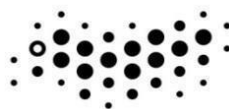


Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИТМО»**

Отчёт
по лабораторной работе №1
по дисциплине «**Web-Программирование**»

Автор: Глушков Кирилл Георгиевич
Факультет: Инфокоммуникационные технологии
Группа: K33422
Преподаватель:



УНИВЕРСИТЕТ ИТМО

Санкт-Петербург 2023

Практическое задание:

1. Реализовать клиентскую и серверную часть приложения. Клиент отправляет серверу сообщение «Hello, server». Сообщение должно отразиться на стороне сервера. Сервер в ответ отправляет клиенту сообщение «Hello, client». Сообщение должно отобразиться у клиента. Обязательно использовать библиотеку socket. Реализовать с помощью протокола UDP

Клиент

```
web > ITMO_ICT_WebDevelopment_2023-2024 > students > k33420 > Kirill_Glushkov > Ir1 > client.py > ...
1  import socket
2
3  # Создаем сокет
4  sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5
6  # Отправляем сообщение серверу
7  message = b'Hello, server'
8  server_address = ('localhost', 12345)
9  sock.sendto(message, server_address)
10
11 # Ждем ответа от сервера
12 data, _ = sock.recvfrom(4096)
13 print('Получено сообщение от сервера:', data.decode())
14
15 # Закрываем сокет
16 sock.close()
```

Сервер с консолью, где видно как работает

```
index.py U  client.py U  server.py U X
web > ITMO_ICT_WebDevelopment_2023-2024 > students > k33420 > Kirill_Glushkov > Ir1 > server.py > ...
1  import socket
2
3  # Создаем сокет
4  sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5
6  # Привязываем сокет к конкретному IP адресу и порту
7  server_address = ('localhost', 12345)
8  print('Старт сервера на {}:{}'.format(*server_address))
9  sock.bind(server_address)
10
11 while True:
12     print('\nОжидание сообщения...')
13     data, address = sock.recvfrom(4096)
14
15     # Печатаем полученное сообщение
16     print('Получено сообщение от {}: {}'.format(address, data.decode()))
17
18     # Отправляем ответное сообщение клиенту
19     message = b'Hello, client'
20     sock.sendto(message, address)
```

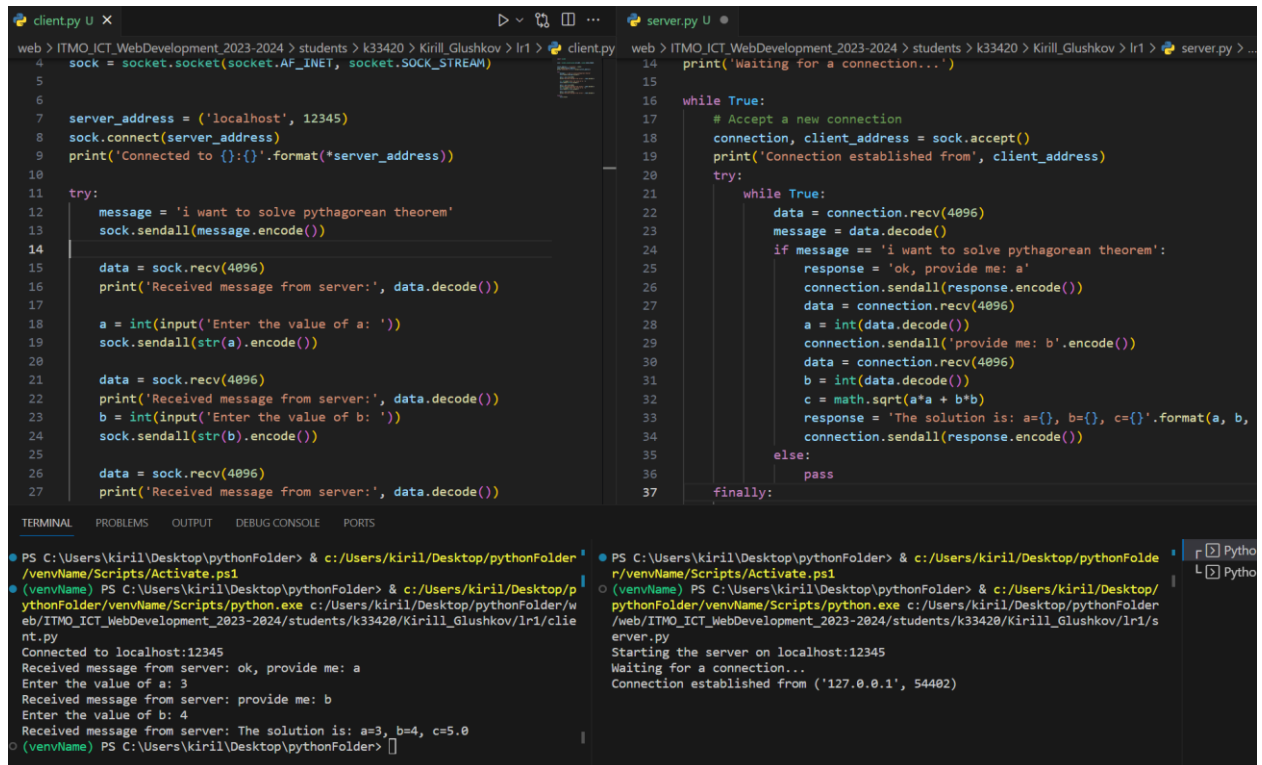
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS

```
PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/Activate.ps1
(venvName) PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/python.exe c:/Users/kiril/Desktop/pythonFolder/web/ITMO_ICT_WebDevelopment_2023-2024/students/k33420/Kirill_Glushkov/Ir1/client.py
Получено сообщение от сервера: Hello, client
(venvName) PS C:\Users\kiril\Desktop\pythonFolder>
```

Ожидание сообщения...
Получено сообщение от ('127.0.0.1', 49689): Hello, server
Ожидание сообщения...
[]

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. Реализовать с помощью протокола TCP
Варианты:

а. Теорема Пифагора



```
client.py
4 sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5
6
7 server_address = ('localhost', 12345)
8 sock.connect(server_address)
9 print('Connected to {}:{}'.format(*server_address))
10
11 try:
12     message = 'i want to solve pythagorean theorem'
13     sock.sendall(message.encode())
14
15     data = sock.recv(4096)
16     print('Received message from server:', data.decode())
17
18     a = int(input('Enter the value of a: '))
19     sock.sendall(str(a).encode())
20
21     data = sock.recv(4096)
22     print('Received message from server:', data.decode())
23     b = int(input('Enter the value of b: '))
24     sock.sendall(str(b).encode())
25
26     data = sock.recv(4096)
27     print('Received message from server:', data.decode())
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
server.py
14 print('Waiting for a connection...')
15
16 while True:
17     # Accept a new connection
18     connection, client_address = sock.accept()
19     print('Connection established from', client_address)
20     try:
21         while True:
22             data = connection.recv(4096)
23             message = data.decode()
24             if message == 'i want to solve pythagorean theorem':
25                 response = 'ok, provide me: a'
26                 connection.sendall(response.encode())
27                 data = connection.recv(4096)
28                 a = int(data.decode())
29                 connection.sendall('provide me: b'.encode())
30                 data = connection.recv(4096)
31                 b = int(data.decode())
32                 c = math.sqrt(a*a + b*b)
33                 response = 'The solution is: a={}, b={}, c={}'.format(a, b,
34                             connection.sendall(response.encode())
35             else:
36                 pass
37         finally:
38             connection.close()
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/Activate.ps1
(venvName) PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/python.exe c:/Users/kiril/Desktop/pythonFolder/web/ITMO_ICT_WebDevelopment_2023-2024/students/k33420/Kirill_Glushkov/lr1/client.py
Connected to localhost:12345
Received message from server: ok, provide me: a
Enter the value of a: 3
Received message from server: provide me: b
Enter the value of b: 4
Received message from server: The solution is: a=3, b=4, c=5.0
(venvName) PS C:\Users\kiril\Desktop\pythonFolder>
```

```
PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/Activate.ps1
(venvName) PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/python.exe c:/Users/kiril/Desktop/pythonFolder/web/ITMO_ICT_WebDevelopment_2023-2024/students/k33420/Kirill_Glushkov/lr1/server.py
Starting the server on localhost:12345
Waiting for a connection...
Connection established from ('127.0.0.1', 54402)
```

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 и работа

```
client.py
1 import socket
2
3 # Create a TCP/IP socket
4 client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5
6 server_address = ('localhost', 8080)
7 client_socket.connect(server_address)
8
9
10 try:
11     client_socket.sendall('GET / HTTP/1.1\r\nHost: {}\r\n\r\n'.format(server_address[0]))
12     response = client_socket.recv(4096).decode()
13     print('Received HTML code:\n')
14     print(response)
15
16 finally:
17     client_socket.close()

```

```
server.py
1 import socket
2
3 def serve_index_page():
4     # Open and read the index.html file
5     with open('index.html', 'r') as file:
6         html_content = file.read()
7
8     return html_content
9
10 # Create a TCP/IP socket
11 server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
12
13 # Bind the socket to a specific address and port
14 server_address = ('localhost', 8080)
15 server_socket.bind(server_address)
16
17 # Listen for incoming connections
18 server_socket.listen(1)
19
20 print('Server started. Listening on {}'.format(*server_address))
21
22 while True:
23     print('Waiting for a client connection...')
24     client_socket, client_address = server_socket.accept()
25     print('Accepted client connection from {}'.format(*client_address))
26
27     try:
28         response = serve_index_page()

```

```

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
</head>
<body>
<p>
    Today is a beautiful day. We go swimming and fishing.
</p>
<p>
    Hello there. How are you?
</p>
</body>
</html>

```

```

PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/Activate.ps1
(venvName) PS C:\Users\kiril\Desktop\pythonFolder> & c:/Users/kiril/Desktop/pythonFolder/venvName/Scripts/python.exe c:/Users/kiril/Desktop/pythonFolder/web/ITMO ICT_WebDevelopment_2023-2024/students/k33420/Kirill_Glushkov/lr1/task3/server.py
Server started. Listening on localhost:8080
Waiting for a client connection...
Accepted client connection from 127.0.0.1:52426
Response sent. Connection closed.
Waiting for a client connection...

```

4. Реализовать двухпользовательский или многопользовательский чат.

Реализация многопользовательского чата позволяет получить максимальное количество баллов. Реализовать с помощью протокола TCP – 100% баллов, с помощью UDP – 80%.

Обязательно использовать библиотеку `threading`.

Для реализации с помощью UDP, `threading` использовать для получения сообщений у клиента. Для применения с TCP необходимо запускать клиентские подключения и прием и отправку сообщений всем юзерам на сервере в потоках. Не забудьте сохранять юзеров, чтобы потом отправлять им сообщения.

```
client.py
1
2
3
4 SERVER_IP = '127.0.0.1'
5 SERVER_PORT = 12345
6
7 def receive_messages(client_socket):
8     while True:
9         try:
10             message = client_socket.recv(1024).decode()
11             print(message)
12         except:
13             break
14
15 def send_message(client_socket):
16     while True:
17         message = input()
18         client_socket.send(message.encode())
19
20     if message == 'exit':
21         client_socket.close()
22         break
23
24 client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
25 client_socket.connect((SERVER_IP, SERVER_PORT))
26 threading.Thread(target=receive_messages, args=(client_socket,)).start()
27 threading.Thread(target=send_message, args=(client_socket,)).start()
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

server.py
1
2
3
4
5 server_socket.listen(5)
6
7
8
9
10
11
12 connected_clients = []
13
14 def handle_client(client_socket):
15     while True:
16         try:
17             message = client_socket.recv(1024).decode()
18             print(message)
19             if not message:
20                 connected_clients.remove(client_socket)
21                 client_socket.close()
22                 break
23
24             for client in connected_clients:
25                 if client != client_socket:
26                     client.send(message.encode())
27
28         except:
29             continue
30
31 def accept_clients():
32     while True:
33         client_socket, client_address = server_socket.accept()
34         connected_clients.append(client_socket)
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

Terminal
PS C:\Users\kirill\Desktop\pythonFolder> & c:/Users/kirill/Desktop/pythonFolder/venvName/Scripts/Activate.ps1
(venvName) PS C:\Users\kirill\Desktop\pythonFolder> & c:/Users/kirill/Desktop/pythonFolder/venvName/Scripts/python.exe c:/Users/kirill/Desktop/pythonFolder/web/ITMO_ICT_WebDevelopment_2023-2024/students/k33420/Kirill_Glushkov/lr1/task4/server.py
hi
```

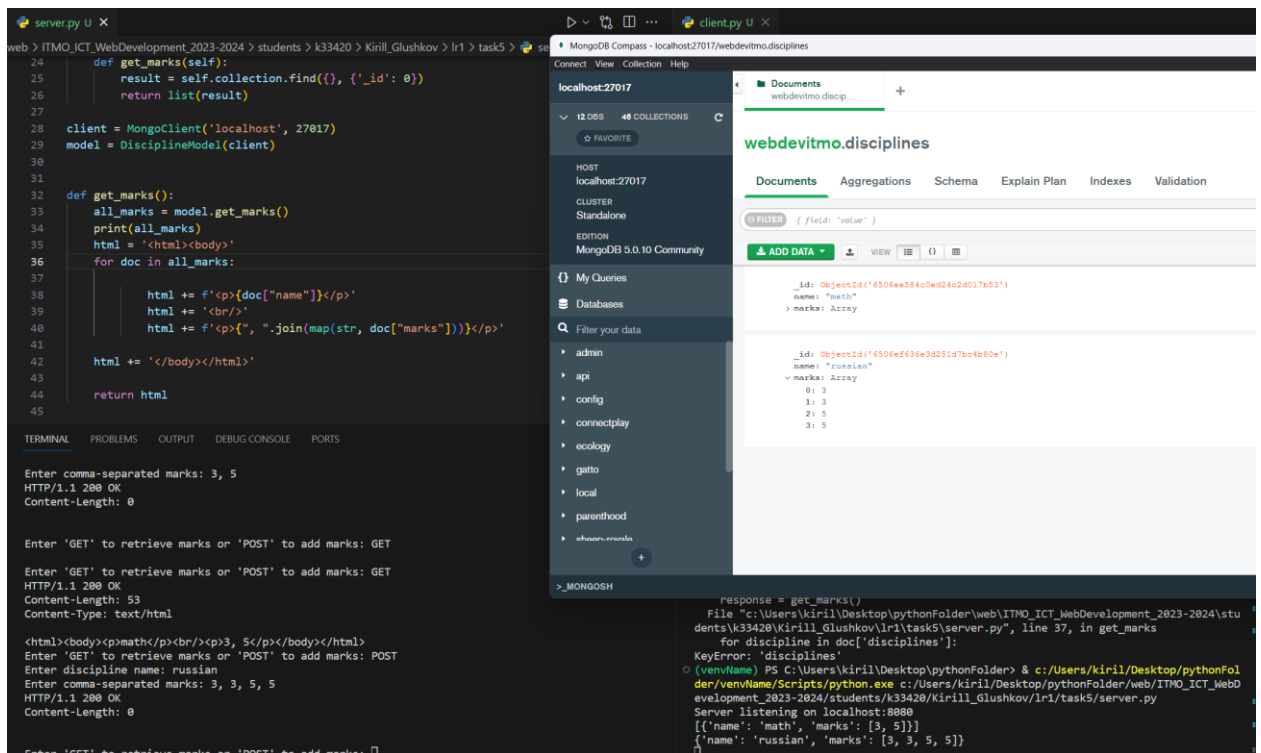
5. Необходимо написать простой web-сервер для обработки GET и POST http запросов средствами Python и библиотеки socket.

Базовый класс для простейшей реализации web-сервера доступен

Подробный мануал по работе доступен <https://iximiuz.com/ru/posts/writing-python->

Задание: сделать сервер, который может:

- Принять и записать информацию о дисциплине и оценке по дисциплине.



- Отдать информацию обо всех оценках по дисциплине в виде html-страницы.

