**RSA**

**Code:**

#include <iostream> #include <time.h> using namespace std;

// Function to check if the number is prime bool isPrime(int num) {

if (num <= 1) { return false;

}

for (int i = 2; i \* i <= num; ++i) {

if (num % i == 0) { return false;

}

}

return true;

}

// Function to generate random prime number int generateRandomPrime(int range) { int randomNum = rand() % range + 1;

while (!isPrime(randomNum)) {

randomNum = rand() % range + 1;

}

return randomNum;

}

// Function to return GCD int gcd(int a, int b){

if(b==0){

return a;

}

return gcd(b, a%b);

}

// Main Function int main(){

srand(time(0));

int p = generateRandomPrime(100); int q = generateRandomPrime(100);

cout << "Step 1:" << endl; cout << "Random Prime Number p: " << p << endl; cout << "Random Prime Number q: " << q << endl; cout << endl;

cout << "Step 2:" << endl; int n = p\*q; cout << "Modulus of Encryption and Decryption: " << n << endl;

cout << endl; cout << "Step 3: " << endl; int phiN = (p-1)\*(q-1); int e=0; for(int i=2; i<phiN ; i++){

if(gcd(i,phiN) == 1){

e = i; break;

}

}

cout << "Value of e: " << e << endl;

cout << endl;

cout << "Step 4:" << endl; cout << "Public Key <e, n>: " << "<" << e << ", " << n << ">" << endl; cout << endl;

int m; // Plain Text cout << "Enter plain text message (m) less than (n): "; cin >> m;

cout << "Step 5:" << endl; int encrypted = (m^e)%n; cout << "Encrypted Text of (m): " << encrypted << endl;

cout << endl; cout << "Step 6:" << endl; int d=(e^(phiN))%phiN; cout << "Private Key <d, n>: " << "<" << d << ", " << n << ">" << endl;

cout << endl;

cout << "Step 7:" << endl; int m2 = (encrypted^d)%n; cout << "Decrypted Message: " << m2 << endl;

}

**Output:**

Step 1:

Random Prime Number p: 59

Random Prime Number q: 41

Step 2:

Modulus of Encryption and Decryption: 2419

Step 3: Value of e: 3

Step 4:

Public Key <e, n>: <3, 2419>

Enter plain text message (m) less than (n): 2000 Step 5:

Encrypted Text of (m): 2003

Step 6:

Private Key <d, n>: <3, 2419>

Step 7:

Decrypted Message: 2000