Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет Инфокоммуникационных Технологий

**Web-программирование**

**Лабораторная работа 6**

Выполнил

Бордюг В. Ю.

Проверила

Марченко Е. В.

Санкт-Петербург, 2025

**Цель:** освоить основы работы с сетевыми сокетами и многопоточностью в Python через разработку различных клиент-серверных приложений.

**Задачи:**

* Реализация простейшего клиент-серверного приложения
* Реализация клиент-серверного приложения для выполнения математической операции
* Реализация серверной части HTTP-приложения
* Реализация многопользовательского чата

**Задание 1**

Был создан сервер для получения сообщения и отправки ответа, код:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1 – server.py

Был создан клиент, отправляющий сообщение и получающий ответ, код:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2 – код client.py

Пример использования, отправка сообщения и получение клиентом:

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3 – результат работы клиента

Результат работы сервера:

Изображение выглядит как текст, Шрифт, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4 – результат работы сервера

**Задание 2**

Был написан сервер, который принимает основания трапеции и высоту и возвращает площадь трапеции:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Контент, сгенерированный ИИ, может содержать ошибки. Рисунок 5 – server.py

Был написан клиент, который отправляет 3 стороны трапеции и получает ответ:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6 – client.py

Результат работы клиента:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7 – результат работы клиента

Результат работы сервера:

Изображение выглядит как текст, Шрифт, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8 – результат работы сервера

**Задание 3**

Был написан сервер, который принимает соединение и отправляет html файл index.html:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9 – server.py

Был написан клиент, который получает html файл:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10 – client.py

Результат работы клиента:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11 – результат работы клиента

Результат работы сервера:

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11 – результат работы сервера

**Задание 4**

Был написан сервер, который принимает в разных потоках клиентов и регистрирует их по ID для чата:

**Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 12 – server.py

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 13 – server.py

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 14 – server.py

Был написан клиент, который подключается к сокету и в бесконечном цикле отправляет сообщение:

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 15 – client.py

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 16 – client.py

Были запущены 2 клиента, пример отправки первым сообщения второму:

**Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 17 – отправка сообщения

Получение вторым пользователем сообщения от первого:

**Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 18 – получение сообщения

Результат работы сервера:

Изображение выглядит как текст, программное обеспечение, Шрифт, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 19 – результат работы сервера

**Вывод:** были освоены основы работы с сетевыми сокетами и многопоточностью в Python через разработку различных клиент-серверных приложений.