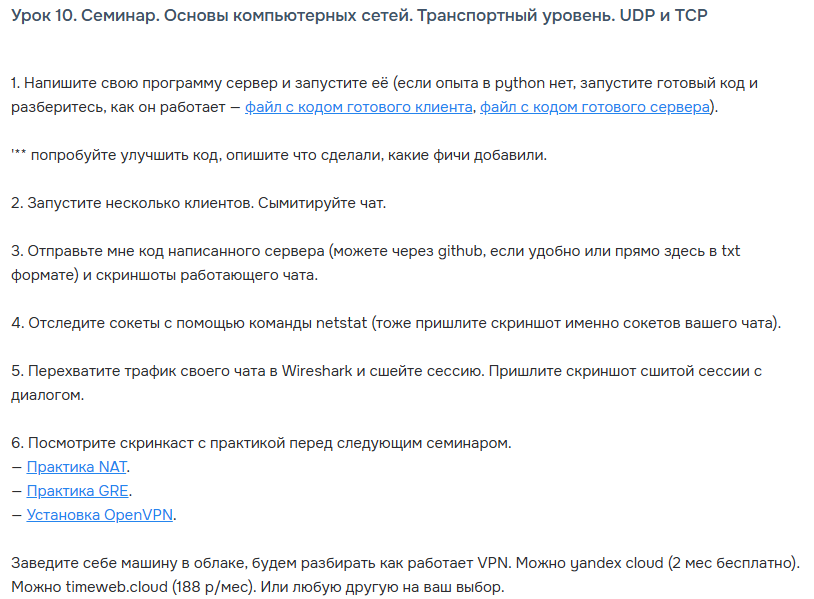
**Компьютерные сети. Обучение в записи**

**Урок 10. Семинар. Основы компьютерных сетей. Транспортный уровень. UDP и TCP**

Оглавление

[Домашнее задание 2](#_Toc195486803)

# Домашнее задание

****

**Ход выполнения домашнего задания:**

*Задание 1:*

1. *Напишите свою программу сервер и запустите её (если опыта в python нет, запустите готовый код и разберитесь, как он работает — файл с кодом готового клиента, файл с кодом готового сервера).*

*\*\* попробуйте улучшить код, опишите что сделали, какие фичи добавили*

* ***замена кодировки с ascii на utf-8;***
* ***добавление/корректировка отработки исключений и вывода сообщения;***
* ***добавление обязательного условия ввода никнейма пользователя;***
* ***добавление возможности выхода пользователя из чата***

*SERVER:*

import socket

import threading

# Connection Data

host = 'localhost'

port = 55555

# Starting Server

server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

server.bind((host, port))

server.listen()

# Lists For Clients and Their Nicknames

clients = []

nicknames = []

# Sending Messages To All Connected Clients

def broadcast(message):

for client in clients:

client.send(message)

# Handling Messages From Clients

def handle(client):

while True:

try:

# Broadcasting Messages

message = client.recv(1024)

broadcast(message)

except:

# Removing And Closing Clients

index = clients.index(client)

clients.remove(client)

client.close()

nickname = nicknames[index]

broadcast('{} left the chat!'.format(nickname).encode('utf-8'))

nicknames.remove(nickname)

break

# Receiving / Listening Function

def receive():

while True:

# Accept Connection

client, address = server.accept()

print("Connected with {}".format(str(address)))

# Request And Store Nickname

client.send('NICK'.encode('utf-8'))

nickname = client.recv(1024).decode('utf-8')

nicknames.append(nickname)

clients.append(client)

# Print And Broadcast Nickname

print("Nickname is {}".format(nickname))

broadcast("{} joined! ".format(nickname).encode('utf-8'))

client.send('Connected to server! To exit the chat, type "Esc"'.encode('utf-8'))

# Start Handling Thread For Client

thread = threading.Thread(target=handle, args=(client,))

thread.start()

print("Server if listening... Do not close this window until the end of the session.")

receive()

*CLIENT:*

import socket

import sys

import threading

# Choosing Nickname

nickname = input("Enter your nickname:\n")

# Обязательнный ввод никнейма

while nickname == '':

nickname = input("Enter your nickname:\n")

# Connecting To Server

client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Отработка возможных ошибок подключения

try:

client.connect(('localhost', 55555))

except ConnectionRefusedError:

print(

"ConnectionRefusedError: [WinError 10061] Подключение не установлено, т.к. конечный компьютер отверг запрос на подключение")

client.close()

sys.exit()

except Exception as e:

print(f"Ошибка подключения: {e}")

client.close()

sys.exit()

# Listening to Server and Sending Nickname

def receive():

while True:

try:

# Receive Message From Server

# If 'NICK' Send Nickname

message = client.recv(1024).decode('utf-8')

if message == 'NICK':

client.send(nickname.encode('utf-8'))

else:

print(message)

except Exception as e:

# Close Connection When Error

print(f"{e}")

client.close()

break

def write():

while True:

message = input('')

if message.lower() == "esc":

client.close()

sys.exit()

else:

out\_message = '{}: {}'.format(nickname, message)

client.send(out\_message.encode('utf-8'))

# Starting Threads For Listening And Writing

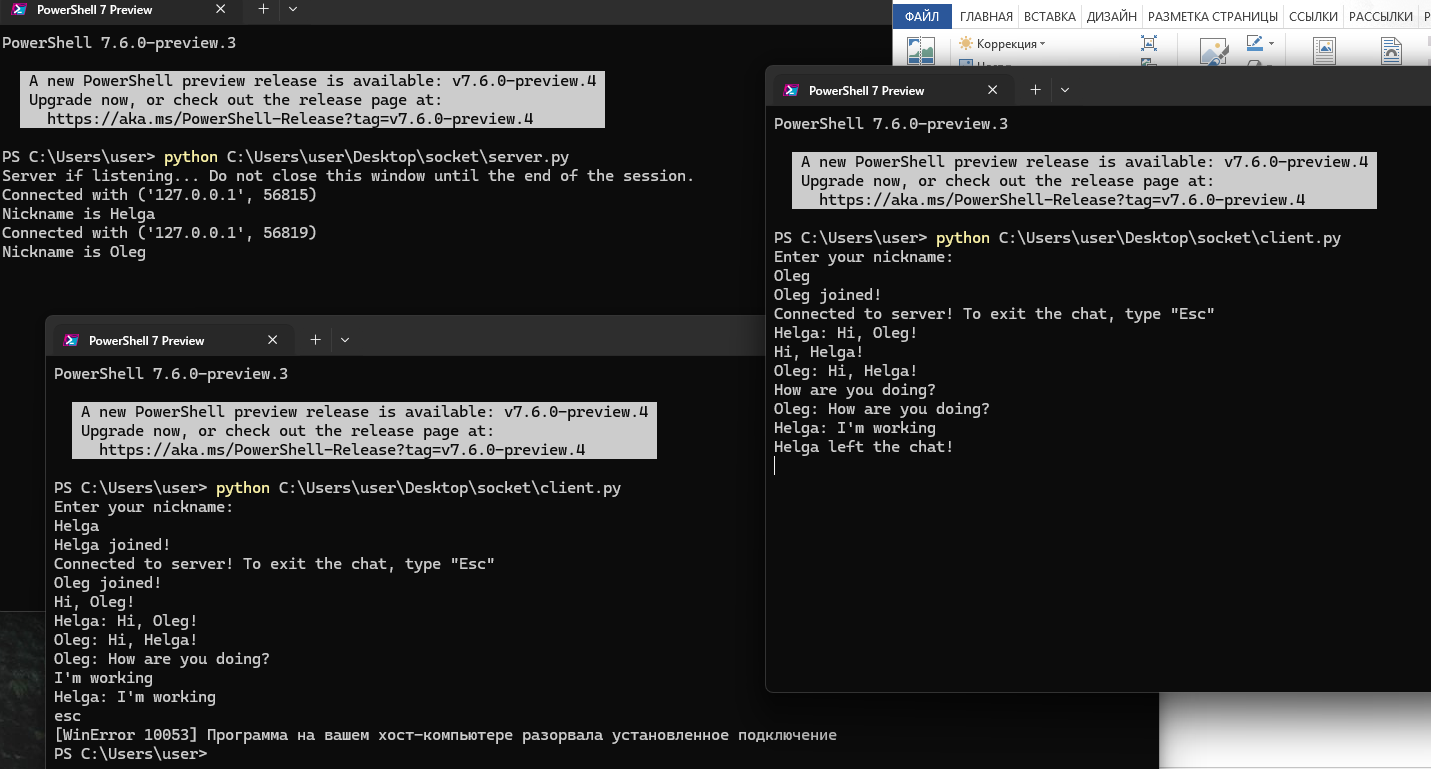
receive\_thread = threading.Thread(target=receive)

receive\_thread.start()

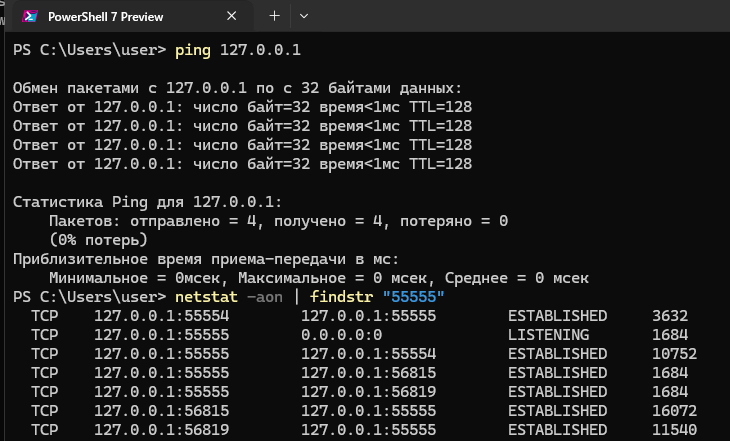
write\_thread = threading.Thread(target=write)

write\_thread.start()

*Задание 2: Запустите несколько клиентов. Сымитируйте чат.*



*Задание 4: Отследите сокеты с помощью команды netstat (тоже пришлите скриншот именно сокетов вашего чата).*



Ссылка на репозиторий:

<https://github.com/olgashenkel/GeekBrains-specialization-ELECTIVES/tree/main/08.%20Computer%20networks>