CNS Assignment 2 : RSA Algorithm
Name : Siddhesh Bajad Rollno : BTITA4

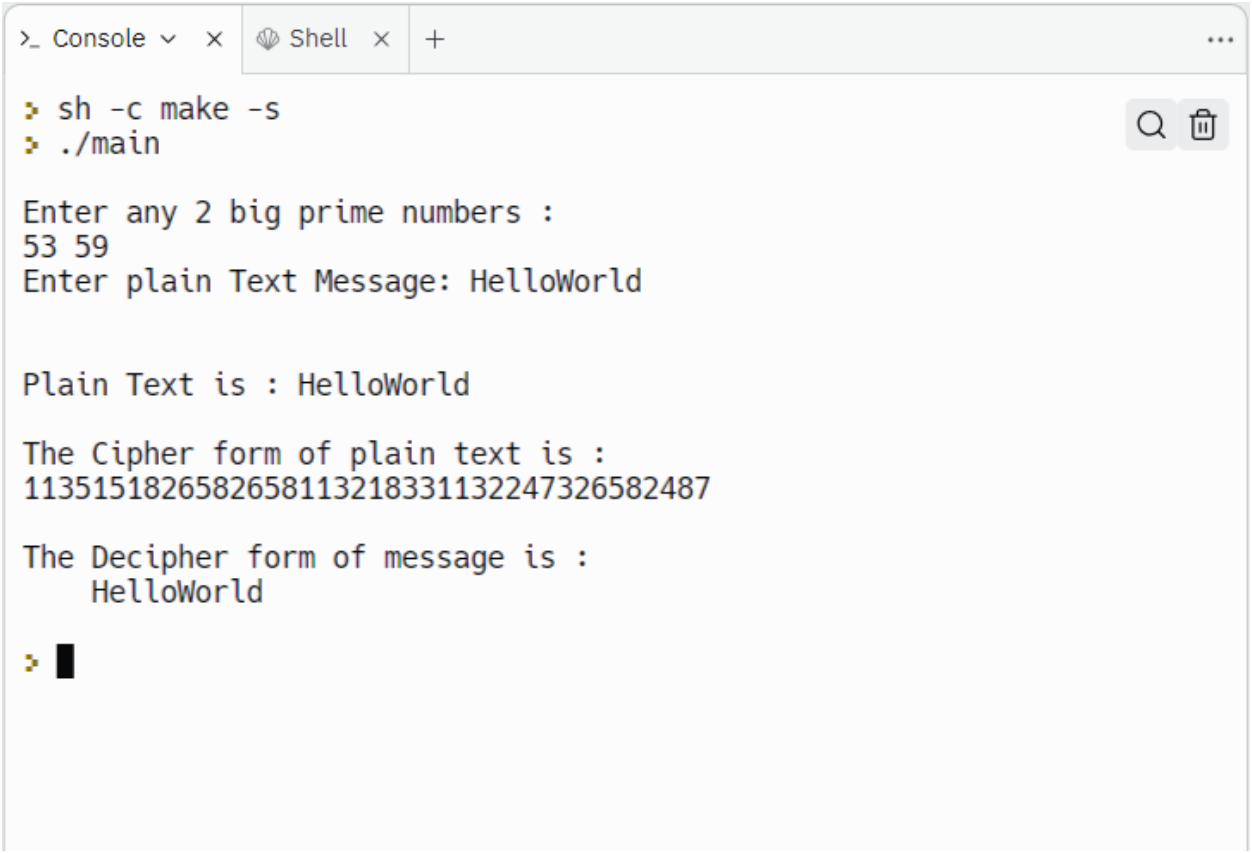
Code :

```
#include <bits/stdc++.h>
using namespace std;

int publicKey;
int privateKey;
int n;

void setKeys() {
  int p, q;
  cout << "\nEnter any 2 big prime numbers : " << endl;
  cin >> p >> q;

  n = p * q;
  int phi = (p - 1) * (q - 1);
  int e = 2;

  while (true) {
    if (__gcd(e, phi) == 1)
      break;
    else
      e++;
  }
  publicKey = e; // Found the public key

  int d = 2;
  while (true) {
    if ((d * e) % phi == 1) {
      break;
    }
    d++;
  }
  privateKey = d; // Found the private Key
}

int encrypt(long long int msg) {
  int e = publicKey;
  long long int cipher = 1;
  while (e--) { // using formula for encryption c = m^e mod n
```

```
    cipher *= msg;
    cipher %= n;
  }
  return cipher;
}

long long int decrypt(int encoded) {
  int d = privateKey;
  long long int plain = 1;
  while (d--) { // using formula for decryption m = c^d mod n
    plain *= encoded;
    plain %= n;
  }
  return plain;
}

vector<int> encoder(string &plain) {
  vector<int> cipher;
  for (char ch : plain) {
    cipher.push_back(encrypt(int(ch))); // calling encrypt function
  }
  return cipher;
}

string decoder(vector<int> &coded) {
  string decipher;
  for (auto num : coded) {
    decipher += decrypt(num);
  }
  return decipher;
}

int main() {
  setKeys();

  string plain;
  cout << "Enter plain Text Message: ";
  cin >> plain;
  cout << "\n\nPlain Text is : " << plain << endl;

  cout << "\nThe Cipher form of plain text is : \n";
  vector<int> coded = encoder(plain);
  for (int a : coded)
    cout << a;
```

```
  cout << endl;

  cout << "\nThe Decipher form of message is : \n\t";
  cout << decoder(coded) << "\n" << endl;
}
```

Output :



CNS Assignment 3
Name : Siddhesh Bajad Rollno : BTITA4

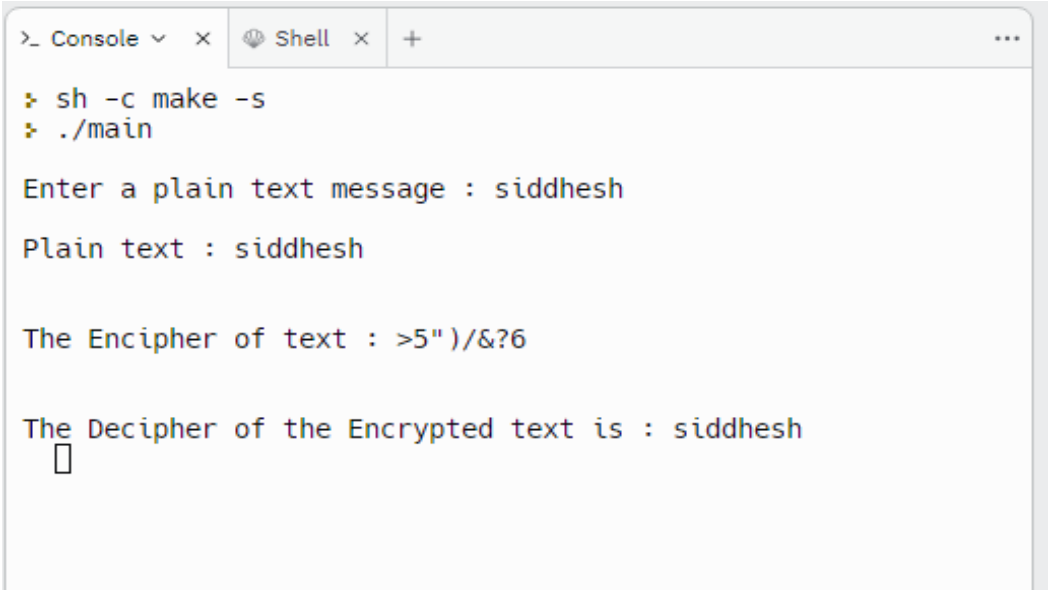
Aim : Design and Implement your own encryption/ decryption algorithm using any programming language.

Code :

```
#include <bits/stdc++.h>
using namespace std;

string key;
// vector<int> hash;
void xor_op(string &str) {
  for(int i=0;i<str.size();i++) {
    str[i] = str[i] ^ key[i];
  }
}

string encrypt(string &plain, vector<int> &hash)
{
  // string key = plain;
  int n = plain.size();
  string cipher = "";
  for(int i=0;i<plain.size();i++)
    {
      if(i %2 == 0) {
        plain[i] = (plain[i] - 97 + 3) % 26 + 65;
      }
      else plain[i] = abs(plain[i] -97 - 3) % 26 + 65;
      // cout << plain[i] << " ";
    }

  xor_op(plain);
  return plain;
}

string decrypt(string &encoded, vector<int> &hash) {
  int n = encoded.size();
  string decipher = "";
  xor_op(encoded);

  for(int i=0;i<encoded.size();i++)
    {
```

```
      if(i %2 == 0) {
        encoded[i] = abs(encoded[i] - 65 - 3) % 26 + 97;
      }
      else encoded[i] = (encoded[i] -65 + 3) % 26 + 97;
      // cout << encoded[i] << " ";
    }

  return encoded;
}

string reverse(string text)
{
  int i=0, j= text.size()-1;
  while(i < j) {
    swap(text[i], text[j]);
    i++;j--;
  }
  return text;
}


int main()
{
  string plain;
  cout << "\nEnter a plain text message : ";
  getline(cin, plain);
  int n = plain.size();
  vector<int> hash(n, 0);
  key = reverse(plain);


  cout << "\nPlain text : " << plain << endl;
  string encoded = encrypt(plain, hash);
  cout << "\n\nThe Encipher of text : " << encoded << endl;
  string decoded = decrypt(plain, hash);
  cout << "\n\nThe Decipher of the Encrypted text is : " << decoded << endl;
  return 0;
}
```

Output :


Console
Shell
sh -c make -s
./main
Enter a plain text message : siddhesh
Plain text : siddhesh
The Encipher of text : >5")/&?6
The Decipher of the Encrypted text is : siddhesh

CNS Assignment 4
Name : Siddhesh Bajad Rollno : BTITA4

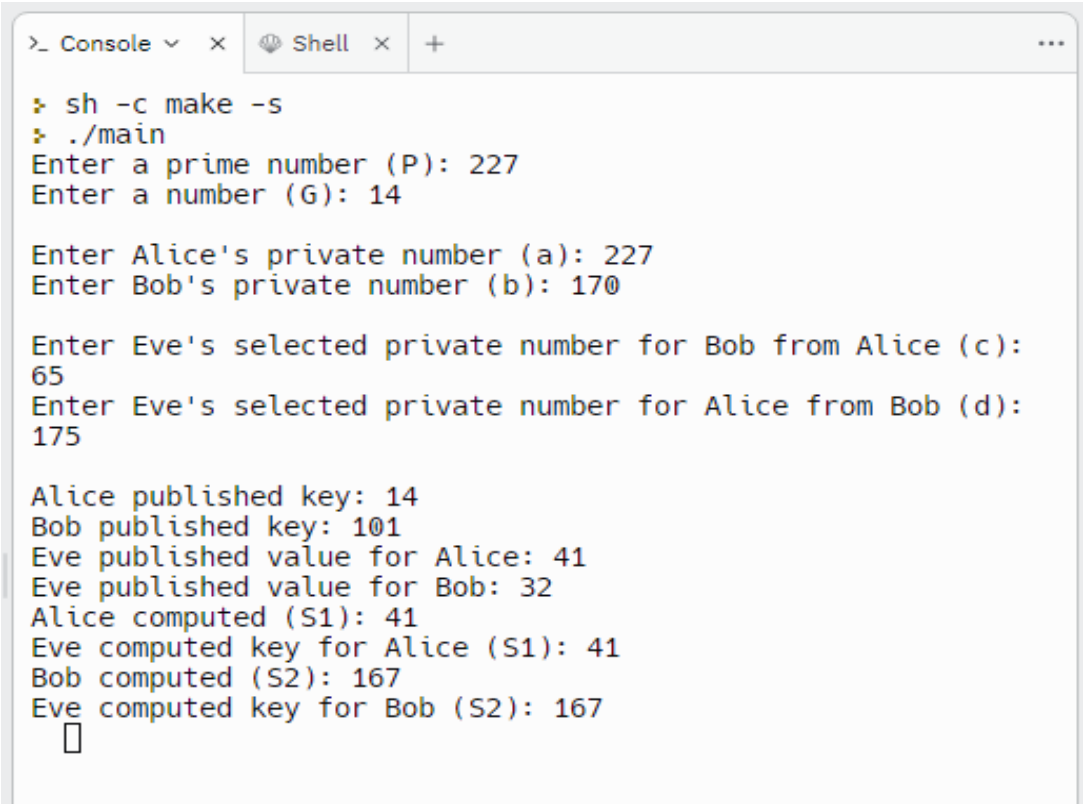
Deffie Hellman Man in the Middle Attack

Code :

```
#include <iostream>

using namespace std;

long long int modular_pow(long long int base, long long int exponent, long long int modulus) {
	if (modulus == 1)
	return 0;

	long long int result = 1;
	base = base % modulus;

	while (exponent > 0) {
	if (exponent % 2 == 1)
	result = (result * base) % modulus;

	exponent = exponent >> 1; // Equivalent to exponent /= 2
	base = (base * base) % modulus;
	}

	return result;
}


int main() {
	long long int p, g, a, b, c, d, x, y, xe, ye, ka, kb, kea, keb;

	cout << "Enter a prime number (P): ";
	cin >> p;
	cout << "Enter a number (G): ";
	cin >> g;

	cout<<endl;

	cout << "Enter Alice's private number (a): ";
	cin >> a;

	cout << "Enter Bob's private number (b): ";
	cin >> b;

	cout<<endl;
```

```
    cout << "Enter Eve's selected private number for Bob from Alice (c): ";
    cin >> c;
    cout << "Enter Eve's selected private number for Alice from Bob (d): ";
    cin >> d;

    cout<<endl;

    x = modular_pow(g, a, p);
    y = modular_pow(g, b, p);
    cout << "Alice published key: " << x << endl;
    cout << "Bob published key: " << y << endl;

    xe = modular_pow(g, c, p);
    ye = modular_pow(g, d, p);

    cout << "Eve published value for Alice: " << xe << endl;
    cout << "Eve published value for Bob: " << ye << endl;

    ka = modular_pow(xe, a, p);
    kea = modular_pow(x, c, p);

    kb = modular_pow(ye, b, p);
    keb = modular_pow(y, d, p);

    cout << "Alice computed (S1): " << ka << endl;
    cout << "Eve computed key for Alice (S1): " << kea << endl;
    cout << "Bob computed (S2): " << kb << endl;
    cout << "Eve computed key for Bob (S2): " << keb << endl;

    return 0;
}
```

Output :


Console
Shell
sh -c make -s
./main
Enter a prime number (P): 227
Enter a number (G): 14
Enter Alice's private number (a): 227
Enter Bob's private number (b): 170
Enter Eve's selected private number for Bob from Alice (c):
65
Enter Eve's selected private number for Alice from Bob (d):
175
Alice published key: 14
Bob published key: 101
Eve published value for Alice: 41
Eve published value for Bob: 32
Alice computed (S1): 41
Eve computed key for Alice (S1): 41
Bob computed (S2): 167
Eve computed key for Bob (S2): 167