**Student Number: 202135760**

**Student Name: 김현지**

**Problem 13**

* **Screenshots**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명**

* **Code**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

#include<string.h>

#include<math.h>

char Message[100000]; //original text

int m[100000];

typedef struct extendedEuclidValue {

int b;

int x;

int y;

}EEV;

//find gcd

int gcd(int a, int b) {

if (b == 0)

return a;

gcd(b, a % b);

}

EEV extendedEuclid(int a, int b) {

if (b == 0) {

EEV eev = { a, 1, 0 };

return eev;

}

else {

EEV eev = extendedEuclid(b, a % b);

EEV eev2 = { eev.b, eev.y, eev.x - (a / b) \* eev.y };

return eev2;

}

}

//compute m^d % n

int RSA(int m, int d, int n) {//m: message, d:private/public key, n:modulus

int bit[20] = { 0, };

unsigned long long temp = 1;

int a[20]; // m^2^0~19 mod n

//Convert exponent 'd' to binary (store reverse order)

for (int i = 0; d > 0; i++) {

bit[i] = d % 2;

d = d / 2;

}

//m^2^a mod n : save the value in advance

a[0] = (unsigned long long)pow(m, 1) % n;

for (int i = 1; i < 10; i++) {

a[i] = (a[i - 1] \* a[i - 1]) % n;

}

//Multiply the pre-calculated value only when the bit is 1

for (int i = 0; i < 10; i++) {

if (bit[i] == 1) {

temp = temp \* a[i];

}

}

return temp % n;

}

void main() {

int p = 17; //prime1 324497

int q = 11; //prime2 295759

int N = 0; //p\*q

int T = 0; //euler totient theorem

int e = 0; //<T and relatively prime to T | public key

int d = 0; //private key

int c[100000] = { 0, };

int m2[100000] = { 0, };

//input

printf("Enter the content you want to encrypt: ");

gets\_s(Message);

for (int i = 0; i < strlen(Message); i++) {

m[i] = Message[i];

}

//calculate key value

N = p \* q; //N

T = (p - 1) \* (q - 1); //T

//e

srand(time(0));

do {

e = rand() % (T - 2) + 2; // 1<e<T

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

//d

EEV eev = extendedEuclid(T, e);

if (eev.y < 0)

d = eev.y + T;

else

d = eev.y;

//printf("p: %d, q:%d, N:%d, T:%d, e:%d, d:%d\n", p, q, N, T, e, d);

printf("\npublic key(%d, %d)\n", e, N);

printf("private key (%d, %d)\n\n", d, N);

// encryption

printf("msg : %s\n", Message);

printf("encryption : ");

char C[1000] = { 0, };

for (int i = 0; i < strlen(Message); i++) {

c[i] = RSA(m[i], e, N);

printf("%c", c[i]);

}

printf("\n");

// decryption

printf("decryption: ");

for (int i = 0; i < strlen(Message); i++) {

m2[i] = RSA(c[i], d, N);

printf("%c", m2[i]);

}

printf("\n");

}