## Name: Patel Pratham Alpeshbhai. Roll No.: CE100

20CEUOS052

Lab: 3

```
#include <bits/stdc++.h>
#define ll long long int
#define MAX PRIME 100000
using namespace std;
string intToBinaryRev(ll n){
    string s="";
    while (n) {
        s = (char)(n%2 + '0') + s;
       n/=2;
    return s;
vector<ll> extendedEuclidean(ll a, ll n){
    ll n save = n;
    ll r1=n, r2=a;
    ll d1=1, d2=0;
    ll t1=0, t2=1;
   ll q, r, d, t;
    while (r2!=0) {
       q=r1/r2;
       r=r1-(r2*q);
       r1=r2;
        r2=r;
        d=d1-(d2*q);
        d1=d2;
        d2=d;
```

```
t=t1-(t2*q);
        t1=t2;
        t2=t;
    d1=(d1%n save+n save)%n save;
    t1=(t1%n save+n save)%n save;
    return {r1,d1,t1};
ll getMultiplicativeInverse(ll a, ll n){
    vector<ll> res = extendedEuclidean(a, n);
    if(res[0]==1)
       return res[2];
    return -1;
11 GCD(ll a, ll b) {
    while(b){
        11 tem=a;
       b = tem%b;
    return a;
ll power(ll a,ll n,ll q){
   ll res = 1;
    n%=q;
    string N = intToBinaryRev(n);
    for (int i = 0; i < N.size(); ++i)
        res = (res * res)%q;
        if(N[i]=='1')
            res = (res * a)%q;
    return res;
11 getRandKeyHelper(11 phi n, int MAX ITER = 1000){
    return 13;
```

```
if(!MAX ITER) return -1;
    ll e = rand()%(phi n-1)+1;
    if(GCD(e, phi n)!=1) return getRandKeyHelper(phi n, MAX ITER-1);
11 RSAProgramEncryption(ll M, ll e, ll n) {
    return power(M, e, n);
11 RSAProgramDecryption(11 C, 11 d, 11 n) {
    return power(C, d, n);
void testSquareAndMultiply(){
    int n;
    cin>>n;
        cout<<"power("<<i<<", "<<n-1<<", "<<n<<") = "<<power(i, n-1,</pre>
n) << endl;
void testRSA() {
    int P = 191;
    int Q = 197;
    ll n = P*Q;
    ll phi n = (P-1)*(Q-1);
    /* RSA public key is e */
    11 e = getRandKeyHelper(phi n);
    11 d = getMultiplicativeInverse(e, phi n);
    //cout<<d<<"\n";
    11 M;
    cin>>M;
    cout<<"Plain text : "<<M<<endl;</pre>
    11 C = RSAProgramEncryption(M, e, n);
    cout<<"Encrypted text : "<<C<<endl;</pre>
    11 m = RSAProgramDecryption(C, d, n);
```

```
void testRSAString() {
    string s;
    cin>>s;
    while (!isPrime(P)) P=rand()%MAX PRIME;
    int Q = rand()%MAX PRIME;
    while (!isPrime(Q)) Q=rand()%MAX PRIME;
    ll n = P*Q;
    ll phi n = (P-1)*(Q-1);
    /* RSA public key is e */
    11 e = getRandKeyHelper(phi n);
    /* RSA private key */
    ll d = getMultiplicativeInverse(e, phi_n);
    vector<ll> C(s.size());
    for(int i=0; i<s.size(); ++i){</pre>
        C[i] = RSAProgramEncryption(s[i], e, n);
    cout << endl;
int main(){
    testRSA();
    return 0;
```