Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информационных технологий автоматизированных систем

Отчет по лабораторной работе №5

по курсу «ИТОКБ»

на тему: «Асимметричное шифрование и электронная цифровая подпись»

Вариант 4

|  |  |
| --- | --- |
| Выполнил магистрант группы 025941: | Колесников В.Г. |
| Проверил: | Боброва Т.С. |

МИНСК

2021

**Задание 1:** Разработать программное обеспечение, реализующее функции генерации секретного и открытого ключей, шифрования и цифровой подписи для алгоритма RSA. Обмен входными и выходными данными должен осуществляться через файлы:

- файл открытого ключа;

- файл секретного ключа;

- файл исходного сообщения;

- файл зашифрованного сообщения.

Код программы приведен в конце лабораторной работы.

Результат работы программы поиска примитивного элемента и генерации секретов приведен на рисунке 1.1.

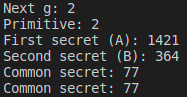


Рисунок 1.1 — Результат поиска примитивного элемента и генерации секретов

**Код программы:**

#include <iostream>

#include <string>

#include <bits/stdc++.h>

#include <cmath>

using namespace std;

// To reverse numbers bits

string reverseBits(string source)

{

string reversed = source;

reverse(reversed.begin(), reversed.end());

return reversed;

}

string getNumberBits(int number)

{

bool foundOne = false;

string bits = "";

for (int i = (sizeof(number) \* \_\_CHAR\_BIT\_\_ - 1); i >= 0; --i)

{

if (((number >> i) & 1) == 1)

{

foundOne = true;

}

if (foundOne)

{

bits.append(((number >> i) & 1) == 1 ? "1" : "0");

}

}

return bits;

}

const int getModulo(int g, int power, int P)

{

int modulo = 1;

while (power)

{

if (power & 1)

{

modulo = (int)(modulo \* 1ll \* g % P);

power--;

}

else

{

g = (int)(g \* 1ll \* g % P);

power >>= 1;

}

}

return modulo;

}

const unsigned long getPrimitive(const unsigned int P)

{

unsigned int g = 1;

bool isAllPrimitive = false;

while (true)

{

g++;

cout << "Next g: " << g << endl;

if (P % g == 0 || g % P == 0)

{

continue;

}

isAllPrimitive = true;

for (int i = 2; i <= g; i++)

{

if (g % i == 0 && P % i == 0)

{

isAllPrimitive = false;

break;

}

}

if (!isAllPrimitive)

{

continue;

}

for (int power = 2; power < P - 1; power++)

{

const int modulo = getModulo(g, power, P);

if (modulo == 0)

{

isAllPrimitive = false;

break;

}

}

if (isAllPrimitive)

{

break;

}

}

return g;

}

int main(int argc, char \*\*argv)

{

setlocale(LC\_ALL, "");

srand(time(NULL));

const unsigned int P = 1877;

// - 1

// а

const unsigned long g = getPrimitive(P);

cout << "Primitive: " << g << endl;

// б

const unsigned int a = 13771877;

const unsigned int b = 18771377;

const unsigned int A = getModulo(g, a, P);

const unsigned int B = getModulo(g, b, P);

cout << "First secret (A): " << A << endl;

cout << "Second secret (B): " << B << endl;

const unsigned int Ab = getModulo(A, b, P);

const unsigned int Ba = getModulo(B, a, P);

cout << "Common secret (Ab): " << Ab << endl;

cout << "Common secret (Ba): " << Ba << endl;

return 0;

}