МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Кафедра Интеллектуальных систем в гуманитарной сфере,

факультет Инфокоммуникационных технологий

Лабораторная работа №1

Работа с сокетами

Выполнил

студент гр. № K3242

Кузьмичев Кирилл Максимович

Проверил преподаватель

Говоров А.И.

Санкт-Петербург
2020

Задание №1

Цель: овладеть практическими навыками и умениями реализации web-серверов и использования сокетов.
Оборудование: компьютерный класс.
Программное обеспечение: Python 2.7-3.6, библиотеки Python: sys, socket.

Текст задания:
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:

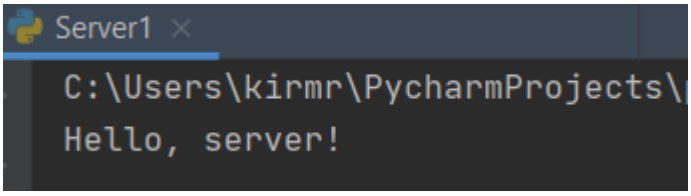
Client1.py

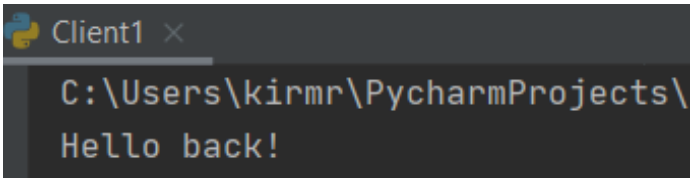
```python
import socket

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.connect(("127.0.0.3", 11042))
Sock.send(b"Hello, server! \n")

data = Sock.recv(1024)
Sock.close()

print(data.decode("utf-8"))
```

Server1.py

```python
import socket

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.bind(("127.0.0.3", 11042))
Sock.listen(1)
conn, addr = Sock.accept()

while True:
    data = conn.recv(1024)
    if not data:
        break
    print(data.decode("utf-8"))
    conn.send(b'Hello back!')

conn.close()
```

Результат:





Задание №2

Текст задания:
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. (Номер в ИСУ: 3; с. Поиск площади трапеции.)

Код:

Client2.py

```python
import socket

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect(('127.0.0.1', 8888))

while True:
    data = sock.recv(1024)
    if not data:
        sock.close()
        break
    print(data.decode("utf-8"))
    sock.sendall(bytes(input(), "utf-8"))

sock.close()
```

Server2.py

```python
import socket
from math import sqrt

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.bind(("127.0.0.1", 8888))
sock.listen(1)
conn, addr = sock.accept()

print('connected:', addr, conn)
d = ''
option = ''
while True:
    conn.sendall(bytes(d + f'Каким способом хотите найти площадь трапеции: \n (a) - Через основания и высоту,
    data = conn.recv(1024)
    option = data.decode()

    if option == 'a':
        conn.sendall(bytes(d + f'Введите значения двух оснований(a, b) и высоты(h): ', "utf-8"))
        data = conn.recv(1024)
        print(data.decode())
        a, b, h = data.decode("utf-8").split(' ')
        S = (int(h) * (int(a)+int(b)))/2
        d = f'Площадь равна = {S} \n'

    if option == 'b':
        conn.sendall(bytes(d + f'Введите значения средней линии(m) и высоты(h): ', "utf-8"))
        data = conn.recv(1024)
        print(data.decode())
        m, h = data.decode("utf-8").split(' ')
        S = int(m) * int(h)
        d = f'Площадь равна = {S} \n'

    if not data:
        break
    sock.close()
```

Результат:

Server2 Client2

```
C:\Users\kirmr\PycharmProjects\pythonProject\venv\Scripts\python.exe
Каким способом хотите найти площадь трапеции:
 (a) - Через основания и высоту, (b) - Через среднюю линию и высоту
a
Введите значения двух оснований(a, b) и высоты(h):
10 5 5
Площадь равна = 37.5
```

Server2 Client2

```
C:\Users\kirmr\PycharmProjects\
connected: ('127.0.0.1', 64448)
10 5 5
```

Задание №3

Текст задания:
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:

Client3.py

```python
import socket

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.connect(("127.0.0.1", 11042))
Sock.send('Hello, server!'.encode("utf-8"))

data = Sock.recv(1024)
Sock.close()

print(data.decode("utf-8"))
```

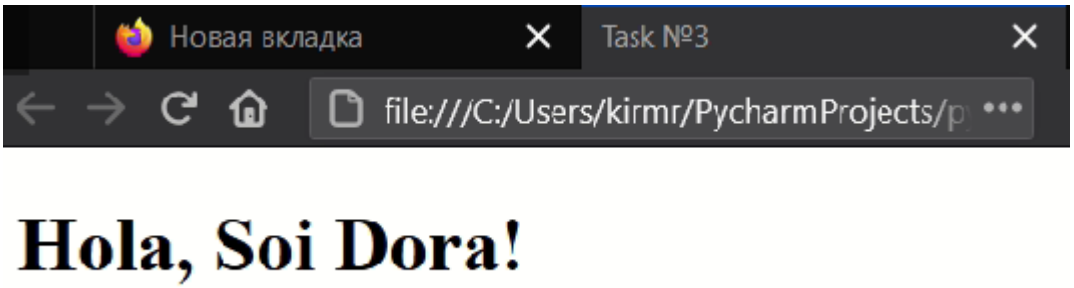
Server3.py

```python
import socket

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.bind(("127.0.0.1", 11042))
Sock.listen(1)
conn, addr = Sock.accept()

while True:
    with open('index.html', 'r') as file:
        browse = file.read()
        conn.sendall(bytes(f'{browse}', 'utf-8'))
    data = conn.recv(1024)
    if not data:
        break

conn.close()
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Hola, soi Dora!</title>
</head>
<body>

</body>
</html>
```

Результат:



Hola, Soi Dora!

Задание №4

Текст задания:
Реализовать двух пользовательский или многопользовательский чат. Реализация многопользовательского часа позволяет получить максимальное количество баллов.

Код:

Client4_1.py

```python
import socket
import threading

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.connect(("127.0.0.2", 11042))

def send():
    while True:
        text = input()
        Sock.send(b"Kirill: " + text.encode("utf-8"))

def get():
    while True:
        feedback = Sock.recv(1024)
        print(feedback.decode("utf-8"))

threading.Thread(target = send).start()
threading.Thread(target = get).start()
```

Client4_2.py

```python
import socket
import threading

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.connect(("127.0.0.2", 11042))

def send():
    while True:
        text = input()
        Sock.send(b"Polina: " + text.encode("utf-8"))

def get():
    while True:
        feedback = Sock.recv(1024)
        print(feedback.decode("utf-8"))

threading.Thread(target = send).start()
threading.Thread(target = get).start()
```

Client4_3.py

```python
import socket
import threading

Sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
Sock.connect(("127.0.0.2", 11042))

def send():
    while True:
        text = input()
        Sock.send(b"Danil: " + text.encode("utf-8"))

def get():
    while True:
        feedback = Sock.recv(1024)
        print(feedback.decode("utf-8"))

threading.Thread(target = send).start()
threading.Thread(target = get).start()
```

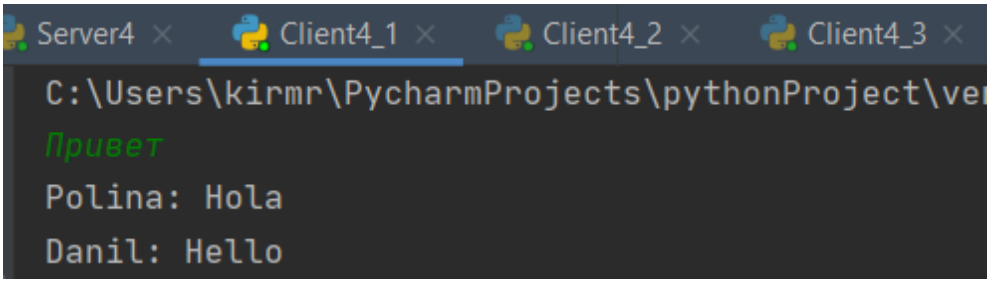
Server4.py

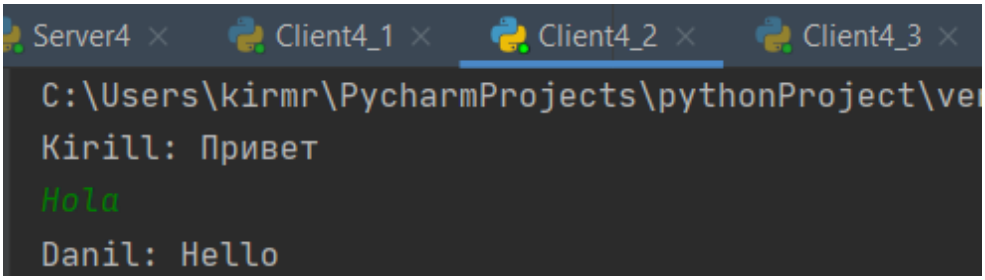
```python
import threading

S = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
S.bind(("127.0.0.2", 11042))
S.listen(3)
Account = []

def add_clients():
    while True:
        Sock, addr = S.accept()
        Account.append(Sock)
        threading.Thread(target = chat,  args = [Sock, addr]).start()

def chat(Sock, addr):
    print(addr)
    while True:
        try:
            feedback = Sock.recv(1024)
            if not feedback:
                break
            for client in Account:
                if client == Sock:
                    continue
                client.sendall(feedback)
        except Exception:
            Account.remove(Sock)
            break
    Sock.close()

threading.Thread(target=add_clients()).start()
```

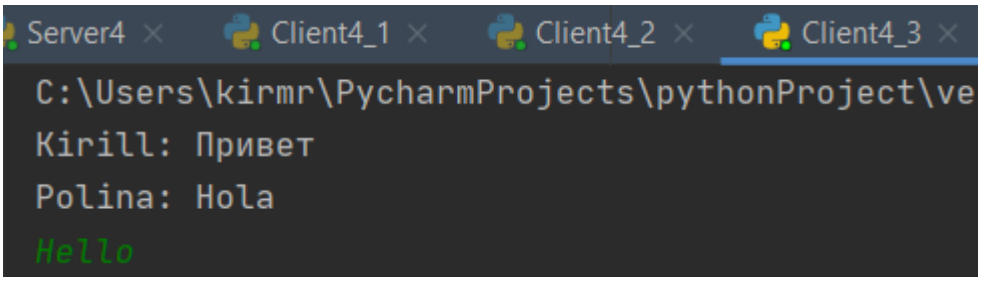
Результат:

Server4 | Client4_1 | Client4_2 | Client4_3

```
C:\Users\kirmr\PycharmProjects\pythonProject\venv
('127.0.0.1', 64476)
('127.0.0.1', 64477)
('127.0.0.1', 64484)
```







Вывод:

В ходе выполнения заданий данной работы были получены практические навыки работы, настройки и реализации web-серверов с использованием сокетов; реализации клиентской и серверной части приложения; реализации двухпользовательского или многопользовательского чата.