

}

**Sample Input:**

p-> 17

q-> 11

**Output:**

Enter prime p->17

Enter prime q->11

Original Message = 9

p =17

q =11

n = pq = 187

phi = 160

e = 7

d = 0.142857

Encrypted message = 70

Decrypted message = 9





