Федеральное государственное автономное образовательное учреждение
высшего образования
"Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики"

Лабораторная работа №1

Вариант №6

Группа: K33401

Студент: Клишин Никита Дмитриевич

Санкт-Петербург, 2020.

Выполнение заданий:

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

```python
import socketserver


class MyTCPHandler(socketserver.StreamRequestHandler):

    # обработчик входящих соединений
    def handle(self):
        self.data = self.request.recv(1024)
        value = bytes.decode(self.data)
        print("CLIENT SEND: " + value)

        self.request.sendall(b"Hello, client")

if __name__ == "__main__":
    # создание сервера
    server = socketserver.TCPServer(("172.18.224.1", 3000), MyTCPHandler)

    print("Starting TCP server...")
    try:
        server.serve_forever()
    except KeyboardInterrupt:
        print("Stopping server...")
```

   Client.py

```python
import socket

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect(("172.18.224.1", 3000))
conn.send(str.encode("Hell  o, server"))
response = conn.recv(1024).decode("UTF-8")
print("Server response: " + response)
```

   Вывод программы:

```
nikita@DESKTOP-M6TP2JK:~/ITMO/WebProgramming/lab_01/task_01$ python3 client.py
Server response: Hello, client
```

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. Вариант: Решение квадратного уравнения.

Код:

```
import socket
import math

HOST = "127.0.0.1"
PORT = 3000

print("Starting...")
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind((HOST, PORT))
server_socket.listen(10)

print("Work...")
while True:
    try:
        # создание серверного сокета
        client_socket, client_addr = server_socket.accept()
        data = client_socket.recv(1024)
        data = data.decode(encoding="UTF-8")
        result = str()
        a, b, c = map(float, data.split(' '))
        d = b**2 - 4*a*c
        if d < 0:
            result = "Нет корней"
        elif d == 0:
            x_1_2 = -b / (2 * a)
            result = "Один корень: " + str(x_1_2)
        else:
            x_1 = (-b - math.sqrt(d)) / (2 * a)
            x_2 = (-b + math.sqrt(d)) / (2 * a)
            result = "Два корня: " + str(x_1) + ' ' + str(x_2)
        result = result.encode(encoding="UTF-8")
        client_socket.sendall(result)
    except socket.error:
        client_socket.close()
    except KeyboardInterrupt:
        print("Stopping...")
        break
```

Вывод программы:

```
(webEnv) nikita@DESKTOP-M6TP2JK:~/ITMO/WebProgramming/lab_01/task_02$ python client.py
Введите коэффициенты
 квадратного уравнения:
a: 2
b: 2
c: -4
Два корня: -2.0 1.0
```

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

```python
import socket
import os
import codecs

server = socket.socket()
host = "127.0.0.1"
port = 536
server.bind((host, port))

server.listen(5)
count = 0
while True:
    client, (client_host, client_port) = server.accept()
    req = client.recv(1024)
    req = req.decode("UTF-8")
    responce_type = "HTTP/1.0 200 OK\n"
    headers = 'Content-Type: text/html\n'
    responce_type = responce_type.encode()
    client.send(responce_type)
    headers = headers.encode()
    client.send(headers)
    index_file = open(os.path.join(os.path.dirname(__file__), 'index.html'), "r")
    http_response = index_file.read()
    body =  '''
     '''
    body += http_response
    print(body)
    client.sendall(body.encode("utf-8"))
    client.close()
```

Вывод программы:

```
    <html>
<head>
    <title>Document</title>
</head>
<body>
    Hello world!
</body>
</html>

    <html>
<head>
    <title>Document</title>
</head>
<body>
    Hello world!
</body>
</html>
```

4. Реализация многопользовательского чата.
   Пример работы программы:
   Server.py

```
(webEnv) nikita@DESKTOP-M6TP2JK:~/ITMO/WebProgramming/lab_01/task_04$ python server3.py
Starting...
Starting db_worker...
Serving...
Client (id: 0, addr: ('127.0.0.1', 53182)) connected
Created client worker for client (id: 0)
Клиент (id: 0) зарегистрирован: nikita
Push history to (id: 0 nickname: nikita)
Добавляем в историю...
Main: get_mail: Hello, everyone!
Send to 0: nikita|Hello, everyone!
Client (id: 1, addr: ('127.0.0.1', 53184)) connected
Created client worker for client (id: 1)
Клиент (id: 1) зарегистрирован: ivan
Push history to (id: 1 nickname: ivan)
Добавляем в историю...
Main: get_mail: Hello, nikita
Send to 0: ivan|Hello, nikita
Send to 1: ivan|Hello, nikita
Добавляем в историю...
Main: get_mail: How are you?
Send to 1: nikita|How are you?
Send to 0: nikita|How are you?
```

   Клиент 1:

```
(webEnv) nikita@DESKTOP-M6TP2JK:~/ITMO/WebProgramming/lab_01/task_04$ python client
.py
Connecting...
Enter your nickname: nikita
Connected successfuly
Hello, everyone!
nikita: Hello, everyone!
ivan: Hello, nikita
How are you?
nikita: How are you?
```

   Клиент 2:

```
(webEnv) nikita@DESKTOP-M6TP2JK:~/ITMO/WebProgramming/lab_01/task_04$ python client
.py
Connecting...
Enter your nickname: ivan
Connected successfuly
nikita: Hello, everyone!
Hello, nikita
ivan: Hello, nikita
nikita: How are you?
```

   При подключении клиента ему предлагается ввести свой никнейм. Затем клиенту подгружается история предыдущих сообщений в чате. При вводе одним из клиентов сообщения, оно отображается у остальных участников беседы с соответствующим никнеймом.