МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Белорусский государственный технологический университет»

Кафедра информационных систем и технологий

**«OpenSSL (дополнение к лаб. раб. №8)»**

Студент:

Септилко Анастасия Антоновна

Преподаватель:

Блинова Евгения Александровна

Минск 2020

**ЗАДАНИЕ:**

**3**. Используя примерно одинаковый порядок ключевой информации, оценить производительность обоих алгоритмов и относительное изменение объемов криптотекстов (по отношению к объемам открытых текстов). Некоторые указания к инсталляции (при желании) библиотеки OpenSSL. OpenSSL – это система защиты и сертификации данных; SSL (Secure Socket Layer – система безопасных сокетов).

Внутри OpenSSL имеются отдельные компоненты (утилиты), отвечающие за то или иное действие. OpenSSL поддерживает, в том числе, много различных стандартов шифрования. К их числу относится RSA. Краткую начальную информация к процедуре инсталляции OpenSSL можно найти по адресу: https://www.xolphin.com/support/OpenSSL/OpenSSL\_-\_Installation\_under\_Windows Для загрузки OpenSSL можно также воспользоваться источником по адресу: https://sourceforge.net/projects/openssl/

Для начала необходимо сгенерировать приватный ключ. В Java класс PKCS8EncodedKeySpec ожидает закрытый ключ RSA с кодировкой PKCS8. (Java-код, н.д.). Сгенерируем с помощью OpenSSL. Команда приведена в Листинге 1.1.

Листинг 1.1

openssl genpkey -out private\_key.pem -algorithm RSA -pkeyopt rsa\_keygen\_bits:4096

После этого генерируем открытый ключ. Открытый ключ генерируется из закрытого ключа, поэтому сначала необходимо иметь закрытый ключ. Команда продемонстрирована в листинге 1.2.

Листинг 1.2

openssl rsa -pubout -outform pem -in private\_key.pem -out public\_key.pem

Можем убедиться, что ключи сгенерированы успешно отрыв файлы с названием «private\_key.pem» и «public\_key.pem». Содержимое этих файлов показаны на рисунке 1.1.

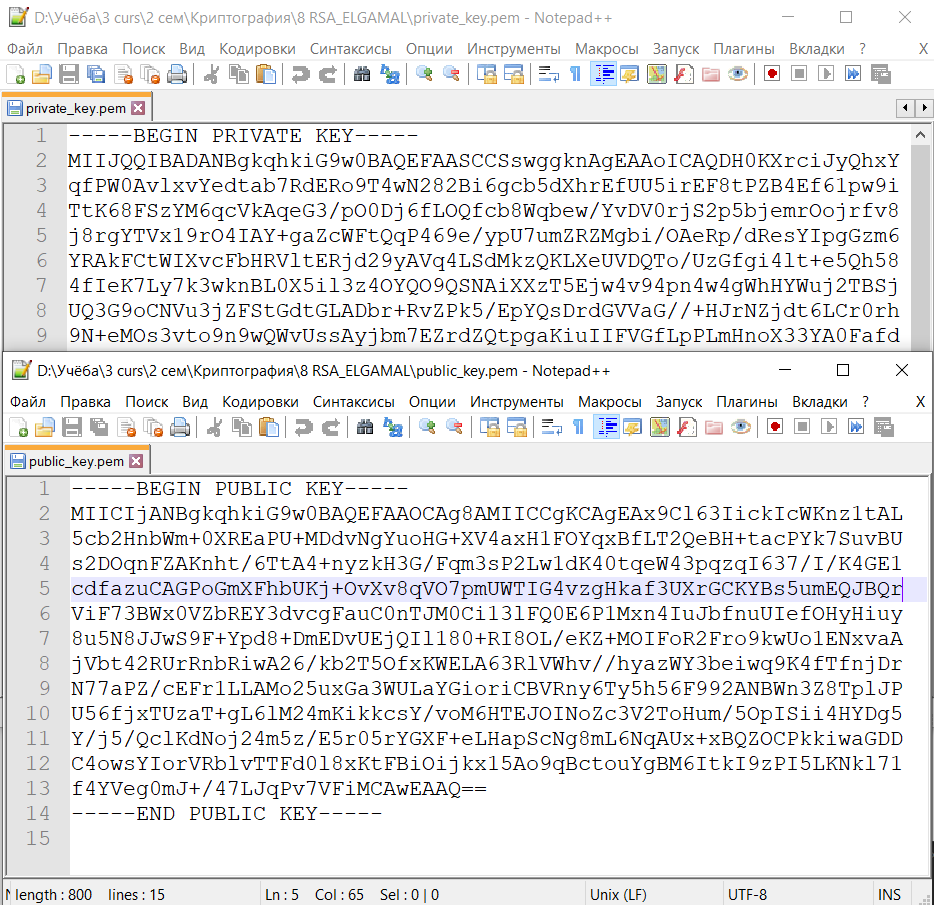


Рисунок 1.1 – Приватный и публичный ключ

Реализуем конструктор для алгоритма RSA (рисунок 1.2). Конструктор принимает 2 параметра privateKeyClassPathResource и publicKeyClassPathResource. Первый параметр – это полное имя пути к классу файла личного ключа. Второй параметр одинаков для файла открытого ключа.

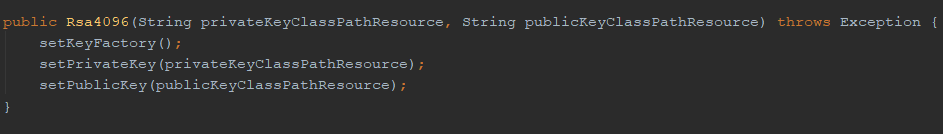


Рисунок 1.2 – Конструктор

Метод setKeyFactory() создает экземпляр класса KeyFactory для алгоритма RSA. Затем будет использоваться для генерации приватного и публичного ключей. Метод представлен на рисунке 1.3.

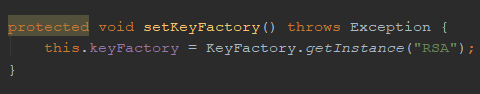


Рисунок 1.3 – Метод setKeyFactory()

Метод setPrivateKey() – установка приватного ключа. Код данного метода продемонстрирован на рисунке 1.4.

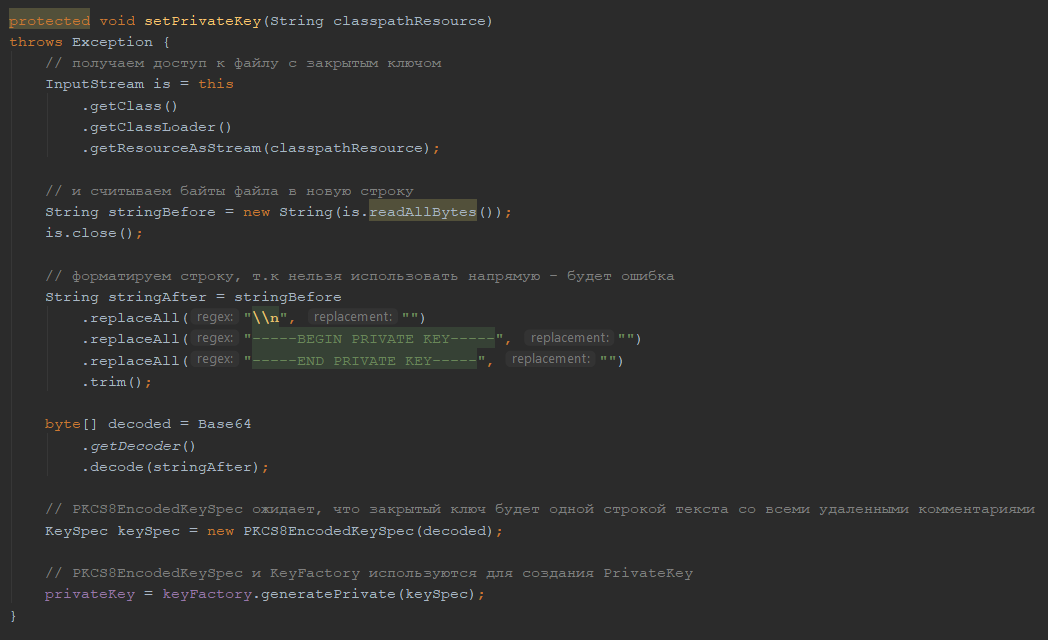


Рисунок 1.4 – Метод setPrivateKey()

Метод setPublicKey() – установка публичного ключа. Код данного метода продемонстрирован на рисунке 1.5.

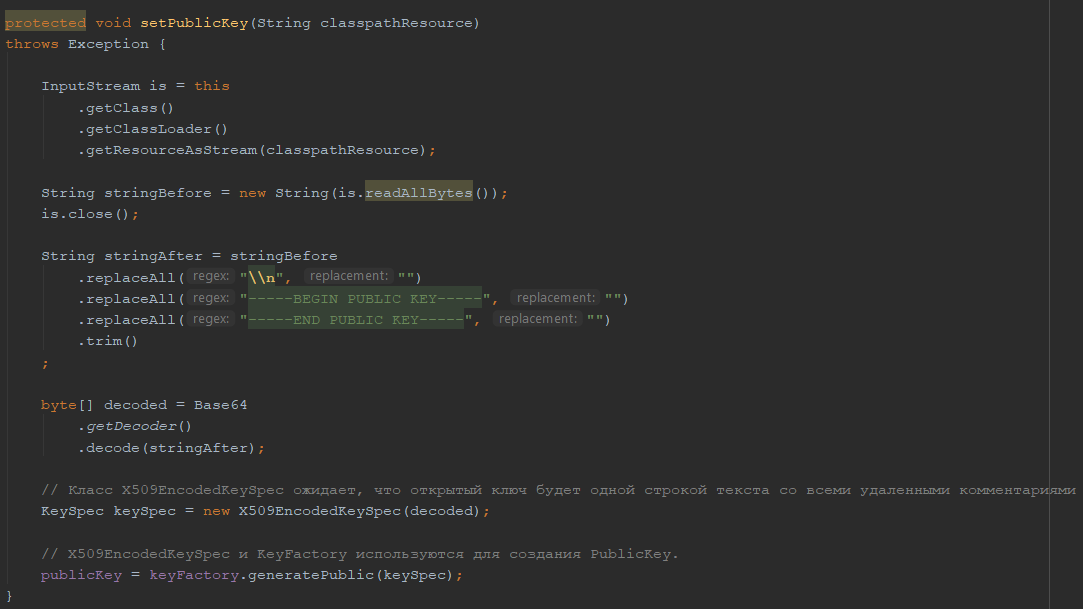


Рисунок 1.5 – Метод setPublicKey()

Метод шифрования представлен ниже на рисунке 1.6. Принимает в качестве параметра строку для шифрования. Возвращаемый тип говорит, что будет возвращена строка в кодировке Base64. Для зашифрования нужен только открытый ключ.

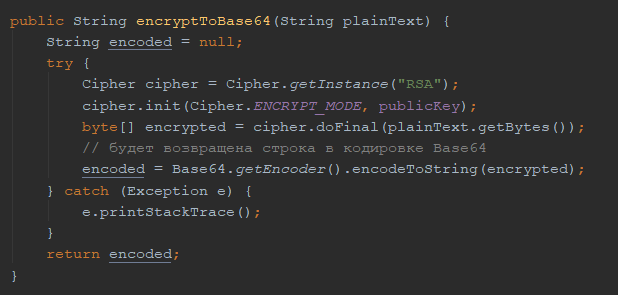


Рисунок 1.7 – Метод encryptToBase64()

Метод расшифрования показан на рисунке 1.8.

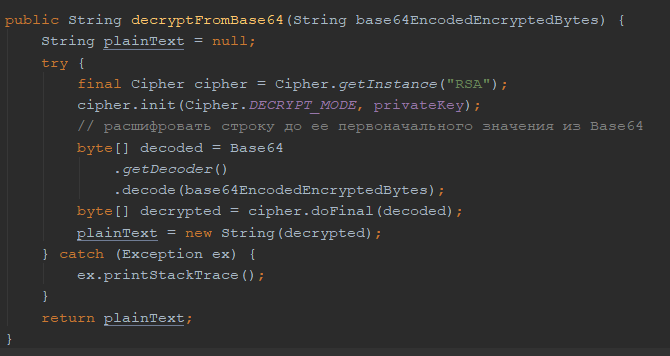


Рисунок 1.8 – Метод decryptFromBase64()

Строка для зашифрования и расшифрования содержит текст “Septilko Anastasiya”. Тест и результат представлен ниже на рисунке 1.9.



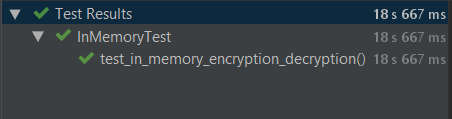


Рисунок 1.9 – Итог выполнения программы