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: Шутов Д. Э.

Группа: К33402

Преподаватель: Говоров А. И.

Дата: 07.10.21

Санкт-Петербург, 2021

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

Ход работы.

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

```python
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.bind(('127.0.0.1', 5000))
conn.listen(10)

clientsocket, address = conn.accept()
data = clientsocket.recv(16384)
data = data.decode('utf-8')
print(data)

clientsocket.send(b'Hello, client! \n')
conn.close()
```

client.py

```python
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect(("127.0.0.1", 5000))
conn.send(b'Hello, server! \n')
data = conn.recv(16384)
data = data.decode('utf-8')
print(data)
conn.close()
```

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.

a. Теорема Пифагора

server.py

```python
# Option "a". Pythagorean theorem
import socket
import math

server = socket.socket()
host = '127.0.0.1'
port = 5000
server.bind((host, port))
server.listen(3)

conn, address = server.accept()
first_side_bin = conn.recv(200)
second_side_bin = conn.recv(200)
first_side = first_side_bin.decode('utf-8')
second_side = second_side_bin.decode('utf-8')
hypotenuse = math.sqrt(int(first_side) ** 2 + int(second_side
conn.send(str(hypotenuse).encode())
conn.close()
```

client.py

```python
# Option "a". Pythagorean theorem
import socket

conn = socket.socket()

conn.connect(("127.0.0.1", 5000))
first_side = input("Enter the length of the first side: ")
second_side = input("Enter the length of the second side: ")
conn.send(first_side.encode())
conn.send(second_side.encode())
hypotenuse_bin = conn.recv(200)
hypotenuse = hypotenuse_bin.decode('utf-8')
print('The hypotenuse of a triangle is:', hypotenuse)
conn.close()
```

3.

server.py index.html

```python
import socket
import sys


class MyHTTPServer:

    def __init__(self, host, port):
        self.host = host
        self.port = port

    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):
        # Client connection handling
        data = clientsocket.recv(16384)
        data = data.decode('utf-8')
        url, method, headers, body = self.parse_request(data)
        resp = self.handle_request(url, method, body)
        if resp:
            self.send_response(clientsocket, resp)
```

```
28
29      def parse_request(self, data):
30          #processing the http + request header
31          data = data.replace('\r', '')
32          lines = data.split('\n')
33          method, url, protocol = lines[0].split()
34          i = lines.index('')
35          headers = lines[1:i]
36          body = lines[-1]
37          return url, method, headers, body
38
39      def handle_request(self, url, method, body):
40          # url processing
41          if url == "/":
42              if method == "GET":
43                  resp = "HTTP/1.1 200 OK\n\n"
44                  with open('index.html', 'r') as f:
45                      resp += f.read()
46                  return resp
47              if method == "POST":
48
49                  newbody = body.split('&')
50                  for a in newbody:
51                      if a.split('=')[0] == 'subject':
52                          subjects.append(a.split('=')[1])
53                      if a.split('=')[0] == 'mark':
54                          marks.append(a.split('=')[1])
55
56                  resp = "HTTP/1.1 200 OK\n\n"
```

```
                resp += "<html><head><title>Journal</title></head><body><table border=1>"
                for s, m in zip(subjects, marks):
                    resp += f"<tr><td>{s}</td><td>{m}</td></tr>"
                resp += "</table></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)
    subjects = []
    marks = []
    try:
        serv.serve_forever()
    except KeyboardInterrupt:
        pass
```

server.py | index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Journal</title>
</head>
<body>
<form action="/" method="post">
    <div>
        <label for="name">Subject:</label>
        <input type="text" id="name" name="subject"/>
    </div>
    <div>
        <label for="mail">Mark:</label>
        <input type="number" id="mail" name="mark"/>
    </div>
    <div>
        <input type="submit">
    </div>

</body>
</html>
```

4. Реализовать многопользовательского чата.

client.py

```python
import socket
from threading import Thread


def send():
    while True:
        message = input()
        conn.send(message.encode('utf-8'))


def get():
    while True:
        data = conn.recv(16384)
        data = data.decode('utf-8')
        print(data)


conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect(("127.0.0.1", 5000))

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

server.py

```python
import socket
from threading import Thread

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.bind(('127.0.0.1', 5000))
conn.listen(10)


def send(message, sender):
    for client in clients:
        if sender != client:
            client.send(message)


def listen(client):
    while True:
        message = client.recv(1024)
        send(message, client)


clients = []
while True:

    new_client, _ = conn.accept()

    if new_client not in clients:
        clients.append(new_client)

    Thread(target=listen, args=[new_client]).start()
```

Вывод: Я научился реализовывать веб-сервер с помощью библиотеки socket в языке программирование python.