## Вводная

В практической работе представлено Java-приложение для организации простейшего защищённого документооборота. В качестве задания требуется реализовать следующие функции:

* Шифрование и дешифрование сообщений;
* Шифрование и дешифрование документов (файлов);
* Создание цифровой подписи для документа.

В качестве алгоритмов шифрования предлагается использоваться комбинацию AES и RSA ключей. Подробнее о схеме шифрования описано в рекомендациях по выполнению.

## Задание 1

Реализовать функцию (де)шифрования сообщений. В интерфейсе SecurityService описаны функции:

String encryptMessage(String message, File publicKeyFile) throws IOException, GeneralSecurityException;

String decryptMessage(String message, File privateKeyFile) throws IOException, GeneralSecurityException, InvalidPassphraseException;

Реализация функции encryptMessage принимает на вход сообщение с интерфейса ввода и файл открытого ключа. Выходным значением будет строка, состоящая из зашифрованного сообщения дополнительно закодированного в Base64 для читаемости и удобства передачи.

Соответственно функция decryptMessage принимает закодированную Base64 строку и файл закрытого ключа.

## Задание 2

Реализовать функцию (де)шифрования документов (и в целом файлов).

В интерфейсе SecurityService описаны функции:

void encryptDocument(File document, File openKey) throws IOException, GeneralSecurityException;

void decryptDocument(File document, File secretKey) throws IOException, GeneralSecurityException, InvalidPassphraseException;

Реализация функции encryptDocument принимает на вход файл документа и файл открытого ключа. Функция не возвращает ничего, однако должна записывать на диск зашифрованный файл с тем же названием, что и у документа и добавить расширение “.sde”.

Реализация decryptDocument принимает файл с зашифрованным документом в форме “.sde” и файл закрытого ключа. Соответственно она записывает на диск файл документа в расшифрованном виде в изначальном формате файла.

## Задание 3

Реализовать функцию цифровой подписи документа закрытым ключом отправителя. А так же проверку подписи через открытый ключ.

В интерфейсе SecuritySerivce для этого описаны два метода:

void signDocument(File document, File privateKey) throws IOException, GeneralSecurityException, InvalidPassphraseException;

boolean verifyDocument(File document, File signFile, File publicKey) throws IOException, GeneralSecurityException;

Реализация метода signDocument принимает на вход файл документа и закрытый ключ. После чего происходит логика создания цифровой подписи и далее происходит запись на диске файла цифровой подписи с тем же названием документа, но в формате “.sig”.

Реализация функции verifyDocument принимает на вход файл документа, файл цифровой подписи и открытый ключ. В результате проверки возвращается true или false в зависимости от верификации документа.