## Алгоритмы асимметричного шифрования

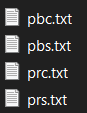
### Цель работы

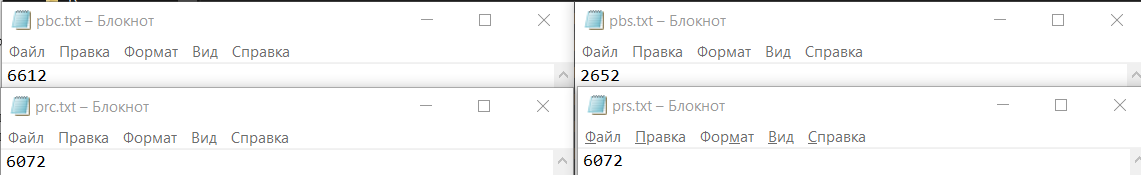
Познакомиться с принципами работы протоколов рукопожатия в современных компьютерных системах

### Дополнительные задания

1. Модифицируйте код клиента и сервера так, чтобы приватный и публичный ключ хранились в текстовых файлах на диске и, таким образом, переиспользовались между запусками.

**Каждый ключ хранится в своем файле:**





2. Проведите рефакторинг кода клиента и сервера так, чтобы все, относящееся к генерации ключей, установлению режима шифрования, шифрованию исходящих и дешифрованию входящих сообщений было отделено от основного алгоритма обмена сообщениями.

**Шифрование находится в файле shifr.py, генерация ключей в keys.py**

Изображение выглядит как текст

Автоматически созданное описание

3. Реализуйте на сервере проверку входящих сертификатов. На сервере должен храниться список разрешенных ключей. Когда клиент посылает на сервер свой публичный ключ, сервер ищет его среди разрешенных и, если такого не находит, разрывает соединение. Проверьте правильность работы не нескольких разных клиентах.

**Все сертифицированные ключи, в виде пар ключей-значения, где ключ – публичный ключ клиента, а значение – приватный ключ**

Изображение выглядит как текст

Автоматически созданное описание

4. Модифицируйте код клиента и сервера таким образом, чтобы установление режима шифрования происходило при подключении на один порт, а основное общение - на другом порту. Номер порта можно передавать как первое зашифрованное сообщение.

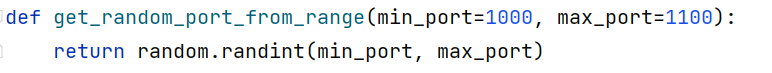
**После проверки ключа создается новое подключение на случайном порте с помощью нижеследующих функций.**

Изображение выглядит как текст

Автоматически созданное описание

5. Реализуйте пул портов.

**Порты могут быть взяты только из ограниченного диапазона [min\_port:max\_port]**



6. Модифицируйте код FTP-сервера таким образом, чтобы он поддерживал шифрование.

**Пример работы с фтп сервером:**

