**МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**

(национальный исследовательский университет)»

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра № 304 «Вычислительные машины, системы и сети»

Защита информации

Отчет по лабораторной работе № 2

RSA-1024 — RSA-65536

Выполнил студент группы М3О-409Б-20:

Мокшин И. Н.

Проверил, Ведьманов И. С.

Москва 2023 г.

Оглавление

[1. Код программы 3](#_Toc153353651)

[2. Результат работы 6](#_Toc153353652)

# Код программы

#pragma once

#include <iostream>

#include <vector>

#include <mpirxx.h>

#include <sstream>

using namespace std;

// Function to generate a random prime number of given bit length

mpz\_class generate\_prime(int bit\_length) {

mpz\_class random\_number;

// Initialize a random number generator

gmp\_randclass r(gmp\_randinit\_default);

r.seed(time(NULL));

// Generate a random number of the given bit length

random\_number = r.get\_z\_bits(bit\_length);

// Find the next prime number

mpz\_nextprime(random\_number.get\_mpz\_t(), random\_number.get\_mpz\_t());

// Clear the random number if it's not prime

while (!mpz\_probab\_prime\_p(random\_number.get\_mpz\_t(), 25)) {

random\_number = r.get\_z\_bits(bit\_length);

mpz\_nextprime(random\_number.get\_mpz\_t(), random\_number.get\_mpz\_t());

}

return random\_number;

}

struct key\_struct {

mpz\_class e;

mpz\_class n;

mpz\_class d;

};

// Function to calculate the public key for RSA encryption

key\_struct get\_keys(mpz\_class p, mpz\_class q) {

key\_struct keys;

mpz\_class n = p \* q;

// Calculate ϕ(n) = (p - 1) \* (q - 1)

mpz\_class phi\_n = (p - 1) \* (q - 1);

// Calculate e such that 1 < e < ϕ(n) and gcd(e, ϕ(n)) = 1

gmp\_randclass r(gmp\_randinit\_default);

mpz\_class e, gcd;

do {

e = r.get\_z\_range(phi\_n - 1) + 1;

mpz\_gcd(gcd.get\_mpz\_t(), e.get\_mpz\_t(), phi\_n.get\_mpz\_t());

} while (gcd != 1);

mpz\_class d;

mpz\_invert(d.get\_mpz\_t(), e.get\_mpz\_t(), phi\_n.get\_mpz\_t());

keys.d = d;

keys.e = e;

keys.n = n;

return keys;

}

// Function to encrypt a message using RSA encryption

mpz\_class encrypt(mpz\_class message, key\_struct keys) {

// Calculate c = m^e (mod n)

mpz\_class c;

mpz\_powm(c.get\_mpz\_t(), message.get\_mpz\_t(), keys.e.get\_mpz\_t(), keys.n.get\_mpz\_t());

return c;

}

// Function to decrypt a message using RSA encryption

mpz\_class decrypt(mpz\_class ciphertext, key\_struct keys) {

// Calculate m = c^d (mod n)

mpz\_class message;

mpz\_powm(message.get\_mpz\_t(), ciphertext.get\_mpz\_t(), keys.d.get\_mpz\_t(), keys.n.get\_mpz\_t());

return message;

}

// Encodes a string to an mpz\_class

std::vector<mpz\_class> encode(string str) {

std::vector<mpz\_class> result;

for (int i = 0; i < str.length(); i++) {

mpz\_class cur;

mpz\_import(cur.get\_mpz\_t(), 1, 1, 1, 0, 0, &str[i]);

result.push\_back(cur);

}

return result;

}

// Decodes an mpz\_class to a string

string decode(std::vector<mpz\_class> message) {

std::stringstream ss;

for (auto& letter : message) // access by reference to avoid copying

{

ss << (char)letter.get\_ui();

}

return ss.str();

}

int lr2(std::string text, int bits = 1024) {

// Generate two random prime numbers of 1024 bits each

mpz\_class p, q, n, e, d, phi;

p = generate\_prime(bits);

q = generate\_prime(bits);

// Calculate the public and private keys

auto keys = get\_keys(p, q);

// Encode

std::vector<mpz\_class> message = encode(text);

// Encrypt the message

std::vector<mpz\_class> ciphertext;

for (auto letter : message)

{

ciphertext.push\_back(encrypt(letter, keys));

}

// Encrypt the message

std::vector<mpz\_class> decrypted\_message;

for (auto letter : ciphertext)

{

decrypted\_message.push\_back(decrypt(letter, keys));

}

// Print

auto decoded\_orig = decode(message);

auto decoded\_after\_ci = decode(decrypted\_message);

std::cout << "Message: " << decoded\_orig << std::endl;

std::cout << "Decrypted message: " << decoded\_after\_ci << std::endl;

return 0;

}

# Результат работы

**RSA-1024:**

Message: This is very long test string.

Decrypted message: This is very long test string.

Сообщение было успешно зашифровано/расшифровано. Процесс занял 15 секунд.

**RSA- 65536:**

Message: This is very long test string.

Decrypted message: This is very long test string.

Сообщение было успешно зашифровано/расшифровано. Процесс занял 14 минут.