Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №1 «Работа с сокетами»

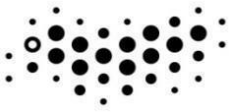
по дисциплине «Web-программирование»

Автор: Лакиза Александр Николаевич

Факультет: ИКТ

Группа: K33421

Преподаватель: Говроров Антоно Игоревич



Санкт-Петербург 2021

Цель работы: овладеть практическими навыками и умениями реализации web-серверов и использовании сокетов.

Ход работы:

Задание 1: Реализовать клиентскую и серверную часть приложения. Клиент отсылает серверу сообщение «Hello, server». Сообщение должно отразиться на стороне сервера. Сервер в ответ отсылает клиенту сообщение «Hello, client». Сообщение должно отобразиться у клиента

server.py client.py

```python
import socket

conn = socket.socket(socket.AF_INET,
                     socket.SOCK_STREAM,
                     proto=0)
conn.bind(('127.0.0.1', 53210))
conn.listen()
client_sock, client_addr = conn.accept()

data = client_sock.recv(1024)
decoded_data = data.decode("utf-8")

if decoded_data:
    print(f"Recieved from client:\n{decoded_data}")
    client_sock.sendall(b'Hello, client')
conn.close()
```

server.py client.py

```python
import socket

conn = socket.socket(socket.AF_INET,
                     socket.SOCK_STREAM,
                     proto=0)
conn.connect(('127.0.0.1', 53210))

conn.sendall(b'Hello, server')

data = conn.recv(1024)
decoded_data = data.decode("utf-8")

if decoded_data:
    print(f"Received from server:\n{decoded_data}")
conn.close()
```

```
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelopment
works/Lakiza_Alexander/laboratory_work_1/task1$ python3 server.py
Recieved from client:
Hello, server
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelopment
works/Lakiza_Alexander/laboratory_work_1/task1$
```

```
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelo
works/Lakiza_Alexander/laboratory_work_1/task1$ python3 client.py
Received from server:
Hello, client
```

Задание 2: Реализовать клиентскую и серверную часть приложения. Клиент запрашивает у сервера выполнение математической операции, параметры, которые вводятся с клавиатуры. Сервер обрабатывает полученные данные и возвращает результат клиенту b) решение квадратного уравнения

server.py client.py

```python
import socket

def solve_quad_eq(a, b, c):
    a, b, c = float(a), float(b), float(c)
    d = (b ** 2) - (4 * a * c)
    if d < 0:
        return 'No solutions'
    elif d == 0:
        x = (-b / (2 * a))
        return f"x = {x}"
    else:
        x0 = ((-b + (d ** (0 / 5))) / (2 * a))
        x1 = ((-b - (d ** (0 / 5))) / (2 * a))
        return f"x0 = {x0}, x1 = {x1}"


conn = socket.socket(socket.AF_INET,
                     socket.SOCK_STREAM,
                     proto=0)
conn.bind(('127.0.0.1', 53210))
conn.listen()
client_sock, client_addr = conn.accept()

data = client_sock.recv(1024)
decoded_data = data.decode("utf-8")

if decoded_data:
    coeffs = decoded_data.split()
    coeffs = [float(i) for i in coeffs]
    result = solve_quad_eq(coeffs[0], coeffs[1], coeffs[2])
    client_sock.sendall(bytes(result, "utf-8"))
conn.close()
```

server.py client.py

```python
import socket

conn = socket.socket(socket.AF_INET,
                     socket.SOCK_STREAM,
                     proto=0)
conn.connect(('127.0.0.1', 53210))

coeffs = input("Enter a, b and c of your quadratic equation separated by space\n")
conn.sendall(bytes(coeffs, "utf-8"))

data = conn.recv(1024)
decoded_data = data.decode("utf-8")

if decoded_data:
    coeffs = coeffs.split()
    print(
        f"Roots of your quadratic equation (({coeffs[0]})*x^2 + ({coeffs[1]})*x + "
        f"({coeffs[2]}) = 0) are:\n{decoded_data}")
conn.close()
```

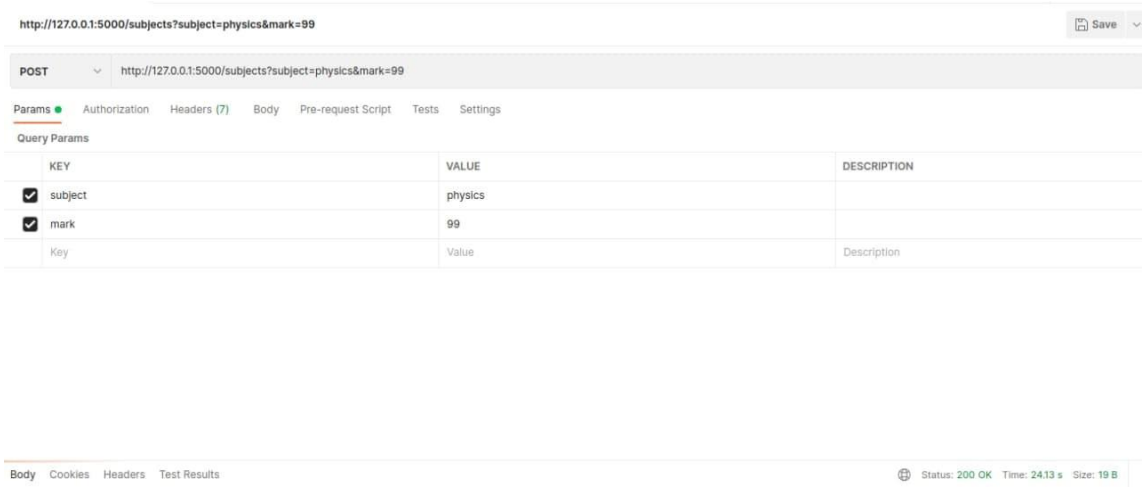
```
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelop
works/Lakiza_Alexander/laboratory_work_1/task2$ python3 server.py
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelop
works/Lakiza_Alexander/laboratory_work_1/task2$ python3 server.py
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelop
works/Lakiza_Alexander/laboratory_work_1/task2$
```

```
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelop
works/Lakiza_Alexander/laboratory_work_1/task2$ python3 client.py
Enter a, b and c of your quadratic equation separated by space
1 -12 36
Roots of your quadratic equation ((1)*x^2 + (-12)*x + (36) = 0) are:
x = 6.0
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelop
works/Lakiza_Alexander/laboratory_work_1/task2$ python3 client.py
Enter a, b and c of your quadratic equation separated by space
1 7 -18
Roots of your quadratic equation ((1)*x^2 + (7)*x + (-18) = 0) are:
x0 = -3.0, x1 = -4.0
alexanderlakiza@alela-KPL-W0X:~/pyfiles/itmo-web/ITMO_ICT_WebDevelop
works/Lakiza_Alexander/laboratory_work_1/task2$
```

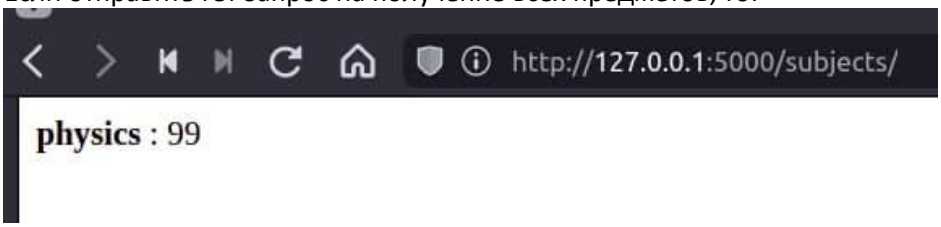
Задание 3: Задание: сделать сервер, который может:

- Принять и записать информацию о дисциплине и оценке по дисциплине.

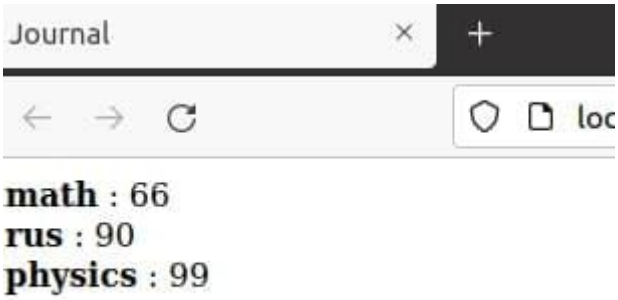
- Отдать информацию обо всех оценах по дсициплине в виде html-страницы



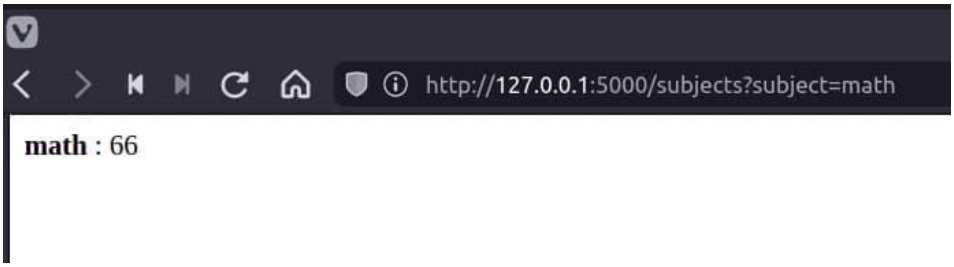
Если отправить гет запрос на получение всех предметов, то:



Добавим еще предметов



Если отправить гет запрос на получение оценки лишь по одному предмету, то:



Код task3/server.py

```
import socket
import re



class MyHTTPServer:
    def __init__(self, host, port):
        self.host = host
        self.port = port
        self.journal = {}
```

```
def serve_forever(self):
    # Starting a server on a socket, handling connections
    conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    conn.bind((self.host, self.port))
    conn.listen(10)
    while True:
        clientsocket, address = conn.accept()
        self.serve_client(clientsocket)

def serve_client(self, clientsocket):
    data = clientsocket.recv(16384)
    data = data.decode('utf-8')
    url, method, headers = self.parse_request(data)
    resp = self.handle_request(url, method)
    if resp:
        self.send_response(clientsocket, resp)

def parse_request(self, data):
    data = data.replace('\r', '')
    lines = data.split('\n')
    method, url, protocol = lines[0].split()
    i = lines.index('')
    headers = lines[1:i]
    print(url)
    return url, method, headers

def handle_request(self, url, method):
    if url == '/subjects/':
        resp = "HTTP/1.1 200 OK\n\n"
        with open('index.html', 'w') as f:
```

```
                text = '<!DOCTYPE html><html
lang="en"><head><html><head><title>Journal</title></head></html><body>
'

                for i in self.journal:

                    text += f"<b>{i}</b> :
{self.journal[i]}</br></body></html>"

                f.write(text)

            resp += text

            return resp

        if method == 'GET':

            resp = "HTTP/1.1 200 OK\n\n"

            subject = re.findall('\?(.*)$', url)[0]

            subject = subject.split('=')[1]

            with open('index.html', 'w') as f:

                text = '<!DOCTYPE html><html
lang="en"><head><html><head><title>Journal</title></head></html><body>
'

                if subject in self.journal:

                    text += f"<b>{subject}</b> :
{self.journal[subject]}</br></body></html>"

                else:

                    text += f"There is no <b>{subject}</b> in the
journal</body></html>"

                f.write(text)

            resp += text

            return resp

        else:

            resp = "HTTP/1.1 200 OK\n\n"

            info = re.findall('\?(.*)$', url)[0]

            info = info.split('&')

            subject = info[0].split('=')[1]

            grade = info[1].split('=')[1]

            self.journal[subject] = grade
```

```
            with open('index.html', 'w') as f:
                text = '<!DOCTYPE html><html
lang="en"><head><html><head><title>Journal</title></head></html><body>
'
                for i in self.journal:
                    print(self.journal)
                    text += f"<b>{i}</b> :
{self.journal[i]}</br></body></html>"
                f.write(text)
            resp += f'<!DOCTYPE html><html
lang="en"><head><html><head><title>Journal</title>' \
                    f'</head><body><b>{subject}</b> successfully
added</body></html>'
            return resp


    def send_response(self, clientsocket, resp):
        clientsocket.send(resp.encode('utf-8'))



if __name__ == '__main__':
    host = '127.0.0.1'
    port = 5000
    serv = MyHTTPServer(host, port)
    try:
        serv.serve_forever()
    except KeyboardInterrupt:
        pass
```

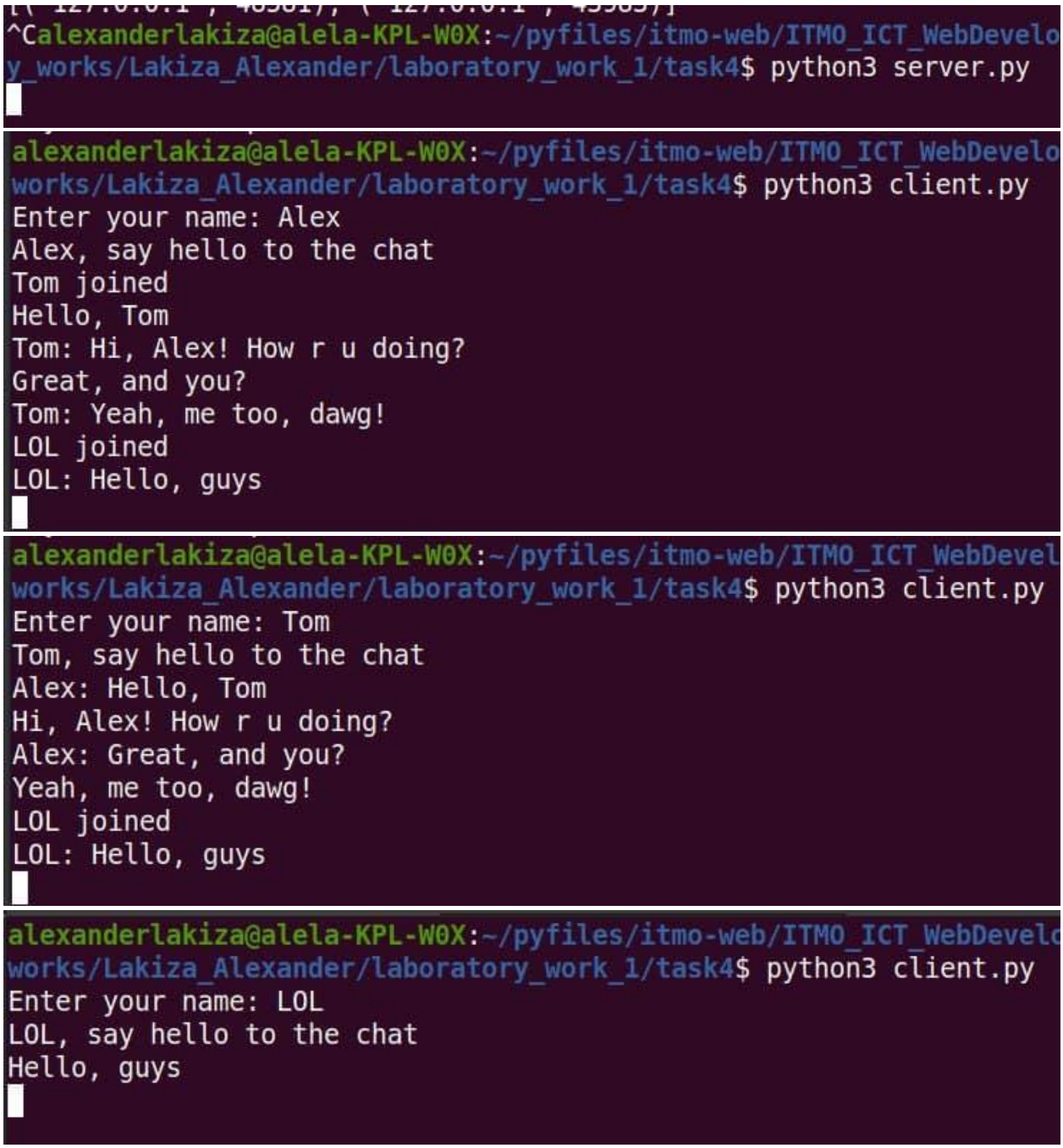
Задание 4: Реализовать двухпользовательский или многопользовательский чат. Реализация многопользовательского часа позволяет получить максимальное количество баллов.

server.py × client.py ×

```python
import socket

conn = socket.socket(socket.AF_INET,
                     socket.SOCK_DGRAM)
conn.bind(("127.0.0.1", 53210))


def run():
    clients = []
    while True:
        try:
            data, client_address = conn.recvfrom(2048)
            if client_address not in clients:
                clients.append(client_address)
            for client in clients:
                if client != client_address:
                    conn.sendto(data, client)
        except KeyboardInterrupt:
            conn.close()
            break


if __name__ == "__main__":
    run()

```

server.py × client.py ×

```python
import socket
import threading

address = "127.0.0.1", 53210
conn = socket.socket(socket.AF_INET,
                     socket.SOCK_DGRAM)
conn.connect(('127.0.0.1', 53210))


def messages():
    while True:
        data = conn.recv(2048)
        print(data.decode("utf-8"))


def chat():
    name = input("Enter your name: ")
    print(f'{name}, say hello to the chat')
    conn.sendall(bytes(f"{name} joined", "utf-8"))
    while True:
        conn.sendto(bytes(f"{name}: {input()}", "utf-8"), address)


if __name__ == "__main__":
    thread1, thread2 = threading.Thread(target=messages), threading.Thread(target=chat)
    thread1.start(), thread2.start()

```



Вывод: овладел практическими навыками и умениями реализации web-серверов и использования сокетов на Python