```cpp
/*
--------------------------------RSA-------------------------------
*/

//header-------------------------------------------------------------

#include <iostream>
#include <string.h>
#include <NTL/ZZ.h>
#include <sstream>
#include <cstdlib>
#include <vector>
#include <ctime>

using namespace std;
using namespace NTL;

class rsa
{
public:
  rsa();
  rsa(ZZ E, ZZ N, ZZ phi_N);
  // Retornar claves ---------------------
  ZZ return_n();
  ZZ return_phi_n();
  ZZ return_e();
  ZZ return_d();
  //-------------------------------------
  ZZ mod(ZZ x, ZZ y);
  ZZ MCD(ZZ x, ZZ y);
  int mod(int x, int y);
  ZZ exp_mod(ZZ x, ZZ y, ZZ z);
  ZZ inversa(ZZ x, ZZ y);
  string convertir(ZZ x);
  string cifrar(string mensaje);
  void claves_pq();
  //string descifrar(string mensaje_cifrado);

  string alfabeto = "ABCDEFGHIJKLMNOPQRSTUVWXYZ,.-(
)abcdefghijklmnopqrstuvwxyz<>*1234567890";

private:
  ZZ n, phi_n;
  ZZ e, d;
};

rsa::rsa()
{
  n = 0;
  phi_n = 0;
  e = 0;
  d = 0;
  claves_pq();
}

int rsa::mod(int x, int y)
{
  int z;
  z = x - x / y * y;
  return z;
}

ZZ rsa::mod(ZZ x, ZZ y)
{
  ZZ z;
  z = x - x / y * y;
```

```
  return z;
}

ZZ rsa::MCD(ZZ x, ZZ y)
{
  if(y == 0)
  {
    return x;
  }
  return MCD(y, mod(x, y));
}

rsa::rsa(ZZ E, ZZ N, ZZ phi_N)
{
  n = N;
  phi_n = phi_N;
  e = E;
  d = inversa(E, phi_N);
}

void rsa::claves_pq()
{
  vector <long long> nros_primos;

  nros_primos.push_back(2);

  for(int i = 0; i <= 1000000; i++) // ------> Números primos de 5 cifras.
  {
    nros_primos.push_back(i);
  }

  int x = 1;
  int y = 2;

  while(y < nros_primos.size() && nros_primos[x] * nros_primos[x] < 1000000)
  {
    if(nros_primos[y] % nros_primos[x] == 0)
    {
      nros_primos.erase(nros_primos.begin() + y);
    }
    else
    {
      y = y + 1;
    }
    if(y == nros_primos.size())
    {
      x = x + 1;
      y = x + 1;
    }
  }
  srand(time(NULL));
  int indice1;
  int indice2;

  do
  {
    indice1 = mod(rand(), nros_primos.size());
  } while (indice1 < 9592);

  do
  {
    indice2 = mod(rand(), nros_primos.size());
  } while (indice2 < 9592);

  ZZ p, q;
  p = ZZ(nros_primos[indice1]);
```

```
132     q = ZZ(nros_primos[indice2]);
133     n = p * q;
134     phi_n = (p - 1) * (q - 1);
135
136     do
137     {
138       e = rand();
139       e = mod(e, phi_n);
140     } while(MCD(e, phi_n) != 1);
141   }
142
143   ZZ rsa::return_n()
144   {
145     return n;
146   }
147
148   ZZ rsa::return_phi_n()
149   {
150     return phi_n;
151   }
152
153   ZZ rsa::return_e()
154   {
155     return e;
156   }
157
158   ZZ rsa:: return_d()
159   {
160     return d;
161   }
162
163   string rsa::convertir(ZZ x)
164   {
165     stringstream bufer;
166     bufer << x;
167     return bufer.str();
168   }
169
170   ZZ rsa::inversa(ZZ x, ZZ y)
171   {
172     ZZ t;
173     ZZ q;
174
175     ZZ s1;
176     s1 = conv<ZZ>(1);
177     ZZ s2;
178     s2 = conv<ZZ>(0);
179     ZZ r = y;
180     while(y > 0)
181     {
182       q = x / y;
183       t = x - q * y;
184       x = y;
185       y = t;
186       t = s1 - q * s2;
187       s1 = s2;
188       s2 = t;
189     }
190     if(s1 < 0)
191     {
192       s1 = s1 + r;
193     }
194     return s1;
195   }
196
197   string rsa::cifrar(string mensaje)
```

```
198 {
199   string mensaje_cifrado;
200   ZZ auxiliar;
201   ZZ expo_mod;
202   ZZ p;
203 
204   for(int i = 0; i < mensaje.length(); i++)
205   {
206     expo_mod = 1;
207     p = alfabeto.find(mensaje[i]);
208     auxiliar = e;
209 
210     while (auxiliar > 0)
211     {
212       if(mod(auxiliar, ZZ(2)) == 1)
213       {
214         expo_mod = mod(expo_mod * p, n);
215       }
216       p = mod(p * p, n);
217       auxiliar = auxiliar / 2;
218     }
219     mensaje_cifrado = mensaje_cifrado + convertir(expo_mod);
220   }
221   return mensaje_cifrado;
222 }
223 
224 
225 //main-------------------------------------------------------------
226 
227 #include <iostream>
228 #include <string.h>
229 #include <NTL/ZZ.h>
230 #include <sstream>
231 #include <cstdlib>
232 #include <vector>
233 #include <ctime>
234 
235 #include "RSA.h"
236 
237 int main()
238 {
239   string mensaje;
240   string mensaje_cifrado;
241   rsa cypher;
242   rsa rsa(cypher.return_e(), cypher.return_n(), cypher.return_phi_n());
243   return 0;
244 
245   cout << "Ingrese el mensaje: ";
246   getline(cin, mensaje);
247 
248   cout << "n: " << cypher.return_n() << endl;
249   cout << "phi n: " << cypher.return_phi_n() << endl;
250   cout << "e: " << cypher.return_e() << endl;
251   cout << "d: " << rsa.return_d() << endl;
252 
253   mensaje_cifrado = rsa.cifrar(mensaje);
254 
255   cout << "Mensaje cifrado:" << endl;
256   cout << mensaje_cifrado;
257 }
```