ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет «Инфокоммуникационных технологий»
Направление подготовки «09.03.03 Мобильные и сетевые технологии»

О Т Ч Е Т

Тема задания: Работа с сокетами

Выполнил:
Студент Шкикавый И.Я. К33402

Проверил:
Преподаватель Говоров А. И.

Санкт-Петербург
2020

Задание 1.
Было необходимо реализовать клиентскую и серверную часть приложения. Клиент должен отсылать сообщение серверу, оно в свою очередь должно отвечать сообщением.
Реализация:

```python
import socket

sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.bind(('localhost', 8080))
sckt.listen(10)

while True:
    clientsocket, address = sckt.accept()
    print('connected:', address)
    data = clientsocket.recv(1024)
    if not data:
        break
    print(data.decode("utf-8"))
    clientsocket.sendall(b'Hello, client!')

clientsocket.close()
```

Рисунок 1 - Код сервера первого задания

```python
import socket

sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.connect(('localhost', 8080))
sckt.sendall(b'Hello, server!')

data = sckt.recv(1024)
print(data.decode("utf-8"))

sckt.close()
```

Рисунок 2 - Код клиента первого задания

```
C:\Users\Иван\AppData\Local\Programs\
connected: ('127.0.0.1', 63122)
Hello, server!
```

Рисунок 3 - вывод сервера

```
C:\Users\Иван\AppData\Local\P
Hello, client!
```

Рисунок 4 - вывод клиента

Задание 2.
Необходимо было реализовать клиент-серверное приложение решения математической задачи. В данном варианте необходимо было решить «Поиск площади параллелограмма»
Реализация:

```
import socket

sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.bind(('localhost', 8080))
sckt.listen(1)

while True:
    clientsocket, address = sckt.accept()
    clientsocket.sendall(bytes(f'Введите основание и высоту', "utf-8"))
    try:
        data = clientsocket.recv(1024)
        if not data:
            break
        a, h = data.decode("utf-8").split(' ')
        s = int(a) * int(h)
        clientsocket.sendall(bytes(f'Площадь параллелограмма {s}', "utf-8"))

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

Рисунок 5 - код сервера 2-го задания

```
import socket

sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.connect(('localhost', 8080))

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

Рисунок 6 - код клиента 2-го задания

```
C:\Users\Иван\AppData\Local\Programs\Pyth
Введите основание и высоту
5 2
Площадь параллелограмма 10
```

Рисунок 7 - вывод клиента

Задание 3.
Клиент должен подключаться к серверу и получать http-сообщение, содержащее интернет страницу
Реализация:

```python
import socket

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.bind(('localhost', 8080))
sock.listen(1)
while True:
    clientsocket, address = sock.accept()
    with open('index.html', 'r') as file:
        html = file.read()
        clientsocket.sendall(bytes(f'HTTP/1.0 200 OK\nContent-Type: text/html\n\n{html}', 'utf-8'))
    data = clientsocket.recv(1024)
    if not data:
        break

clientsocket.close()
```

Рисунок 8 - код сервера 3-его задания

```python
import socket

sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.connect(('localhost', 8080))

while True:
    data = sckt.recv(1024)
    if not data:
        break
    print(data.decode("utf-8"))

sckt.close()
```

Рисунок 9 - код клиента 3-его задания

```html
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Task 3</title>
</head>
<body>
        <h1>Zdarova, otec</h1>
</body>
</html>
```

Рисунок 10 - html страница

```
HTTP/1.0 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Task 3</title>
</head>
<body>
        <h1>Zdarova, otec</h1>
</body>
</html>
```

Рисунок 11 - вывод результата

Задание 4.
Было необходимо реализовать многопользовательский чат
Реализация:

```
import socket
import threading
sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.connect(('localhost', 9090))
name = input('Введите ник: ')
def send_message():
    try:
        while True:
            text = input()
            sckt.sendall(bytes(name + ": " + text, "utf-8"))
            if text == "X":
                sckt.close()
                break
    except Exception:
        pass
def get_message():
    try:
        while True:
            data = sckt.recv(1024).decode("utf-8")
            if not data:
                break
            print(data)
        sckt.close()
    except Exception:
        pass
threading.Thread(target=send_message).start()
threading.Thread(target=get_message).start()
```

Рисунок 12 - код клиента чата

```
import socket
import threading
sckt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sckt.bind(('127.0.0.1', 9090))
sckt.listen(10)
users = set()
def connect_users():
    while True:
        clientsocket, addr = sckt.accept()
        users.add(clientsocket)
        print('connected:' + str(addr))
        threading.Thread(target=message, args=[clientsocket, addr]).start()
def message(clientsocket, addr):
    print('typing' + str(addr))
    while True:
        try:
            data = clientsocket.recv(1024)
            if not data:
                break
            for user in users:
                if user == clientsocket:
                    continue
                user.sendall(data)
        except Exception:
            users.remove(clientsocket)
            break
    print('left chat' + str(addr))
    clientsocket.close()
threading.Thread(target=connect_users()).start()
```

Рисунок 13 - код сервера чата

```
Введите ник: Ваня
Привет, ребят
Матвей: Привет, Вань
Миша: Всем привет
Как дела?
Матвей: Все хорошо, только мне пора спешить, я пойду тогда, хорошо?
Матвей: Пока
Матвей: X
Миша: Ладно, хорошо, пока тогда
Миша: X
Пока
X
```

Рисунок 14 - чат с клиентской части

```
connected:('127.0.0.1', 50340)
typing('127.0.0.1', 50340)
connected:('127.0.0.1', 50341)
typing('127.0.0.1', 50341)
connected:('127.0.0.1', 50342)
typing('127.0.0.1', 50342)
left chat('127.0.0.1', 50341)
left chat('127.0.0.1', 50342)
left chat('127.0.0.1', 50340)
```

Рисунок 15 - чат с серверной части

Вывод: в ходе выполнения лабораторной работы я познакомился с принципами работы с сокетами на языке Python, выполняя задачи разного уровня направленности.