# Лабораторная работа №1 Работа с сокетами

### Темы для предварительного изучения

Примитивные и ссылочные типы данных.

Операторы и управление ходом выполнения программы.

Классы-обертки.

Массивы и строки.

Интерфейсы и наследование.

Исключения, их создание и обработка.

Потоки инструкций, взаимодействие потоков инструкций.

Потоки данных и их разновидности.

Лексический анализ строк и потоков данных.

Сериализация.

Работа с сокетами на Java.

### Задание на лабораторную работу

В процессе разработки учебного приложения ознакомиться с особенностями создания клиент-серверных приложений на основе сокетов.

### Введение

Часто необходимость создания клиент-серверного приложения возникает в случае, если на клиентской машине решение некоторой задачи (например, вычислительной), не представляется возможным. В этом случае клиент лишь формирует задание для сервера, а потом получает результаты. В данной работе предлагается создать клиент-серверное приложение, реализующее решение на серверной стороне некоторой вычислительной задачи.

### Задание 1

Написать класс матриц, поддерживающий операцию сложения или перемножения (в зависимости от вашего варианта, см. таблицу 1).

Должен быть описан один класс, который должен:

- инкапсулировать в себе массив-матрицу;

- иметь конструктор, параметрами которого являются размеры матрицы;

- реализовывать получение размеров матрицы (два метода, возвращающие количество строк и количество столбцов соответственно);

- реализовывать получение и изменение отдельных элементов матрицы (два метода, позволяющие получить и изменить значение элемента матрицы по его координатам);

- иметь статический метод сложения или перемножения матриц, возвращающий результат действия;

- иметь статический метод, позволяющий записать матрицу в текстовый файл;

- иметь статический метод, позволяющий считать матрицу из текстового файла.

### Задание 2

Разработать клиент-серверную программу: клиент считывает исходные данные из файлов и передает их для решения серверу, после чего получает ответ от сервера и записывает его в файл.

При запуске клиентской части в командной строке передаются следующие параметры: имя файла с первой матрицей, имя файла со второй матрицей, имя файла для результирующей матрицы.

Передача параметров серверу может производиться различными способами: передача непосредственно исходных данных (при этом сервер сам создает объекты для решения задачи) в виде последовательности чисел или в виде сериализованного массива, либо передача уже сформированных объектов задачи (методами сериализации), для которых на серверной стороне лишь вызывается метод решения. В обоих случаях предполагается, что на серверной стороне наличествует байт-код необходимых классов.

Сервер также можно реализовать различными способами: с последовательной обработкой запросов (в один момент времени сервер выполняет только один запрос), либо с параллельной (каждый запрос выполняется в отдельном потоке инструкций).

В выходной файл клиентской программы должен быть записан результат выполнения действия. Если выполнение действия невозможно, в выходной файл должно быть записано текстовое сообщение о невозможности выполнения действия.

Требования к конкретной реализации в зависимости от варианта задания приведены в таблице 1.

Таблица 1. Требования к реализации.

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Вид сервера | Передача параметров | Действие над матрицами |
| 1 | Последовательный | Последовательность элементов | Сложение |
| 2 | Последовательный | Последовательность элементов | Умножение |
| 3 | Последовательный | Сериализованные массивы | Сложение |
| 4 | Последовательный | Сериализованные массивы | Умножение |
| 5 | Последовательный | Сериализованные объекты | Сложение |
| 6 | Последовательный | Сериализованные объекты | Умножение |
| 7 | Параллельный | Последовательность элементов | Сложение |
| 8 | Параллельный | Последовательность элементов | Умножение |
| 9 | Параллельный | Сериализованные массивы | Сложение |
| 10 | Параллельный | Сериализованные массивы | Умножение |
| 11 | Параллельный | Сериализованные объекты | Сложение |
| 12 | Параллельный | Сериализованные объекты | Умножение |