МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

ОТЧЕТ

по лабораторной работе №5

«Асинхронное шифрование»

по дисциплине

«Методы и средства защиты информации»

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ Капранов С. Н.\_\_

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сухоруков В.А.\_\_\_

(подпись) (фамилия, и.,о.)

\_\_\_\_\_\_19-ВМ\_\_\_\_\_\_\_\_

(шифр группы)

Работа защищена «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2022

# Задание №8.1

Реализовать алгоритм шифрования данных «RSA».

# Описание алгоритма.

*Генерация ключей.*

1. Выбираются 2 простых числа p и q
2. Вычисляется их произведение n
3. Вычисляется функция Эйлера по формуле f(n) = (p-1)\*(q-1)
4. Выбирается открытая экспонента e, взаимно простая со значением функции f(n)
5. Вычисляется число d (секретная экспонента), мультипликативно обратное числу е по модулю f(n). Это число должно удовлетворять сравнению:



1. Пара e и n – есть открытый ключ, пара b и n – закрытый ключ

*Шифрование.*

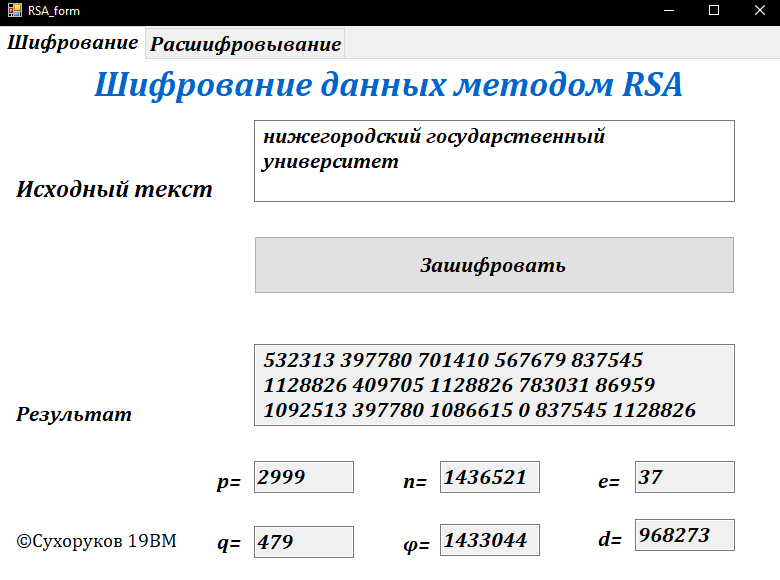
1. Берем значение символа из открытого текста и вычисляем новое значение по формуле 
2. Повторяем для каждого символа

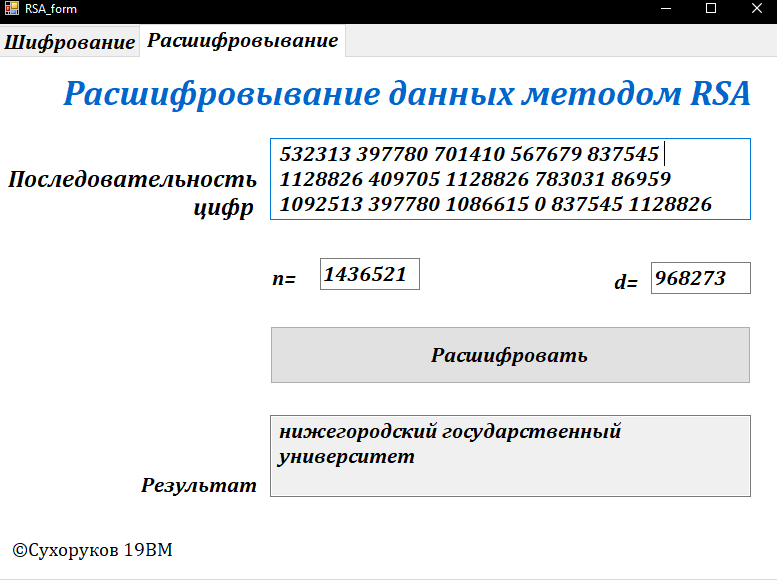
*Расшифровка.*

1. Берем значение символа из зашифрованного сообщения и вычисляем расшифрованное значение по формуле 
2. Повторяем для каждого символа.

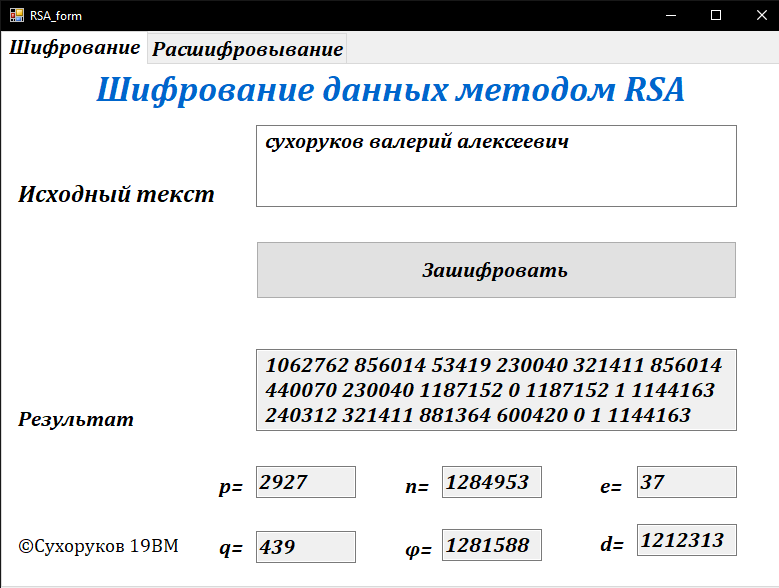
# Примеры шифрования и расшифровывания

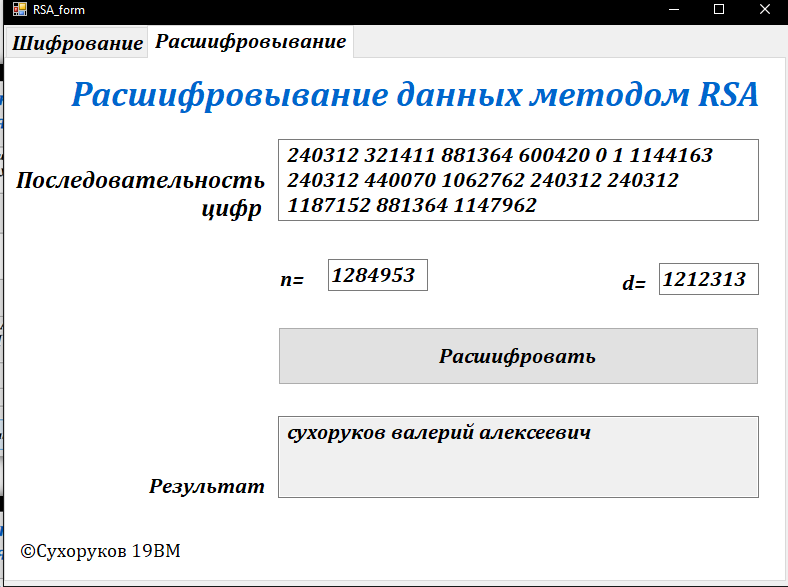
* Нижегородский государственный университет





* Сухоруков Валерий Алексеевич





# Текст программы

* Функция вызываемая при нажатии на кнопку "Зашифровать" в форме.

//--------------------------------------------------------------

//Функция, вызываемая при нажатии на кнопку "Зашифровать" в форме.

//Устанавливает в текстовое поле 3 результат функции шифрования.

System::Void LR5UI::RSA\_form::button1\_Click(System::Object^ sender, System::EventArgs^ e){

int e1, d, n, f, p, q;

string text = "";

string out = "";

srand(time(0));

text = marshal\_as<string>(textBox1->Text);

p = genPrime();

q = genPrime();

n = p \* q;

f = (p - 1) \* (q - 1);

e1 = genCoprime(f);

d = gend(f, e1);

encrypt(out, text, e1, n);

textBox3->Text = marshal\_as<System::String^>(out);

textBox4->Text= marshal\_as<System::String^>(to\_string(p));

textBox5->Text = marshal\_as<System::String^>(to\_string(q));

textBox6->Text = marshal\_as<System::String^>(to\_string(n));

textBox7->Text = marshal\_as<System::String^>(to\_string(f));

textBox8->Text = marshal\_as<System::String^>(to\_string(e1));

textBox9->Text = marshal\_as<System::String^>(to\_string(d));

return System::Void();

}

* Функция вызываемая при нажатии на кнопку "Расшифровать" в форме.

//------------------------------------------------------------------

//Функция, вызываемая при нажатии на кнопку "Расшифровать" в форме.

//Устанавливает в текстовое поле 12 результат функции //расшифровывания.

System::Void LR5UI::RSA\_form::button2\_Click(System::Object^ sender, System::EventArgs^ e){

int d, n;

string text = "";

string out = "";

string d\_string = "";

string n\_string = "";

text = marshal\_as<string>(textBox2->Text);

d\_string = marshal\_as<string>(textBox11->Text);

n\_string = marshal\_as<string>(textBox10->Text);

d = stoll(d\_string);

n = stoll(n\_string);

decrypt(out, text, d, n);

textBox12->Text = marshal\_as<System::String^>(out);

return System::Void();

}

* Функция генерации простого числа

//--------------------------------------------------------------------

//функция генерации простого числа

int genPrime() {

int t = rand() % INT\_MAX/10;

do {

t++;

}

while (!prime(t));

return t;

}

* Функция проверки на простое число

//функция проверки на простое число

bool prime(int n) {

bool k = true;

if (n != 2) {

for (int i = 2; i < (int)sqrt(n) + 1; i++) {

//делим число на все числа до него

if (n % i == 0) {

// если остаток 0, число не простое

k = false;

break;

}

}

}

return k;

}

* Функция нахождения НОД

//--------------------------------------------------------------------

//функция нахождения НОД методом Евклида

int evklid(int num1, int num2) {

vector<int> r;

int i = 1;

r.push\_back(max(num1, num2));

r.push\_back(min(num1, num2));

while (r[i] != 0){

i++;

r.push\_back(r[i - 2] % r[i - 1]);

}

return r[i - 1];

}

* Функция генерации взаимного простого числа

//--------------------------------------------------------------------

//функция генерации взаимно простого числа

int genCoprime(int x){

int i = 1 + rand() % 99;

do{

i++;

}

while (evklid(x, i) != 1);

return i;

}

* Функция вычисления секретной экспоненты

//--------------------------------------------------------------------

//функция вычисления секретной экспоненты

int gend(int f, int e) {

for (int i = 1; i < 100; i++){

if (((f \* (i)+1) % e) == 0){

return (int)((f \* (i)+1) / e);

}

}

}

* Функция возведения в степень k по модулю n

//--------------------------------------------------------------------

//функция возведения в степень для больших чисел

long long int mul(int& x, int& k, int& n){

long long int res = 1;

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

res = res \* x % n;

}

return res;

}

* Функция зашифровки всего текста

//--------------------------------------------------------------------

//функция шифрования

void encrypt(string& out, const string& input, int& e, int& n){

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

for (int j = 0; j < sizeof(ALPHABET); j++){

if (ALPHABET[j] == input[i]){

//формула шифрования

int res = mul(j, e, n);

out += to\_string(res);

out += " ";

}

}

}

}

* Функция расшифровки текста

//--------------------------------------------------------------------

//функция расшифровки

void decrypt(string& out1, const string& input, int& d, int& n){

stringstream text(input);

string letter;

while (text >> letter){

int code = stoi(letter);

//фурмула расшифровки

int r = mul(code, d, n);

out1 += ALPHABET[r];

}

}