**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Факультет безопасности информационных технологий**

**Дисциплина:**

«Основы кибербезопасности»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

«**Асимметричные криптосистемы.**»

**Выполнил:**

студент группы K32211

Шкода Глеб Ярославович

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

**Проверил:**

Таранов Сергей Владимирович

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(отметка о выполнении)

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

Санкт-Петербург

2022 г.

**Цель:** изучить основные принципы работы асимметричных криптосистем на примере алгоритма RSA.

**Задачи практической работы**

1. Проанализировать эмуляцию алгоритма RSA и примитивных атак на шифр, используя Cryptool 2. Выделить основные необходимые настройки шифра и требуемые ограничения на параметры.
2. Программно реализовать и модифицировать любую асимметричную криптосистему. В случае отсутствия опыта программирования подойдет реализация алгоритма на псевдокоде или в виде блок схем, включающих основные этапы алгоритма с отображением формул и основных математических действий. Атаки и модификации, приведенные ниже, указаны для RSA. Если атака или модификация не применима для реализуемого алгоритма разрешается найти любую альтернативу (атаки, применимой к алгоритму; модификации для ускорения алгоритма и дополнительной защиты).
3. Для созданной реализации криптосистемы предлагается провести примитивный криптоанализ на устойчивость к следующим атакам, а также сделать минимальные модификации по оптимизации (ускорению процессов шифрования, дешифрования, процесса генерации ключей).

**Ход работы.**

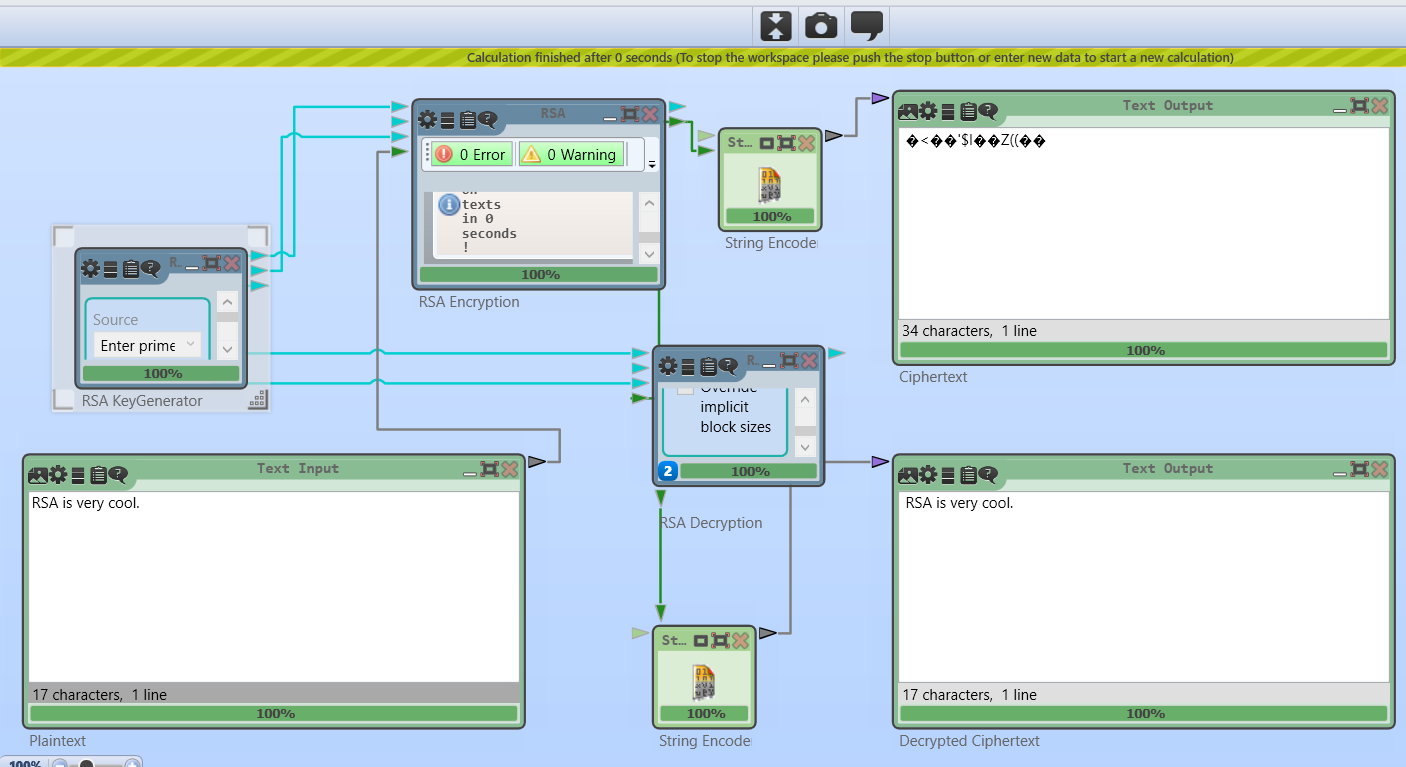
1. Для начала работы я скачал и установил Cryptool 2 на свой компьютер. Далее я создал workspace по шаблону RSA Cypher. Чтобы зашифровать небольшой текст я выбрал 2 простых числа p = 29 и q = 83, а также подобрал взаимно простое с ними число e = 23 (открытый ключ). На основе этих данных программа сама составила закрытый ключ d.



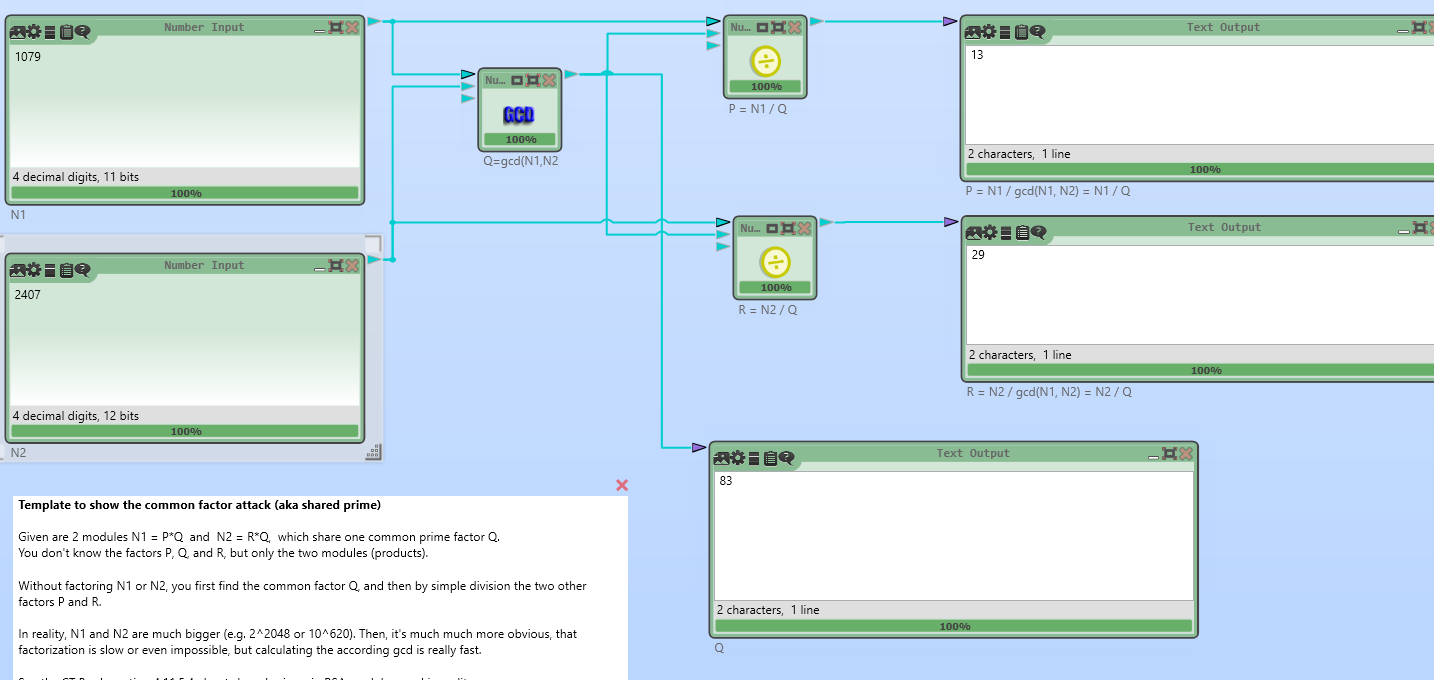
Закрытый ключ d можно вычислить как мультипликативное обратное к числу e по модулю (p – 1)\*(q – 1). Полученное значение закрытого ключа d = 599 (вычислить мультипликативное обратное можно, например, с помощью [онлайн калькулятора](https://planetcalc.ru/3311/)).

Итак, получив ключи, можно перейти к шифрованию текста. Как показано на скриншоте ниже, в результате работы Cryptool с помощью открытого ключа был получен зашифрованный текст, который затем был успешно расшифрован с помощью уже закрытого ключа.

Формулы шифрования и дешифрования:

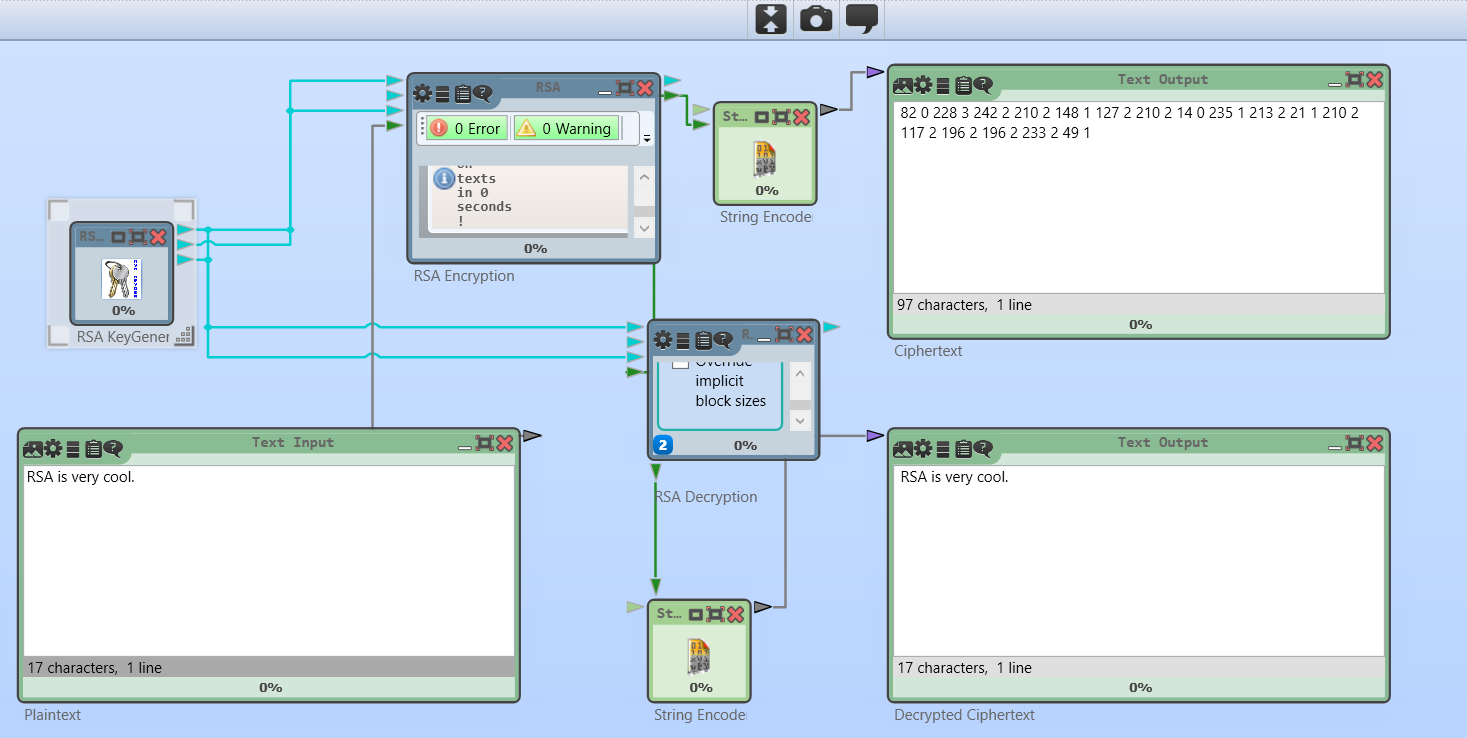


1. Пусть дано 2 модуля шифрования, каждый из 2 простых чисел p и q. Тогда, если p1=p2 или q1=q2, возможна атака на основе общего делителя. Пусть N1 = p1 \* q = 13 \* 83 = 1079 и N2 = p2 \* q = 29 \* 83 =2407. В этом случае, очевидно, что q может быть найден как НОД(N1, N2), а зная q простым делением можно получить p1 и p2.

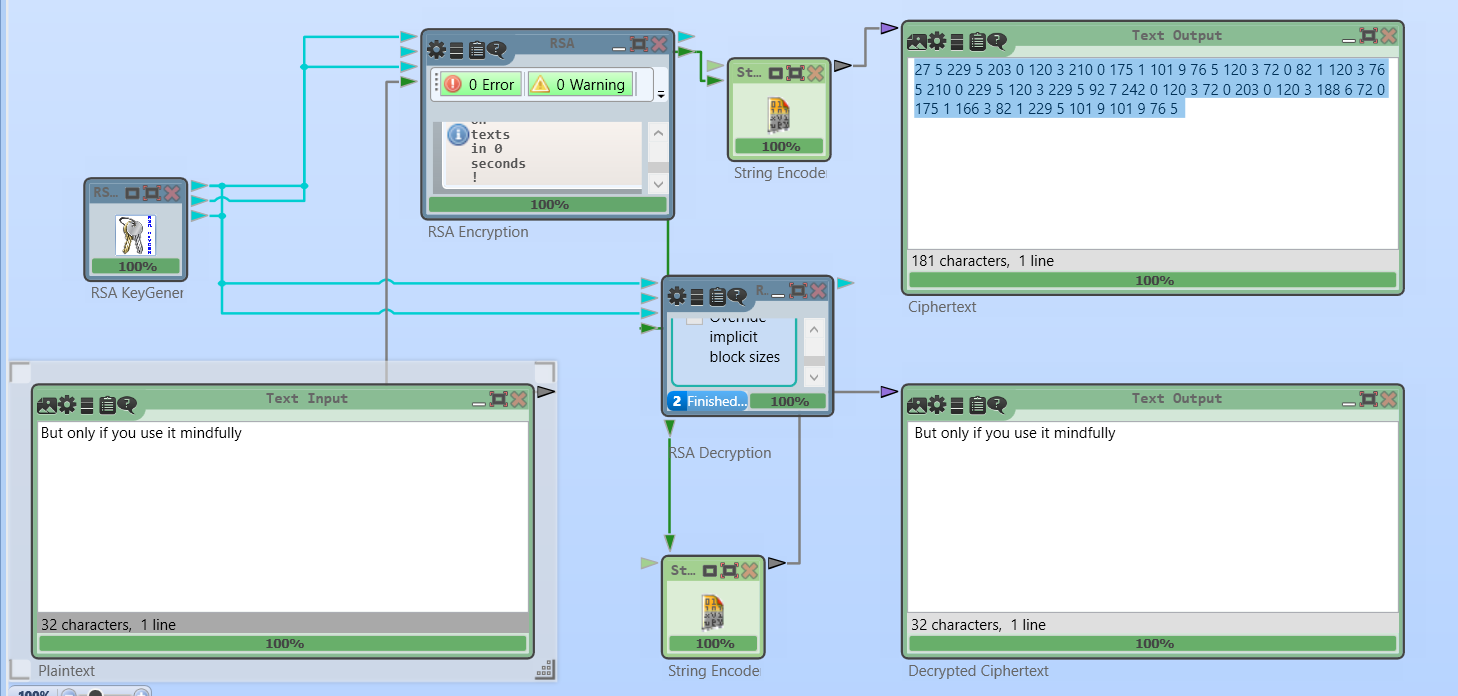


В прошлом пункте было показано, что, когда известны p, q и e (открытый ключ всегда известен, так как с помощью него проходит шифрование), несложно получить закрытый ключ d. Таким образом, если известно, что информация зашифрована с модулем шифрования N1 или N2, то можно найти закрытые ключи для этих модулей d1 и d2 и расшифровав информацию по модулю N1 и N2 останется только выбрать, тот результат, который выглядит подходяще.

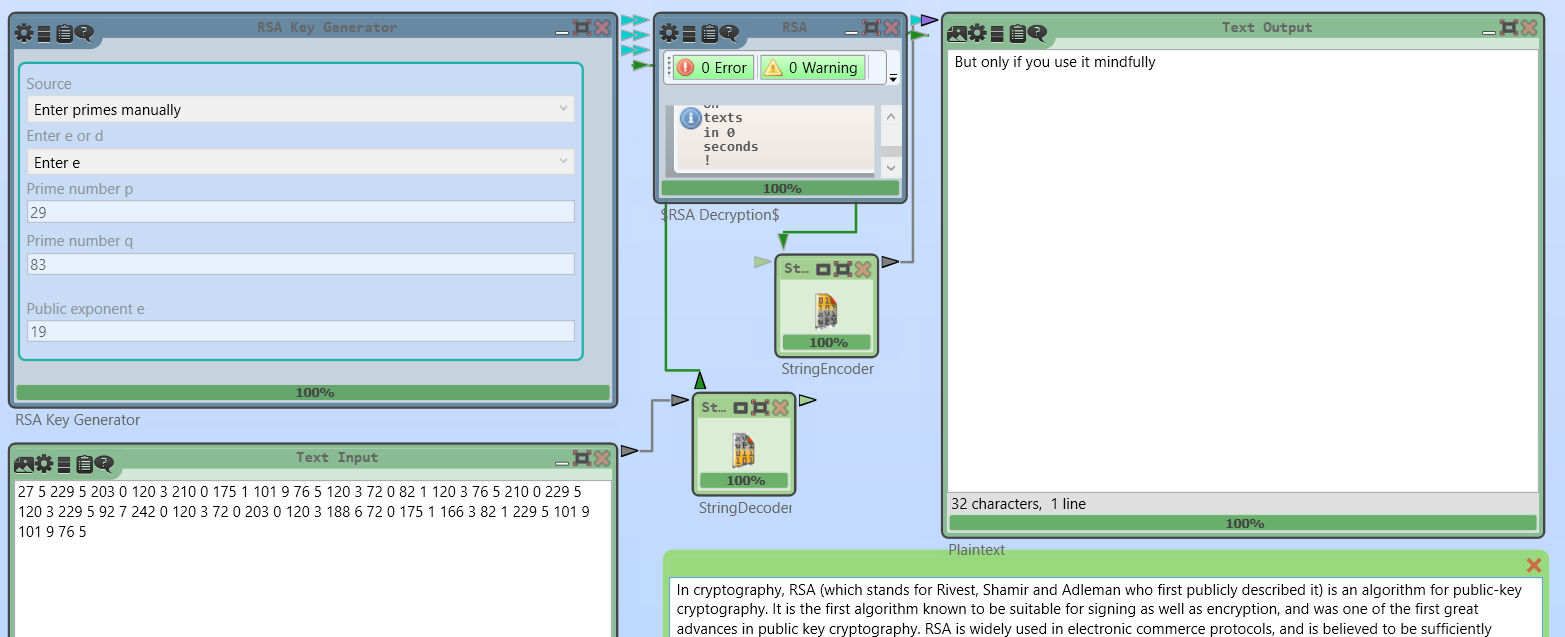
Пример. Зашифруем первый текст “RSA is very cool” (q = 83; p = 13; e = 19).



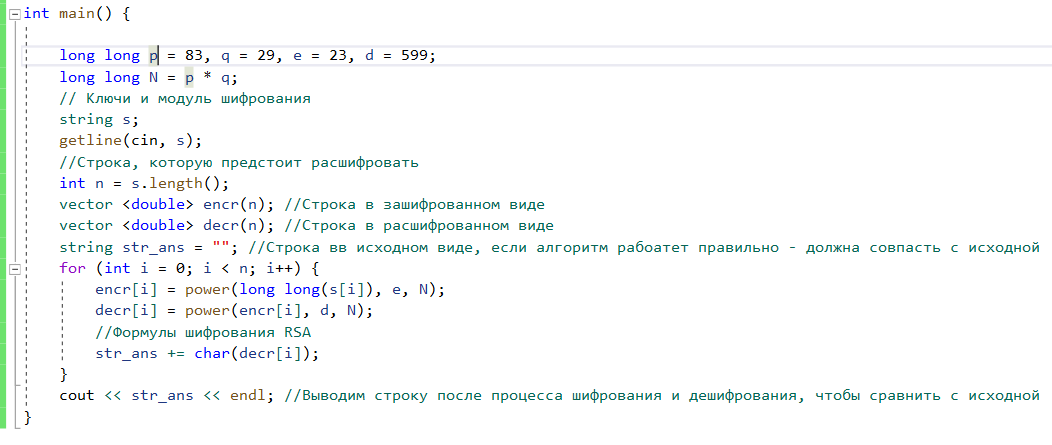
Зашифруем второй текст “But only if you use it mindfully” (q = 83; p = 29; e = 19).

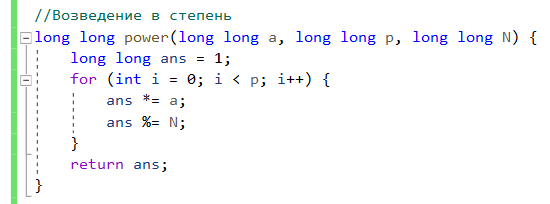


С помощью атаки на основе общего делителя, зная только N1 и N2 были восстановлены p и q для обоих модулей. Зная p и q, открытый ключ и саму зашифрованную информацию мне не составило труда её расшифровать с помощью RSA Decryption.

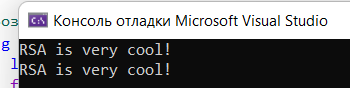


1. В этом пункте я реализовал алгоритм RSA на C++ с заранее известными p, q, e, d. К отчёту приложу файл с исходным кодом.

Также прикладываю код функции power().



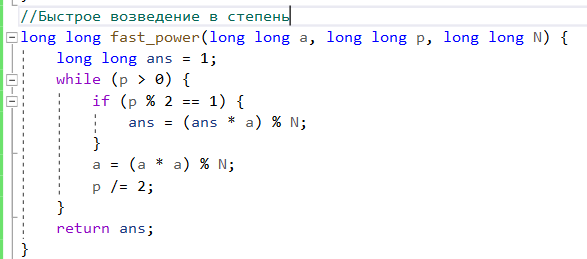
В данной программе на вход подаётся строка, которая сначала шифруется открытым ключом e, и сразу же дешифруется закрытым ключом d. Чтобы убедиться, что всё работает правильно, я сразу же восстанавливаю строку после применения алгоритма и вывожу её на экран.



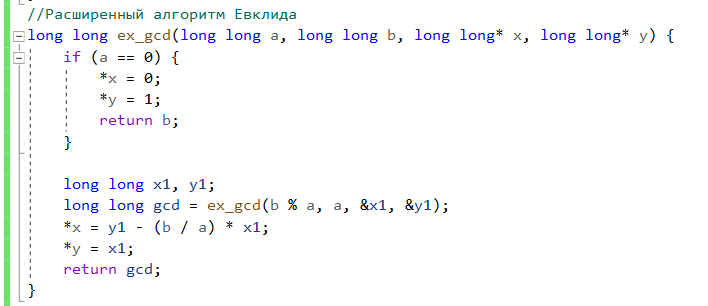
Как видно из скриншота, результат соответствует исходной строке.

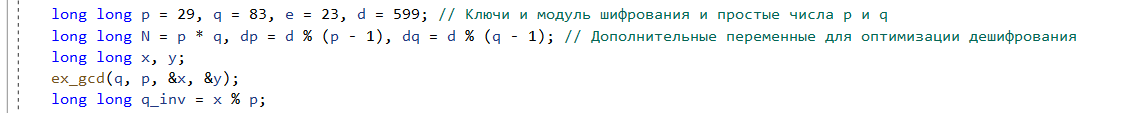
1. Увеличим эффективность написанной программой несколькими улучшениями

* Так как программа часто обращается к операции возведения в степень, следует обратить внимание на оптимизацию именно этой операции. Для этого я реализовал новую функцию возведения в степень.

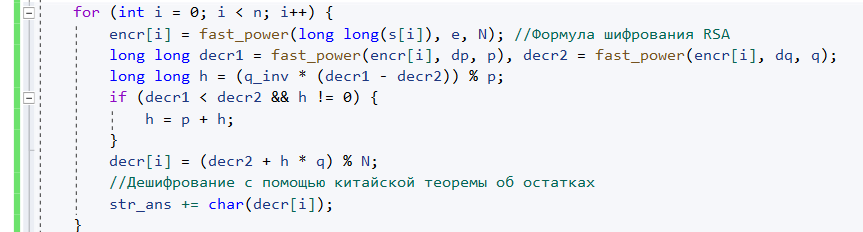


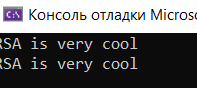
* Воспользуемся китайской теоремой об остатках, чтобы оптимизировать процесс дешифрования информации. Для этого сначала реализуем расширенный алгоритм Евклида, с помощью которого будем находить мультипликативное обратное.



Далее введём дополнительные переменные, которые помогут быстрее дешифровать сообщение. 

И, наконец, проведём само дешифрование новым способов

После всех манипуляций программа продолжает выдавать верный ответ.



**Вывод**

В результате выполнения данной лабораторной работы я изучил алгоритм шифрования RSA. С помощью программы CrypTool я смоделировал работу алгоритма на небольшом тесте и изучил его потенциальную уязвимость к атаке на основе общего делителя. Также я реализовал алгоритм RSA на языке C++ и внедрил в своё решения некоторые улучшения, которые сделали написанный код более эффективным.