**Problem Statement:**

Write a program for simple RSA algorithm to encrypt and decrypt the data.

**Theory:**

The RSA algorithm is named after Ron Rivest, Adi Shamir and Len Adleman, who invented it in 1977.The RSA algorithm can be used for both public key encryption and digital signatures. Its security is based on the difficulty of factoring large integers.

**Algorithm (*computing public key and the private key)***

1. Choose two large prime numbers, p and q
2. Compute n = p x q and z = (p - 1) x (q - 1).
3. Choose a number that is relatively prime to **z** and call it **e**

i.e. gcd (z, e) =1

1. Find **d** such that e x d = 1 mod z.

Public key = {e, n} and private key = {d, n}.

**Algorithm (*encryption and decryption*)**

Let P be plaintext (an Integer) to be encrypted, 0 ≤ P ≤n

1. To encrypt compute C = Pe (mod n)
2. To decrypt C, compute P = Cd (mod n).

RSA is based on the following key property:

**Pde (mod n) = P (mod n)**

Modular arithmetic involving large numbers can be simplified by using the following property.

**(ab) mod n = ((a mod n )(b mod n )) mod**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File Name : RSA.c

\*

\* Description : Simple RSA algorithm to encrypt and decrypt the data.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "stdafx.h"

#include<stdio.h>

#include<math.h>

#include<string.h>

#define MAX\_SZ 100

unsigned int min(unsigned int x, unsigned int y)

{

return(x<y?x:y);

}

unsigned int max(unsigned int x, unsigned int y)

{

return(x>y?x:y);

}

unsigned int gcd(unsigned int x, unsigned int y)

{

if(x==y)

return(x);

else

return(gcd(min(x,y),max(x,y)-min(x,y)));

}

unsigned int xpowy\_modn(unsigned int x,unsigned int y, unsigned int n)

{

unsigned int r=1;

while(y>0)

{

if((int)(y%2)==1)

r=(r\*x)%n;

x=(x\*x)%n;

y=y/2;

}

return(r);

}

unsigned int find\_encrypt\_key(unsigned int z)

{

unsigned int e;

do

{

printf("\n Enter a number e that is relatively prime to z and < z:");

scanf("%d",&e);

if(e>=z)

continue;

}while(gcd(e,z)!=1);

return(e);

}

unsigned int find\_decrypt\_key(unsigned int e,unsigned int z)

{

unsigned int d;

for(d=2;d<z;++d)

{

if(((long int)(e\*d)%(long int)z)==1)

break;

}

return(d);

}

int main()

{

//long double

unsigned int plain\_txt[MAX\_SZ], cipher\_txt[MAX\_SZ];

unsigned int p,q,z,n,e,d;

char msg[MAX\_SZ];

int i;

read:

do

{

printf("\n Enter two large prime numbers p and q:");

scanf("%d%d",&p,&q);

}while(p==q);

n=p\*q;

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

printf("\n n=%d,z=%d",n,z);

if(n < 120)

{

printf("\n\nPlease keep n >= 122");

goto read;

}

e=find\_encrypt\_key(z);

d=find\_decrypt\_key(e,z);

printf("\nPublic key ={%d,%d}",e,n);

printf("\nPrivate key ={%d,%d}",d,n);

printf("\nEnter a string consisting only letters(a-zA-Z) :");

scanf("%s", msg);

for(i=0;i<strlen(msg);i++)

{

plain\_txt[i]=msg[i];

printf("\n%c = %d:",plain\_txt[i],plain\_txt[i]);

}

// find cipher text

printf("\n\nCipher Text:\n");

for(i=0; i<strlen(msg); i++)

{

cipher\_txt[i] = xpowy\_modn(plain\_txt[i], e,n);

printf("\n%d=%c",cipher\_txt[i],cipher\_txt[i]);

}

printf("\n\nPlain Text:\n ");

for(i=0; i<strlen(msg); i++)

{

plain\_txt[i] = xpowy\_modn(cipher\_txt[i],d,n);

printf("\n%c = %d",plain\_txt[i],plain\_txt[i]);

}

return(0);

}