**Упражнение 2: MultiClient/UDP/MultCast/SSL Sockets**

**Задача 1:**

Напишете мулти клиент-сървър приложение, което използва **TCP сокети** за комуникация. Клиентът ще изпраща съобщение до сървъра, а сървърът ще отговаря с IP адреса на клиента и информация за мрежовия интерфейс на сървъра.

**Изисквания:**

1. **Сървърът**:
   * Сървърът трябва да приема връзки от клиента, като използва **ServerSocket**.
   * Когато клиент се свърже, сървърът трябва да изпрати IP адреса на клиента и информация за мрежовия интерфейс на сървъра.
2. **Клиентът**:
   * Клиентът трябва да изпрати кратко съобщение до сървъра.
   * Клиентът трябва да изчака и да получи отговор от сървъра, след което да го отпечата.

**Стъпки:**

1. Създайте сървър, който слуша за връзки от клиент и използва **InetAddress**, за да получи IP адреса на клиента.
2. Сървърът трябва да изпрати информация за мрежовия интерфейс към клиента, като използва **NetworkInterface**.
3. Клиентът трябва да изпрати съобщение към сървъра и да получи информацията за IP адреса и мрежовия интерфейс.

Примерен изход:

Server is started on port 9238...

IP address of client: 127.0.0.1

NetworkAddressOfServer:

Interface: utun3

Address: fe80:0:0:0:ce81:b1c:bd2c:69e%utun3

Interface: utun2

Address: fe80:0:0:0:7f9e:e87e:b2e0:6ad0%utun2

Interface: utun1

Address: fe80:0:0:0:807f:3647:86af:d1a5%utun1

Interface: utun0

Address: fe80:0:0:0:9b15:9fdb:1112:3ae0%utun0

Interface: llw0

Address: fe80:0:0:0:c44d:c9ff:fe15:9742%llw0

Interface: awdl0

Address: fe80:0:0:0:c44d:c9ff:fe15:9742%awdl0

Interface: ap1

Address: fe80:0:0:0:bcd0:74ff:fe73:3817%ap1

Interface: utun4

Address: fc00:35d5:1ba1:45fb:5b60:8195:67c3:e6b8%utun4

Address: fe80:0:0:0:bed0:74ff:fe73:3817%utun4

Address: 10.147.43.24

Interface: bridge100

Address: fd32:8233:d26c:df62:4e9:469d:1e64:18ff%bridge100

Address: fe80:0:0:0:bcd0:74ff:fe37:9464%bridge100

Address: 192.168.205.1

……

**Задача 2. Grades(Дневник)**

Задачата от упражнение 1 , част 2 да се направи със сокети:

Клиентите ще изпраща заявки към сървъра за добавяне на студент, преглед на всички оценки или изчисляване на средната оценка. Сървърът ще обработва тези заявки, ще чете/записва във файл и ще връща резултата на клиента.

Сървър**:**

Сървърът приема заявки от клиентите чрез TCP сокети. Управлява файл, в който се съхраняват данните за студентите и техните оценки.

Обработва заявки като добавяне на студент, преглед на списъка с оценки и изчисляване на средна оценка.

Клиент:

Клиентът изпраща заявки към сървъра (добавяне на студент, преглед на оценки или изчисляване на средна оценка) и получава резултати обратно от сървъра.

Изходен код:

--- Меню ---

1. Добавяне на студент

2. Преглед на всички студенти

3. Търсене на студент по име

4. Средна оценка на всички студенти

5. Изход

Изберете опция (1-5): 1

Въведете името на студента: test

Въведете оценката на студента: 5

Студентът е добавен успешно.

StudentProcessor addStudent

INFO: Успешно добавихте студента: test с оценка: 5

**Задача 3. UDP сървър**

Вие сте студент от първи курс на техническия университет, и искате да създадете прост чат чрез използване на **UDP сокети**. Задачата ви е да създадете приложение, което да позволява на два или повече потребителя да комуникират помежду си през мрежата. За разлика от TCP, UDP не гарантира доставянето на съобщенията, което прави задачата още по-интересна, тъй като ще научите как да изпращате и получавате съобщения в една "ненадеждна" среда.

**Задачи:**

1. Създайте **UDP сървър**, който да може да приема съобщения от клиенти и да изпраща обратно потвърждения.
2. Създайте **UDP клиент**, който да може да изпраща съобщения до сървъра и да получава отговори.
3. Разширете приложението така, че всеки клиент да може да изпраща съобщения и към други клиенти чрез сървъра.

**Как работи програмата:**

1. **UDP Сървър:**
   * Сървърът слуша на порт 9876 за входящи UDP пакети.
   * Всеки път, когато получи съобщение от клиент, сървърът го принтира на конзолата и връща потвърждение на същия клиент.
2. **UDP Клиент:**
   * Клиентът изпраща съобщение до сървъра на локалния адрес (localhost) на порт 9876.
   * След като съобщението бъде изпратено, клиентът чака отговор от сървъра и го извежда на екрана.

**Упътване:**

Използвайте DatagramSocket и DatagramPacket

serverSocket.receive(packet)

Изходен код:

UDP чат сървър стартиран на порт 9876

Получено съобщение: hello от /127.0.0.1:60331

Получено съобщение: test от /127.0.0.1:60331

Получено съобщение: test от /127.0.0.1:60879

Получено съобщение: reas от /127.0.0.1:60879

Получено съобщение: quit от /127.0.0.1:60879

Enter message: hello

Response: Сървър: получено съобщение 'hello'

Enter message: test

Response: Сървър: получено съобщение 'test'

Enter message: quit

Response: Сървър: получено съобщение 'quit'

Process finished with exit code 0

**Задача 4 :**

**Проста игра "Камък, ножица, хартия" с UDP**

**Описание**: Направете игра "Камък, ножица, хартия" между два клиента, които комуникират чрез UDP.

**Изисквания**:

* Два клиента изпращат своите ходове към сървър.
* Сървърът определя победителя и изпраща резултата обратно към клиентите.
* Клиентите могат да играят няколко рунда.

// Create a Datagram Socket  
 DatagramSocket clientSocket = new DatagramSocket(порт);

<https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/net/DatagramSocket.html>

<https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/net/DatagramPacket.html>

**Задача 5. Прогноза за времето (Multicast)**

**Описание**: Създайте сървър, който предоставя информация за прогноза за времето за различни градове.

**Изисквания**:

* Клиентът изпраща заявка за прогноза, като посочва име на град.
* Сървърът връща текущата прогноза за посочения град (може да бъде фиктивна информация, ако не се използва реална API услуга).
* Клиентът отпечатва прогнозата на екрана.

Още задачи:

1. **Преброяване на думи (TCP)**

**Описание**: Направете клиент-сървър приложение, при което клиентът изпраща текст към сървъра, а сървърът връща броя на думите в текста.

**Изисквания**:

* Клиентът изпраща произволен текст към сървъра.
* Сървърът преброява думите в текста и връща броя на думите на клиента.
* Клиентът отпечатва броя на думите.

1. **Пренос на криптирани съобщения (TCP)**

**Описание**: Направете приложение, което шифрира съобщенията преди изпращането и ги дешифрира при получаване.

**Изисквания**:

* Клиентът въвежда съобщение, което се шифрира (например с проста Цезарова шифровка) и изпраща към сървъра.
* Сървърът дешифрира съобщението и го отпечатва на екрана.
* Добавете опция сървърът също да връща криптирани отговори.