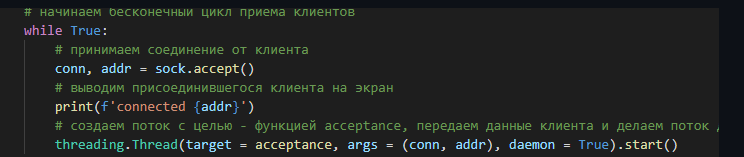
***Создание простого многопоточного сервера (доп. пп. 2-4)***

**Цель работы**

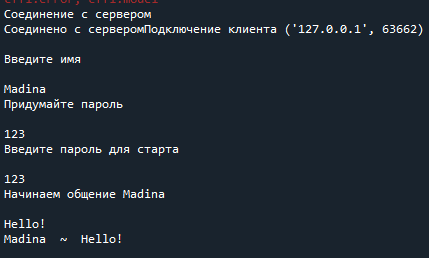
Познакомиться с приемами работы с многопоточностью на примере создания сокетного TCP-сервера, способного работать с несколькими клиентами одновременно.

**Дополнительные задания**

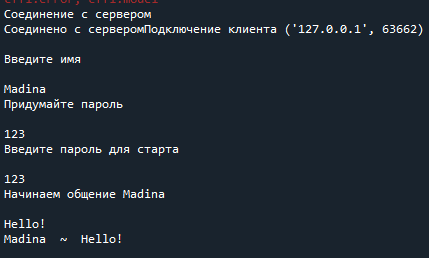
2. Модифицировать простой эхо-сервер таким образом, чтобы при подключении клиента создавался новый поток, в котором происходило взаимодействие с ним.

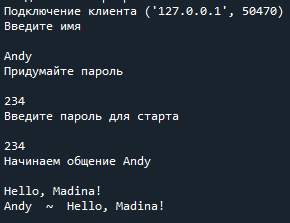




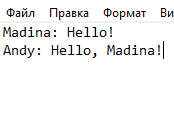


3. Реализовать простой чат сервер на базе сервера аутентификации. Сервер должен обеспечивать подключение многих пользователей одновременно, отслеживание имен пользователей, поддерживать историю сообщений и пересылку сообщений от каждого пользователя всем остальным.



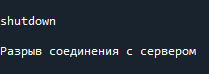


***История сообщений:***

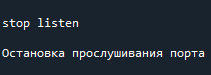


4. Реализовать сервер с управляющим потоком. При создании сервера прослушивание портов происходит в отдельном потоке, а главный поток программы в это время способен принимать команды от пользователя. Необходимо реализовать следующие команды:

I. Отключение сервера (завершение программы);



II. Пауза (остановка прослушивание порта);



У клиента ничего не происходит, он ожидает соединения, так как мы запретили подключения Если мы введем **start listen** то клиент сразу подключится

https://user-images.githubusercontent.com/58771506/135754425-f3507791-564c-4c72-a983-f78b5b821cd0.png

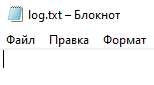
III. Показ логов;

Показ логов выключается и включается командами **stop log start log**



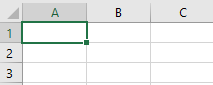
IV. Очистка логов;

Происходит путём ввода команды **«Очистка»**



V. Очистка файла идентификации.

Происходит путём ввода команды **«Удалить клиентов»**



**Вывод:** Таким образом, видим, что программа работает корректно и выполняет свои функции.