Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

Высшего образования

Московский авиационный институт

(Национальный исследовательский университет)

Институт №3

«Системы управления, информатика и электроэнергетика»

Кафедра 304

«Вычислительные машины, системы и сети»

**Отчет по лабораторной работе №10**

по учебной дисциплине «Web-технологии»

на тему «Чат на сокетах»

25 вариант

Исполнители:

обучающийся группы М3О-109Б-23

Черных А. П.

Руководители работы:

Титов Ю. П.

Москва 2024

Лабораторная работа №10. Чат на сокетах

1.Разработать программу сервер, которая хранит список всех подключении клиентов

(например, в динамическом массиве). Хранится сокет подключения, полученный в

результате функции accept. Для каждого клиента сервер запрашивает кодовое имя

(nickname). При получении сообщения от клиента (с соответствующим nickname)

происходит передача этого сообщения всем остальным клиентам через сокетное

соединение. Сервер сохраняет сообщение пользователя в файл. При подключении

клиента передаются (от сервера к клиенту) последние 10 сообщений из данного

файла.

2.Разработать программу клиент, которая работает в диалоге с пользователем и

позволяет ему вводить данные. При подключении к серверу (ip адрес и порт

сервера вводится пользователем) сервер запрашивает nickname, а пользователь

вводит его в программе-клиенте. Программа клиент позволяет вводить сообщения

пользователем от имени nickname и выводит сообщения от сервера на экран.

Последние 5 сообщений сохраняются в текстовом файле (эмулирую кэширование).

При запуске программы выводится содержимое данного файла.

3.Развернуть данное приложение в сети с одним сервером и 3-мя запущенными

программами-клиентами. В отчете привести файлы клиента и сервера, а также

скриншот работы программы.

1.многопоточный client на python:

**import** socket

**import** threading

curCache = []

**def** editStr(str):

new = str.split('```')

**return** new

**def** fiveElem(elem):

length = len(curCache)

**if** length>=5:

**for** i **in** range(length-4):

curCache.pop(-5)

curCache.append(elem)

**def** cacheStr():

str = ''

**for** i **in** curCache:

str+=(i)

**return** str

**def** handle\_client(client\_socket, lock):

**while** True:

**try**:

data = client\_socket.recv(1024)

**for** i **in** editStr(data.decode()):

**print**(i)

fiveElem(i)

**with** lock:

**with** open("client00.txt", "w", encoding="utf-8") **as** file:

file.write(cacheStr())

**if** **not** data:

**break**

**except** Exception **as** e:

**print**(f"error: {e}")

**break**

client\_socket.close()

**def** main():

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

**try**:

lock = threading.Lock()

client\_socket.connect(("127.0.0.1", 8080))

thread = threading.Thread(target=handle\_client, args=(client\_socket,lock))

thread.start()

**except** Exception **as** e:

**print**(f"error: {e}")

client\_socket.close()

**while** True:

message = input()

fiveElem(f"you: {message}\n")

**with** lock:

**with** open("client00.txt", "w", encoding="utf-8") **as** file:

file.write(cacheStr())

client\_socket.sendall(message.encode())

**if** \_\_name\_\_ == "\_\_main\_\_":

main()

2.многопроцессорный сервер на python:

**import** socket

**import** time

**import** multiprocessing

**def** search\_nick(addr, client\_base):

**for** i **in** client\_base:

**if** str(i[1])==str(addr):

**return** i[2]

**def** send(addr, mes, client\_base):

**for** i **in** client\_base:

**if** str(i[1])!=str(addr):

sen = str(search\_nick(addr,client\_base),': ', mes)

i[0].sendall(sen.encode())

**def** edit\_client(sock, addr, client\_base,lock):

sock.send('введите никнейм'.encode())

nickname = sock.recv(1024)

ip\_bind\_nick = [sock, addr, nickname.decode()]

client\_base.append(ip\_bind\_nick)

**print**(nickname.decode(), 'зашел в чат')

**with** lock:

**with** open("textF.txt") **as** file:

lines = file.readlines()

lines\_new=[]

**for** line **in** lines:

lines\_new.append(line[1:])

flag = False

**if** len(lines\_new)<=10:

**for** i **in** lines\_new:

sock.send(i.encode())

flag = True

**else**:

**for** i **in** lines\_new[-10:]:

sock.send((i+'"символ разделитель"').encode())

flag = True

**if** flag:

sock.sendall('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\nhistory of chat ↑↑↑\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n'.encode())

**while** True:

**try**:

data = sock.recv(1024)

**with** lock:

**with** open("textF.txt", "a") **as** file:

file.write(f"{nickname}: {data.decode()}"+'\n')

mes = data.decode()

**print**(search\_nick(addr,client\_base),': <',mes,'>')

**for** i **in** client\_base:

**if** str(addr)!=str(i[1]):

sen = str(search\_nick(addr,client\_base))+': '+str(mes)

i[0].sendall(('\n'+sen+'\n').encode())

**except** Exception:

sock.close()

**return**

**print**("Client disconnected")

sock.close()

**def** main():

client\_base = multiprocessing.Manager().list()

lock = multiprocessing.Lock()

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

server\_ip = '127.0.0.1'

server\_port = 8080

server\_socket.bind((server\_ip, server\_port))

server\_socket.listen(5)

**print**("Сервер запущен. Ожидание соединений...")

**while** True:

client\_socket, addr = server\_socket.accept()

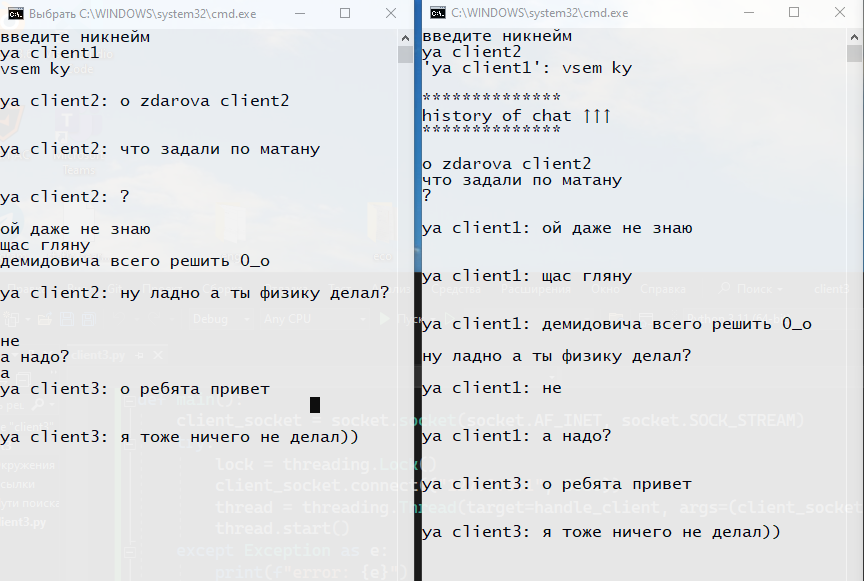
process = multiprocessing.Process(target=edit\_client, args=(client\_socket, addr, client\_base,lock))

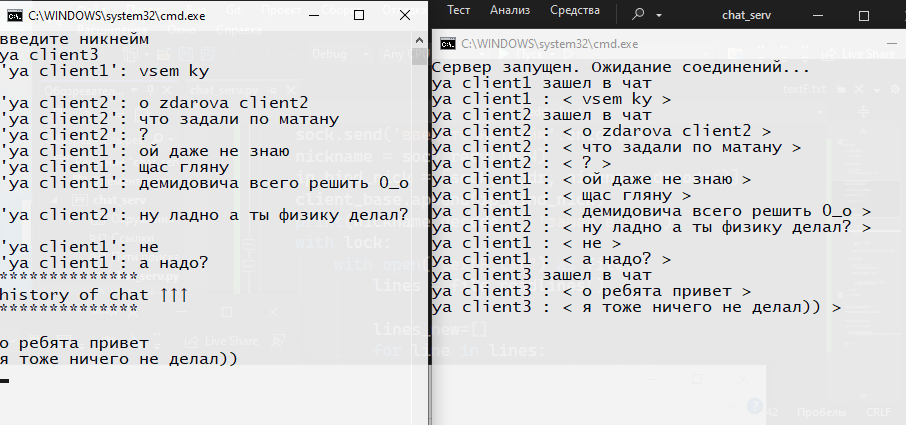
process.start()

**if** \_\_name\_\_ == '\_\_main\_\_':

main()

3.запустим сервер и 3 клиента, протестируем чат:





Рассмотрим кэш:

