Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ №1

по теме: сокеты
по дисциплине: Web-программирование

Специальность:
09.03.03 Мобильные и сетевые технологии

Проверил:
Говоров А.И. _____
Дата: «08» октября 2021г.
Оценка _____

Выполнил:
студент группы К33401
Фоменко Иван

Санкт-Петербург 2021 г.

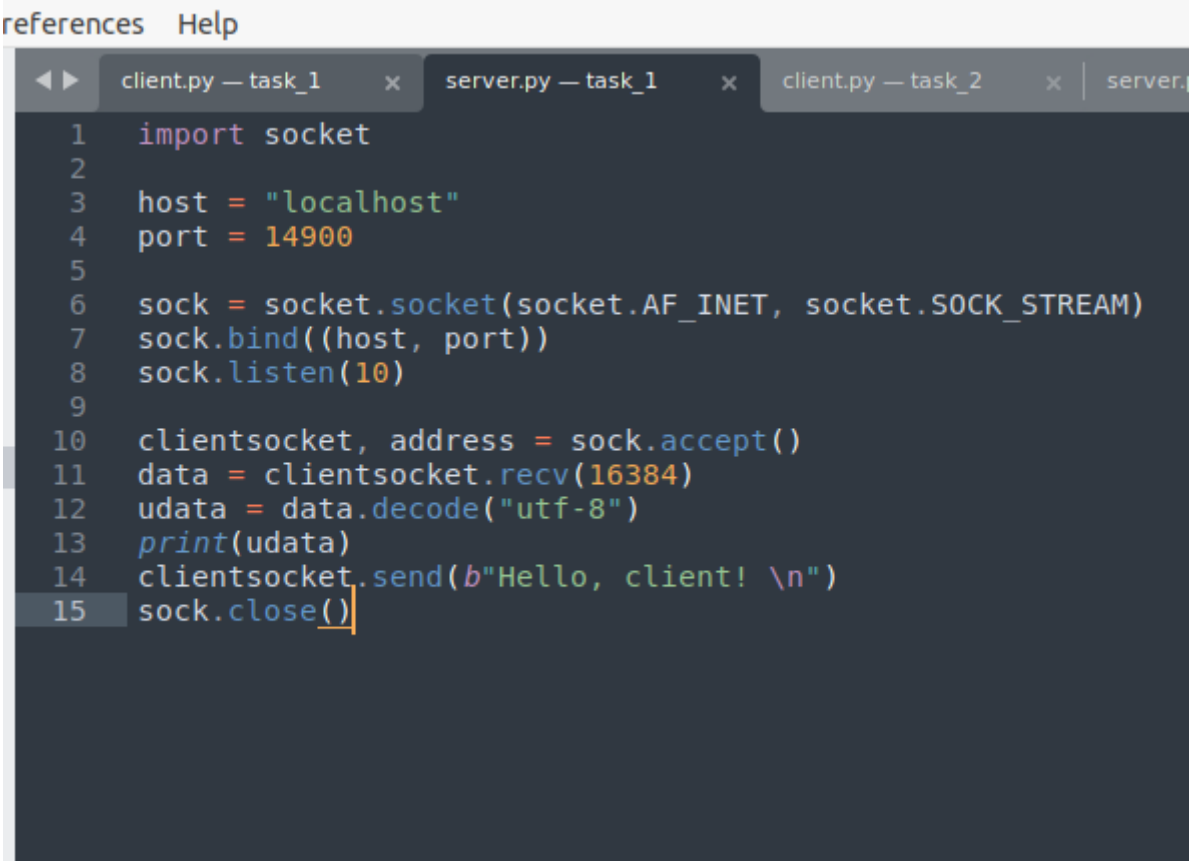
Цель работы:

Реализация клиентской и серверной части приложений

Выполнение:

Задание 1.

Сервер



Клиент

client.py — task_1 | server.py — task_1 | client.py — task_2 | server.py —

```python
import socket

host = "localhost"
port = 14900

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((host, port))
sock.send(b"Hello, server! \n")

data = sock.recv(16384)
print(data.decode("utf-8"))
sock.close()
```

Пример работы

ivan@ivan-VirtualBox: ~/web/ITMO_ICT_WebDevelopment_...

```
ivan@ivan-VirtualBox:~$ cd web/ITMO_ICT_WebDevelopment_2021-2022/
.git/     students/
ivan@ivan-VirtualBox:~$ cd web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401
/Fomenko_Ivan/Lr1/task_1
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ alias python=python3
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ python -i client.py
Hello, client!

>>>
```

ivan@ivan-VirtualBox: ~/web/ITMO_ICT_WebDevelopment_...

```
ivan@ivan-VirtualBox:~/unix/lab_1/trasher$ ./trash.sh -l
ivan@ivan-VirtualBox:~/unix/lab_1/trasher$ cd ..
ivan@ivan-VirtualBox:~/unix/lab_1$ cd ..
civan@ivan-VirtualBox:~/unix$ cd ..
ivan@ivan-VirtualBox:~$ cd web/
ivan@ivan-VirtualBox:~/web$ cd ITMO_ICT_WebDevelopment_2021-2022/
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022$ cd
.git/     students/
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022$ cd students/K33401
/Fomenko_Ivan/
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan$ ls
Lr1
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan$ cd Lr1/
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1$ cd task_1
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ alias python=python3
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ python -i server.py
Hello, server!

>>>
```

Задание 2.

Сервер

client.py — task_1 | server.py — task_1 | client.py — task_2 | server.py — task_2

```python
import socket

host = "localhost"
port = 14900

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.bind((host, port))
sock.listen(10)

#Version with trapezoid

clientsocket, address = sock.accept()
clientsocket.send(b"Enter large trapezoid base:")
data = clientsocket.recv(16384)
large_base = int(data.decode())
clientsocket.send(b"Enter small trapezoid base:")
data = clientsocket.recv(16384)
small_base = int(data.decode())
clientsocket.send(b"Enter trapezoid height:")
data = clientsocket.recv(16384)
height = int(data.decode())
area = height * ((large_base + small_base) / 2)
clientsocket.send(f"Trapezoid area: {area}".encode())
sock.close()
```

Клиент

client.py — task_1 | server.py — task_1 | client.py — task_2 | serve

```python
import socket

host = "localhost"
port = 14900

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((host, port))

#Version with trapezoid

for i in range(3):
    data = sock.recv(16384)
    text = data.decode()
    print(text)
    proportions = input()
    sock.send(proportions.encode())

data = sock.recv(16384)
trapezoid_area = data.decode()
print(trapezoid_area)
sock.close()
```

Пример работы

```
ivan@ivan-VirtualBox: ~/web/ITMO_ICT_WebDevelopment_...
ivan@ivan-VirtualBox:~$ cd web/ITMO_ICT_WebDevelopment_2021-2022/
.git/     students/
ivan@ivan-VirtualBox:~$ cd web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401
/Fomenko_Ivan/Lr1/task_1
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ alias python=python3
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ python -i client.py
Hello, client!

>>>
[1]+  Остановлен    python3 -i client.py
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_1$ cd ..
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1$ cd task_2
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_2$ python -i server.py
>>>
```

```
ivan@ivan-VirtualBox: ~/web/ITMO_ICT_WebDevelopment_...
ivan@ivan-VirtualBox:~$ cd web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401
/Fomenko_Ivan/Lr1/task_2
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_2$ alias python=python3
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_2$ python -i client.py
Enter large trapezoid base:
5
Enter small trapezoid base:
3
Enter trapezoid height:
10
Trapezoid area: 40.0
>>>
```

Задание 3

Server:

server.py

```python
import socket
import mimetypes
import sys
from email.parser import Parser
from functools import lru_cache
from urllib.parse import parse_qs, urlparse

MAX_LINE = 64 * 1024
MAX_HEADERS = 100

class MyHTTPServer:
    def __init__(self, host: str, port: int, server_name: str):
        self._host = host
        self._port = port
        self._server_name = server_name
        self._data: dict[str, list[str]] = {}

    def serve_forever(self):
        serv_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM, proto=0)

        try:
            serv_sock.bind((self._host, self._port))
            serv_sock.listen()

            while True:
                clientsocket, _ = serv_sock.accept()
                try:
                    self.serve_client(clientsocket)
                except Exception as e:
                    print('Client serving failed:', e)
        finally:
            serv_sock.close()

    def serve_client(self, clientsocket):
        try:
            req = self.parse_request(clientsocket)
            resp = self.handle_request(req)
            self.send_response(clientsocket, resp)
        except ConnectionResetError:
            clientsocket = None
        except Exception as e:
            print('Client serving failed:', e)

        if clientsocket:
            clientsocket.close()
```

```python
    def parse_request(self, clientsocket):
        rfile = clientsocket.makefile('rb')
        raw = rfile.readline(MAX_LINE + 1)

        if len(raw) > MAX_LINE:
            raise Exception('Request line is too long')

        req_line = str(raw, 'iso-8859-1')
        req_line = req_line.rstrip('\r\n')
        words = req_line.split()
        if len(words) != 3:
            raise Exception('Malformed request line')

        method, target, ver = words

        if ver != 'HTTP/1.1':
            raise Exception('Unexpected HTTP version')

        headers = self.parse_headers(rfile)
        host = headers.get('Host')
        if not host:
            raise Exception('Bad request')
        if host not in (self._server_name, f'{self._server_name}:{self._port}'):
            raise Exception('Not found')


        return Request(method, target, ver, headers, rfile)

    def parse_headers(self, rfile):
        headers = []
        while True:
            line = rfile.readline(MAX_LINE + 1)
            if len(line) > MAX_LINE:
                raise Exception('Header line is too long')

            if line in (b'\r\n', b'\n', b''):
                break

            headers.append(line)
            if len(headers) > MAX_LINE:
                raise Exception('Too many headers')
        sheaders = b''.join(headers).decode('iso-8859-1')

        return Parser().parsestr(sheaders)
```

```python
def handle_request(self, req):
    if req.path == '/rating' and req.method == 'POST':
        return self.handle_post_grades(req)

    if req.path.startswith('/rating/') and req.method == 'GET':
        subject_name = req.path[len('/rating/'):]
        return self.handle_get_rating(req, subject_name)

    raise Exception('Request not found')

def handle_post_grades(self, req):
    subject_name = req.query['subject_name'][0]
    grade = req.query['grade'][0]
    if subject_name in self._data.keys():
        self._data[subject_name].append(grade)
    else:
        new_subject = []
        new_subject.append(grade)
        self._data[subject_name] = new_subject

    return Response(204, 'Created')


def handle_get_rating(self, req, subject_name):
    if subject_name not in self._data.keys():
        raise Exception('Request not found')

    content_type = 'text/html; charset=utf-8'
    body = '<html><head></head><body>'
    ans = f'{subject_name}: '

    for grade in self._data[subject_name]:
        ans += grade + ' '

    body += f'<div>{ans}</div>'
    body += '</body></html>'
    body = body.encode('utf-8')
    headers = [('Content-Type', content_type), ('Content-Length', len(body))]
    return Response(200, 'OK', headers, body)
```

```python
    def send_response(self, clientsocket, response):
        wfile = clientsocket.makefile('wb')
        status_line = f'HTTP/1.1 {response.status} {response.reason}\r\n'
        wfile.write(status_line.encode('iso-8859-1'))

        if response.headers:
            for (key, value) in response.headers:
                header_line = f'{key}: {value}\r\n'
                wfile.write(header_line.encode('iso-8859-1'))

        wfile.write(b'\r\n')

        if response.body:
            wfile.write(response.body)

        wfile.flush()
        wfile.close()

class Request:
    def __init__(self, method, target, version, headers, rfile):
        self.method = method
        self.target = target
        self.version = version
        self.headers = headers
        self.rfile = rfile

    @property
    def path(self):
        return self.url.path

    @property
    @lru_cache(maxsize=None)
    def query(self):
        return parse_qs(self.url.query)

    @property
    @lru_cache(maxsize=None)
    def url(self):
        return urlparse(self.target)


class Response:
    def __init__(self, status, reason, headers=None, body=None):
        self.status = status
        self.reason = reason
        self.headers = headers
        self.body = body
```

```python
if __name__ == '__main__':
    host = sys.argv[1]
    port = int(sys.argv[2])
    name = sys.argv[3]
    server = MyHTTPServer(host, port, name)

    try:
        server.serve_forever()
    except KeyboardInterrupt:
        pass
```

Пример работы:

```
^Civan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fomenko_Ivan/Lr1/task_3$ python server.py 127.0.0.1 14900 example.local
^Civan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fomenko_Ivan/Lr1/task_3$
```

```
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_3$ nc localhost 14900
POST /rating?subject_name=Math&grade=5 HTTP/1.1
Host: example.local

HTTP/1.1 204 Created

^C
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_3$ nc localhost 14900
POST /rating?subject_name=Russian&grade=3 HTTP/1.1
Host: example.local

HTTP/1.1 204 Created

^C
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_3$ nc localhost 14900
POST /rating?subject_name=Math&grade=4 HTTP/1.1
Host: example.local

HTTP/1.1 204 Created

^C
```

```
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_3$ nc localhost 14900
GET /rating/Math HTTP/1.1
Host: example.local

HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 60

<html><head></head><body><div>Math: 5 4 </div></body></html>
^C
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_3$ nc localhost 14900
GET /rating/Russian HTTP/1.1
Host: example.local

HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 61

<html><head></head><body><div>Russian: 3 </div></body></html>
^C
ivan@ivan-VirtualBox:~/web/ITMO_ICT_WebDevelopment_2021-2022/students/K33401/Fom
enko_Ivan/Lr1/task_3$
```

Задание 4

Server

server.py — task_3 | client.py | client1.py | server.py — task_4

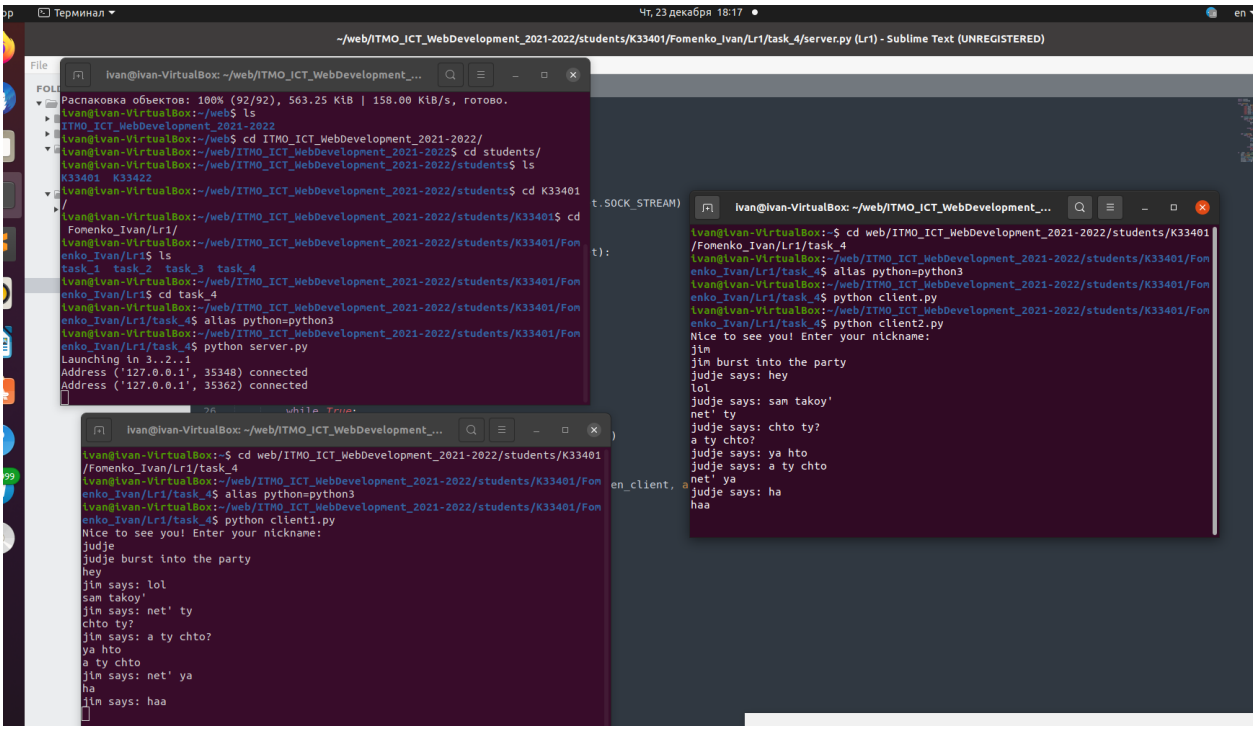
```python
import socket
import threading

class Server:
    def __init__(self, host: str, port: int):
        self.host = host
        self.port = port
        self.users = []
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.bind((self.host, self.port))
        self.sock.listen(10)

    def listen_client(self, client_socket: socket.socket):
        client_socket.settimeout(60)
        while True:
            try:
                data = client_socket.recv(1024)
                for cliento in self.users.copy():
                    if cliento != client_socket:
                        cliento.send(data)
            except KeyboardInterrupt:
                self.sock.close()


    def run(self):
        while True:
            try:
                client_socket, address = self.sock.accept()
                print(f"Address {address} connected")
                if client_socket not in self.users:
                    self.users.append(client_socket)
                thread = threading.Thread(target=self.listen_client, args=(client_socket,))
                thread.start()
            except KeyboardInterrupt:
                self.sock.close()
                break

if __name__ == '__main__':
    host = "localhost"
    port = 14900
    print("Launching in 3..2..1")
    server = Server(host, port)
    thread1 = threading.Thread(target=server.run)
    thread1.start()
```

Client

```python
import socket
import threading

def welcome():
    print("Nice to see you! Enter your nickname: ")
    nickname = input()
    print(f"{nickname} burst into the party")
    return nickname

class Client:
    def __init__(self, host: str, port: int, username: str):
        self.host = host
        self.port = port
        self.username = username
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    def get_message(self):
        while True:
            try:
                data = self.sock.recv(1024)
                udata = data.decode("utf-8")
                print(udata)
            except KeyboardInterrupt:
                self.sock.close()

    def send_message(self):
        while True:
            try:
                data = input()
                self.sock.send(f"{self.username} says: {data}".encode())
            except KeyboardInterrupt:
                self.sock.close()

    def run(self):
        self.sock.connect((self.host, self.port))
        thread2, thread1 = threading.Thread(target=self.get_message), threading.Thread(target=self.send_message)
        thread1.start()
        thread2.start()
```

```python
from client import Client, welcome

if __name__ == '__main__':
    host = "localhost"
    port = 14900
    username = welcome()
    client = Client(host, port, username)
    client.run()
```

Пример работы:



Выводы:

Поддерживать чат довольно интересно, а в HTTP запросах уже давно за нас сделали парсеры