МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет
о лабораторной работе №1
по дисциплине
«Web программирование»

Выполнила
Голуб А.Л.
группа К33421

Проверил
Говоров А. И.

Санкт-Петербург
2021

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.

Код

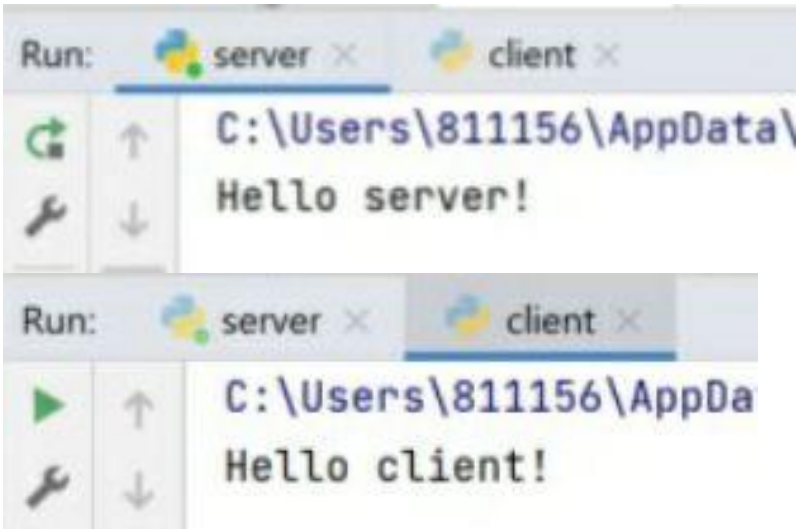
server.py

```
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.bind(('127.0.0.1', 9000))
conn.listen(10)

while True:
    try:
        # receiving client's message
        client_socket, address = conn.accept()
        data = client_socket.recv(16384)
        data = data.decode('utf-8')
        print(data)

        # sending a response
        client_socket.send(b'Hello client! \n')

    except KeyboardInterrupt:
        conn.close()
        break
```

client.py

```
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect(("127.0.0.1", 9000))

# sending message to server
conn.send(b'Hello server! \n')

# receiving server's response
data = conn.recv(16384)
data = data.decode('utf-8')
print(data)

conn.close()
```

Скриншоты выполнения программы



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. Математическая операция – поиск площади трапеции.

Код

server.py

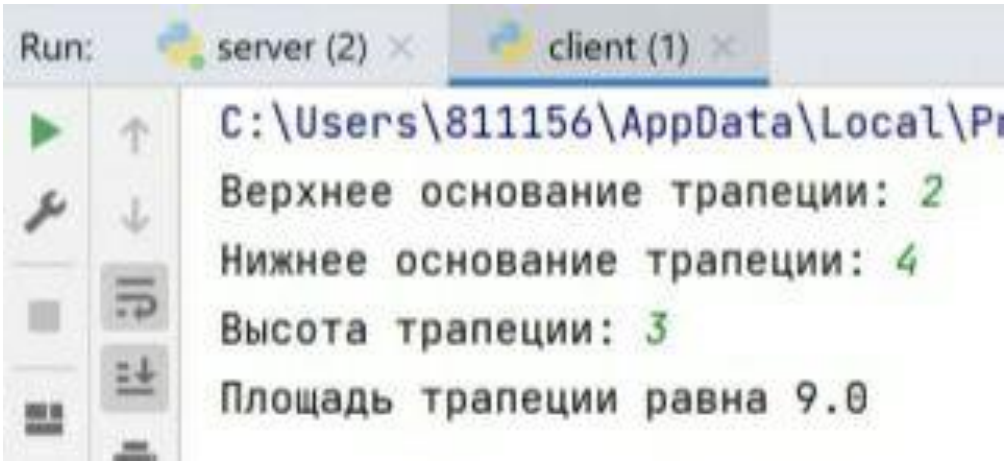
```
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.bind(('127.0.0.1', 9000))
conn.listen(10)

while True:
    try:
        # receiving client's message
        client_socket, address = conn.accept()
        data = client_socket.recv(16384)
        data = data.decode('utf-8')
        a, b, h = map(int, data.lstrip().rstrip().split())
        print('a =', a)
        print('b =', b)
        print('h =', h)

        # calculating and sending response
        s = 0.5 * (a + b) * h
        s = str(s).encode()
        client_socket.send(s)
```

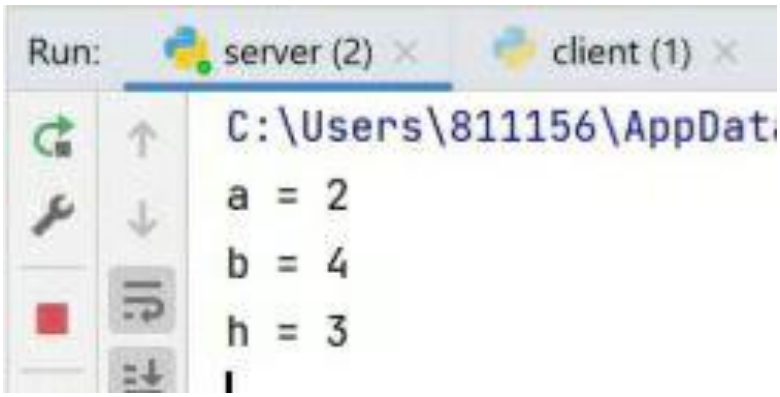
```
    except KeyboardInterrupt:
        conn.close()
        break
```

client.py

```
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect(("127.0.0.1", 9000))

# sending message to server
a = input('Верхнее основание трапеции: ')
b = input('Нижнее основание трапеции: ')
h = input('Высота трапеции: ')
message = ' '.join([str(a), str(b), str(h)]).encode()
conn.send(message)

# receiving server's response
data = conn.recv(16384)
data = data.decode('utf-8')
print('Площадь трапеции равна', data)

conn.close()
```

Скриншоты выполнения программы





Задание 3

Реализовать серверную часть приложения. Клиент подключается к серверу. В ответ клиент получает http-сообщение, содержащее html-страницу, которую сервер подгружает из файла index.html.

Код

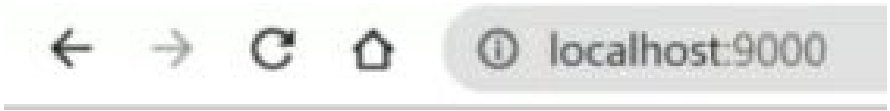
server.py

```
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.bind(('127.0.0.1', 9000))
conn.listen(10)

while True:
    try:
        client_socket, address = conn.accept()
        with open('index.html', 'r') as index_file:
            message = index_file.read()
        client_socket.send(b'HTTP/1.1 200 OK\n' + b'Content-Type:
text/html\n' +
                           b'\n' + message.encode())
    except KeyboardInterrupt:
        conn.close()
        break
```

client.py

```
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect(("127.0.0.1", 9000))
conn.close()
```

Скриншоты выполнения программы



Задание 4

Реализовать двухпользовательский или многопользовательский чат.

Код

server.py

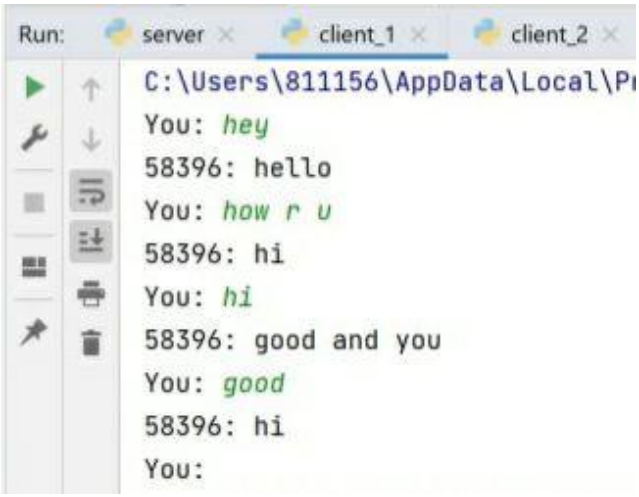
```
import socket
from threading import *


# chat thread class
class ChatThread(Thread):
    def __init__(self, conn, client_name):
        Thread.__init__(self)
        self.conn = conn
        self.client_name = client_name

    def run(self):
        while True:
            try:
                # receive and broadcast message
                message = self.conn.recv(1024)
                message = self.client_name + ': ' + message.decode()
                print(message)
                broadcast(message, self.client_name)
            except:
                self.conn.close()


# broadcast message to all other chat members
def broadcast(message, author):
    for client in client_list:
        if client.client_name != author:
            try:
                client.conn.send(message.encode())
            except:
                client.conn.close()
                client_list.remove(client)


# main part
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('127.0.0.1', 9000))
server.listen(10)

client_list = []
while True:
    try:
        # connect to a new client
        conn, addr = server.accept()
        print(addr, 'connected')

        # create a new thread
        new_thread = ChatThread(conn, str(addr[1]))
        client_list.append(new_thread)
        new_thread.start()

    except KeyboardInterrupt:
        server.close()
```

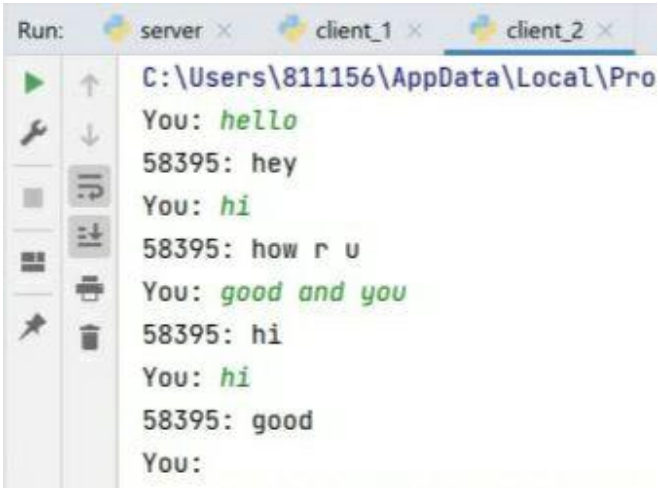
```
        for client in client_list:
            client.conn.close()
```

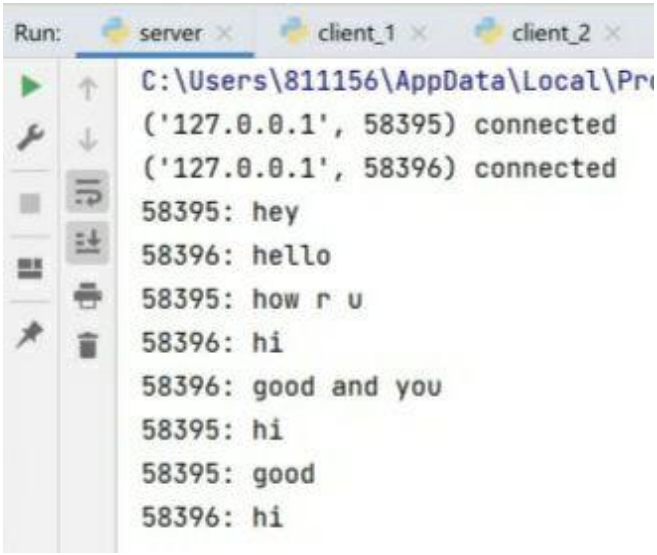
client.py

```
import socket
import sys

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('127.0.0.1', 9000))

sockets_list = [sys.stdin, client]

while True:
    try:
        # read text from input and send it
        message = input('You: ')
        client.send(message.encode())

        # receive message from server
        message = client.recv(1024)
        message = message.decode()
        print(message)

    except KeyboardInterrupt:
        client.close()

    except:
        continue
```

Скриншоты выполнения программы




Run:
server
client_1
client_2
C:\Users\811156\AppData\Local\Pro
You: hello
58395: hey
You: hi
58395: how r u
You: good and you
58395: hi
You: hi
58395: good
You:


Run:
server
client_1
client_2
C:\Users\811156\AppData\Local\Pr
('127.0.0.1', 58395) connected
('127.0.0.1', 58396) connected
58395: hey
58396: hello
58395: how r u
58396: hi
58396: good and you
58395: hi
58395: good
58396: hi