

Санкт-Петербургский политехнический университет Петра Великого  
Институт компьютерных наук и технологий  
Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе**  
«Организация сетевого взаимодействия. Протоколы TCP и UDP»  
Дисциплина: Сети и телекоммуникации

Работу выполнила студентка гр. 43501/3:  
Работу принял преподаватель:

Ивашкевич О.А.  
Алексюк А.О.

## Содержание

1Цель работы.....	3
2Индивидуальное задание.....	3
3Разработанный прикладной протокол.....	4
3.1. Описание структуры приложение на основе ТСР.....	6
4Результаты тестирования.....	7
4.1. Тестирование со стороны серверного приложения.....	7
4.2. Тестирование со стороны клиентского приложения.....	9
5Выводы.....	10

## 1 Цель работы

Изучение принципов программирования сокетов с использованием протоколов TCP и UDP.

## 2 Индивидуальное задание

Разработать приложение–клиент и приложение–сервер электронного магазина. Товар в электронном магазине имеет уникальный идентификатор, наименование, цену.

Необходимые операции для сервера:

- 1) Прослушивание определенного порта
- 2) Обработка запросов на подключение по этому порту от клиентов электронного магазина
- 3) Поддержка одновременной работы нескольких клиентов электронного магазина через механизм нитей
- 4) Прием запросов на добавление или покупку товара
- 5) Осуществление добавления товара, учет количества единиц товара
- 6) Передача клиенту электронного магазина информации о товарах и подтверждений о совершении покупки
- 7) Обработка запроса на отключение клиента
- 8) Принудительное отключение клиента

Необходимые функции для клиента:

- 1) Установление соединения с сервером
- 2) Передача запросов о добавлении, покупке товаров серверу
- 3) Получение ответов на запросы от сервера
- 4) Разрыв соединения
- 5) Обработка ситуации отключения клиента сервером

**Настройки приложений.** Разработанное клиентское приложение должно предоставлять пользователю настройку IP–адреса или доменного имени сервера электронного магазина и номера порта, используемого сервером.

**Методика тестирования.** Для тестирования приложений запускается сервер электронного магазина и несколько клиентов. В процессе тестирования проверяются основные возможности приложений по передаче и приему сообщений.

**Вариант реализации:** сервер – Windows, клиент – Linux

### 3 Разработанный прикладной протокол

Первоначально сервер прослушивает порт и находится в режиме ожидания новых клиентов. Когда клиент хочет подключиться к серверу, он устанавливает соединение. Когда соединение установлено, сервер отправляет приветствие. Клиент и сервер, затем обмениваются командами и ответами (соответственно), пока соединение не будет закрыто или прервано.

Передаваемые данные представляют собой строку символов. Сервер обрабатывает принятую строку и в зависимости от первого принятого байта или названия отправленной команды и отправляет соответствующий запросу ответ. При этом сервер хранит актуальную информацию о результатах клиента. Для клиента принятая строка является результатом запроса, который надо вывести на экран. Также возможно получение команды на отключение от сервера.

Пока не отослали команду и не получили ответ, нельзя посылать новые команды. В соответствии с заданием был введен протокол обмена. Возможные запросы представлены в таблице 1.

*Таблица 1. Перечень запросов*

Запрос	Пример	Действие	Возможные ответы
add "Name product: " NAME "Amount product: " VALUE	"Message TO Server:" add "Name product: " koka-kola "Amount product: " 5	Добавление нового товара  Ограничения: NAME – 50 символов	<ul style="list-style-type: none"><li>▪ "Successfully. New product added."</li><li>▪ "Successfully. You update."</li></ul>
show	"Message TO Server:" show	Получить всю информацию о продуктах	<ul style="list-style-type: none"><li>▪ "ID_0 Name: koka-kola Amount:5 All list",</li><li>▪ "No product"</li></ul>
buy "Input ID product: " ID "Input amount product: " VALUE	"Message TO Server:" buy "Input ID product: " 0 "Input amount product: " 3	Покупка существующего товара	<ul style="list-style-type: none"><li>▪ "Nothing products!"</li><li>▪ "This's number of id not exist!"</li><li>▪ "Failed buy. Max amount 5. Try again."</li><li>▪ "You buy: koka-kola Amount: 3"</li></ul>
exit	"Message TO Server:" exit	Выход клиента	<ul style="list-style-type: none"><li>▪ "No User online"</li><li>▪ "1 user online"</li></ul>
Любая некорректная операция	"Message TO Server:" hi	Любая некорректная операция	"Not this operation"

Дополнительно к обработке каждого клиента в своем потоке были реализованы функции предоставления информации о всех подключенных клиентах, ручного отключения определенного/ всех подключенных клиентов.

*Таблица 2. Перечень запросов*

Запрос	Действие	Пример
show	Просмотр информации о всех подключенных клиентах	>show 0__192.168.0.103:57456 1__192.162.0.103:57458
kill	Отключение определенного клиента	>kill Client ID: 0  Возможные ответы: ▪ “Kill Client with id= 0” ▪ "This's number of id not exist!"
killall	Отключение всех клиентов	>killall Возможные ответы: ▪ “Kill Client with id= 0 Kill Client with id= 1 Kill Client with id= 2”
quit	Завершение работы сервера	>quit

По команде kill ID\_USER главный поток (при наличии клиента) останавливает обрабатывающий данного клиента поток и закрывает сокет. При этом в рабочий поток, связанным с клиентом, возвращается флаг “сокет неверен”, по которому происходит удаление клиента. Количество активных клиентов уменьшается на 1.

Команда quit действует также, как и команда kill, только останавливает и удаляет все клиентские потоки. После их завершения, она останавливает работу слушающего потока и закрывает его. Поток main останавливается.

### 3.1. Описание структуры приложение на основе ТСП

Сервер:

- > Инициализация всех переменных
- > Создание сокета
- > Создание потока для прослушивание сокета
- > Чтение команд от сервера и реакция
- > Добавление данных о подключенных клиентах
- > Чтение команд от клиента и реакция
- > Получения сообщений от клиентов
- > Анализ сообщение
- > Ответ на сообщение

Клиент:

- > Чтение IP адреса сервера
- > Подключение к серверу
- > Создание потока
- > Чтение данных от сервера и реакция
- > Отправка команд серверу
- > Поток для получения данных от сервера

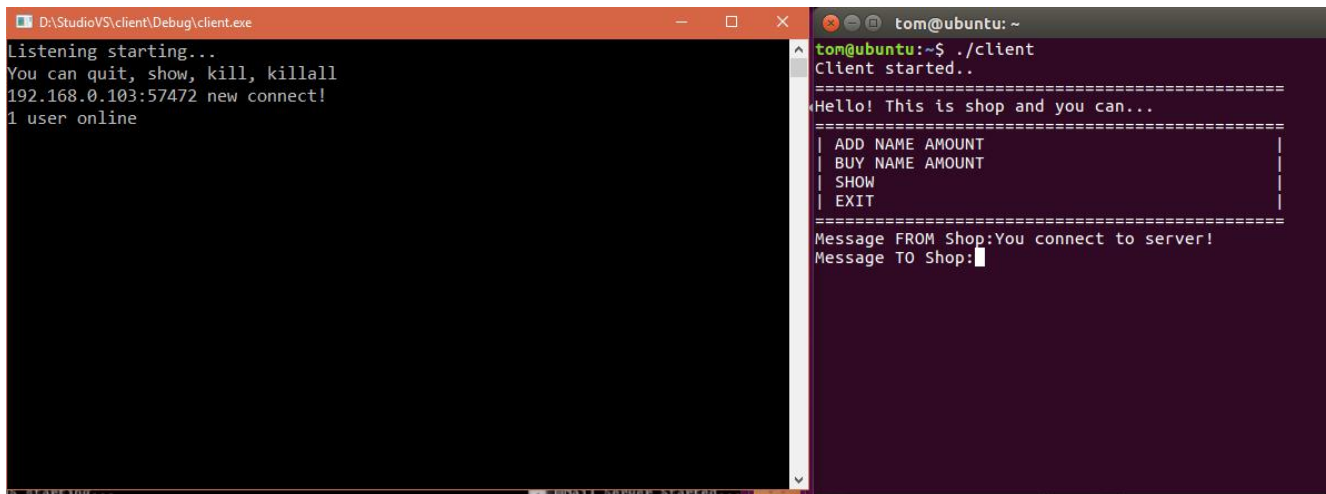
Сеанс проходит через ряд состояний во время продолжительности жизни. Как только соединение будет открыто и сервер отправил приветствие, сеанс переходит к идентификации, клиенту присваивается идентификационный номер. Как только идентификация проходит успешно, сессия переходит в состояние обмена данными. В этом состоянии клиент запрашивает действия со стороны сервера.

Когда клиент заканчивает сеанс, выдается команда “quit”, сеанс переходит в состояние update. В этом состоянии, сервер освобождает любые ресурсы, связанные с клиентом и прощается. Затем соединение закрыто.

## 4 Результаты тестирования

### 4.1. Тестирование со стороны серверного приложения

Запускаем приложение.

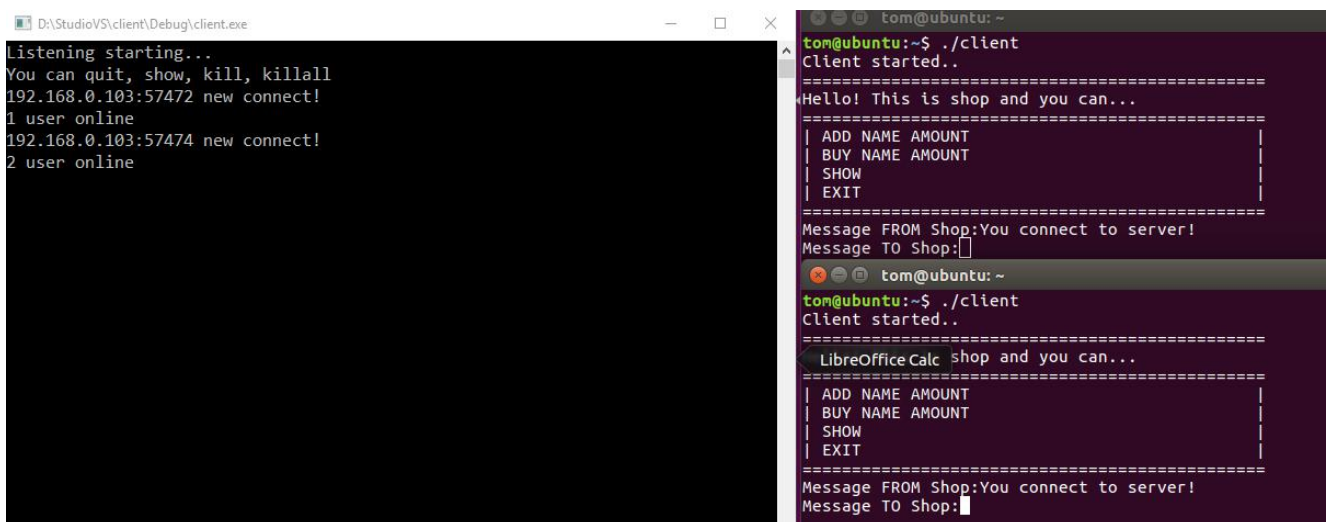


```
D:\StudioVS\client\Debug\client.exe
Listening starting...
You can quit, show, kill, killall
192.168.0.103:57472 new connect!
1 user online

tom@ubuntu: ~
tom@ubuntu:~$ ./client
Client started..
=====
Hello! This is shop and you can...
=====
| ADD NAME AMOUNT
| BUY NAME AMOUNT
| SHOW
| EXIT
=====
Message FROM Shop:You connect to server!
Message TO Shop:
```

Рис. 1. Клиент-серверное приложение

+1 клиент



```
D:\StudioVS\client\Debug\client.exe
Listening starting...
You can quit, show, kill, killall
192.168.0.103:57472 new connect!
1 user online
192.168.0.103:57474 new connect!
2 user online

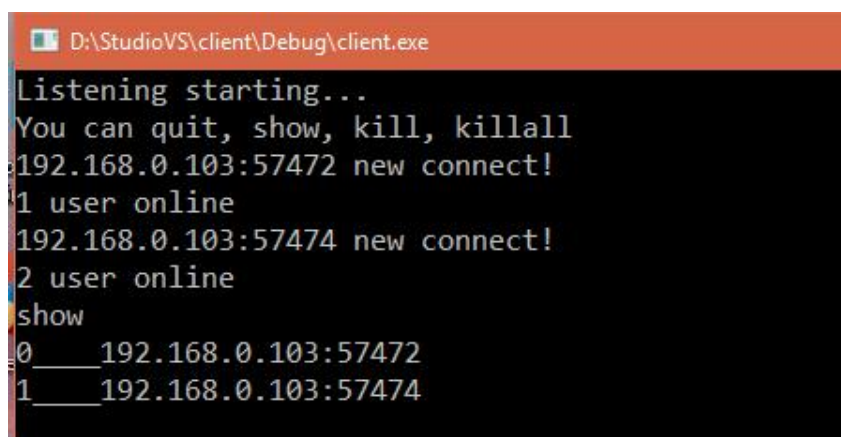
tom@ubuntu: ~
tom@ubuntu:~$ ./client
Client started..
=====
Hello! This is shop and you can...
=====
| ADD NAME AMOUNT
| BUY NAME AMOUNT
| SHOW
| EXIT
=====
Message FROM Shop:You connect to server!
Message TO Shop:

tom@ubuntu: ~
tom@ubuntu:~$ ./client
Client started..
=====
LibreOfficeCalc shop and you can...
=====
| ADD NAME AMOUNT
| BUY NAME AMOUNT
| SHOW
| EXIT
=====
Message FROM Shop:You connect to server!
Message TO Shop:
```

Рис. 2. Добавление нового клиента

Проверим команды show, kill, killall, quit.

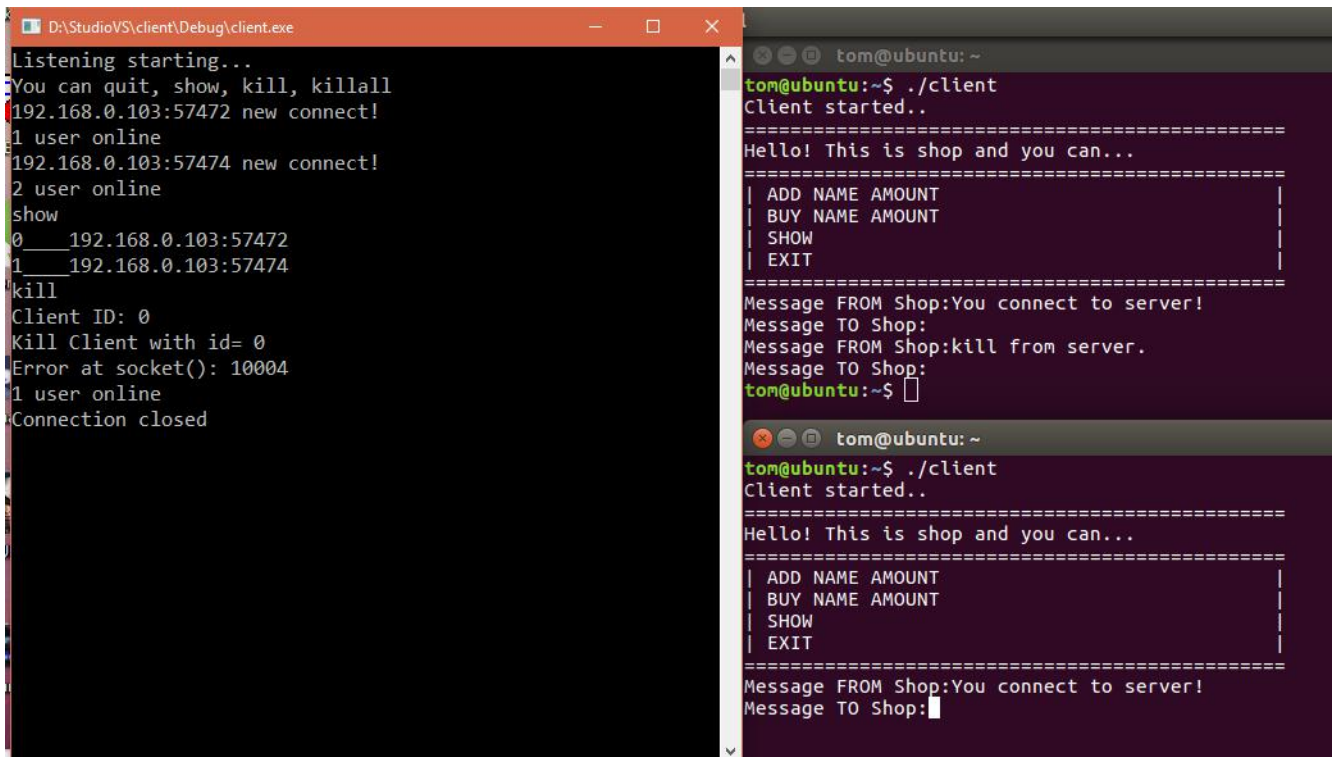
Show:



```
D:\StudioVS\client\Debug\client.exe
Listening starting...
You can quit, show, kill, killall
192.168.0.103:57472 new connect!
1 user online
192.168.0.103:57474 new connect!
2 user online
show
0___192.168.0.103:57472
1___192.168.0.103:57474
```

Рис. 3. Обзор всех клиентов

kill:



```
D:\StudioVS\client\Debug\client.exe
Listening starting...
You can quit, show, kill, killall
192.168.0.103:57472 new connect!
1 user online
192.168.0.103:57474 new connect!
2 user online
show
0 192.168.0.103:57472
1 192.168.0.103:57474
kill
Client ID: 0
Kill Client with id= 0
Error at socket(): 10004
1 user online
Connection closed

tom@ubuntu: ~
tom@ubuntu:~$ ./client
Client started..
=====
Hello! This is shop and you can...
=====
| ADD NAME AMOUNT
| BUY NAME AMOUNT
| SHOW
| EXIT
=====
Message FROM Shop:You connect to server!
Message TO Shop:
Message FROM Shop:kill from server.
Message TO Shop:
tom@ubuntu:~$
```

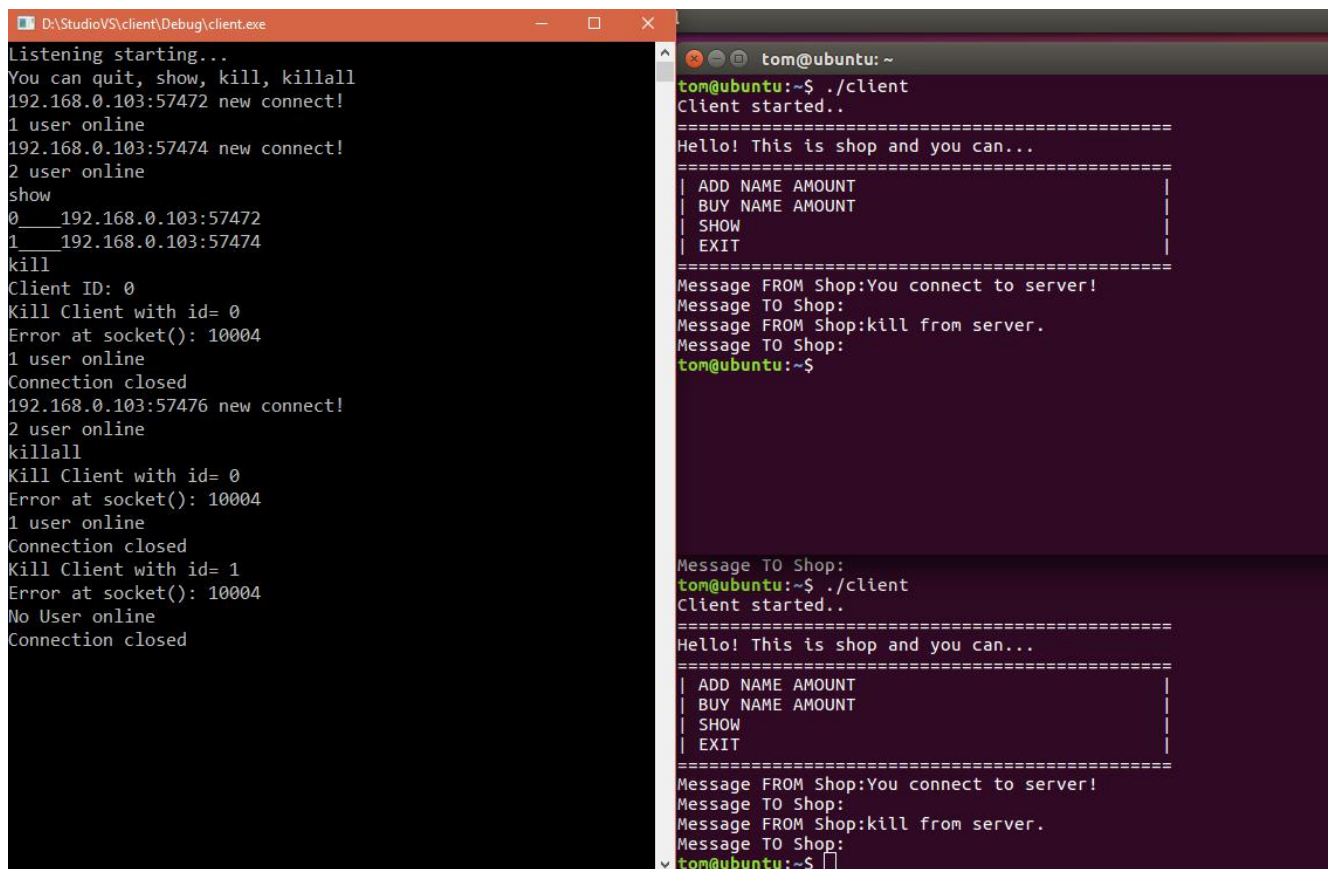
Рис. 4. Отключили клиента с ID 0.

В окне клиента появилось сообщение «kill from server»

В окне сервера видно удачное закрытия сокета и уменьшение кол-ва клиентов на 1.

Вновь добавим клиента и воспользуемся командой killall.

Killall:



```
D:\StudioVS\client\Debug\client.exe
Listening starting...
You can quit, show, kill, killall
192.168.0.103:57472 new connect!
1 user online
192.168.0.103:57474 new connect!
2 user online
show
0 192.168.0.103:57472
1 192.168.0.103:57474
kill
Client ID: 0
Kill Client with id= 0
Error at socket(): 10004
1 user online
Connection closed
192.168.0.103:57476 new connect!
2 user online
killall
Kill Client with id= 0
Error at socket(): 10004
1 user online
Connection closed
Kill Client with id= 1
Error at socket(): 10004
No User online
Connection closed

tom@ubuntu: ~
tom@ubuntu:~$ ./client
Client started..
=====
Hello! This is shop and you can...
=====
| ADD NAME AMOUNT
| BUY NAME AMOUNT
| SHOW
| EXIT
=====
Message FROM Shop:You connect to server!
Message TO Shop:
Message FROM Shop:kill from server.
Message TO Shop:
tom@ubuntu:~$
```

Рис. 5. Отключили всех клиентов.



## 4.2. Тестирование со стороны клиентского приложения

Добавим товар командой add:

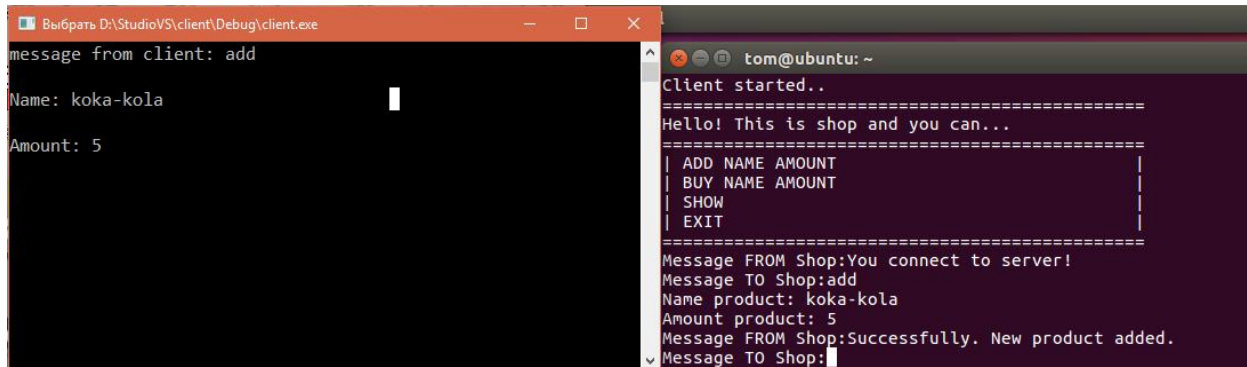


Рис. 6. Добавление 5 единиц товара koka-kola.

Добавим еще один товар и выведем список товаров:

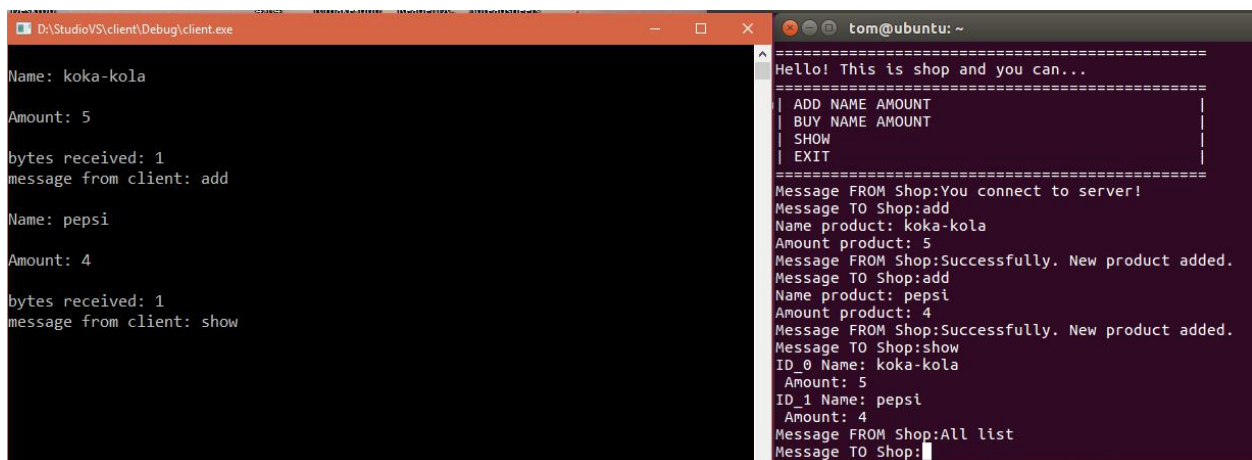


Рис. 7. Список товаров.

Покупка товара:

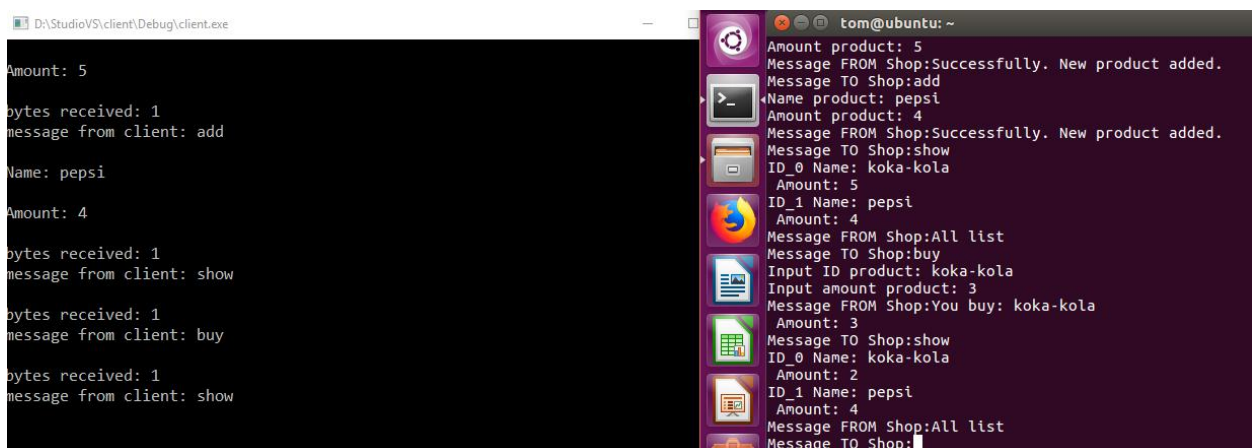


Рис. 8. Покупка товара.

Купили 3 единицы koka-kola, вывели список товаров, чтобы убедиться, что покупка реально совершилась видим, что осталось 2 единицы.

## 5 Выводы

В результате работы был реализован в виде приложений прикладной протокол. Сокеты оказались мощным средством организации обмена между процессами. TCP нужно установление соединения, поэтому на сервере выделяется поток, в котором происходит прием запросов на соединение от клиентов через выделенный на это сокет. После подключения порождается отдельный поток, осуществляющий обмен пакетами с этим клиентом через отдельный сокет, также он принимает заголово сообщения и вызывает обработчик.

При использовании на транспортном уровне потокового протокола (TCP) для обработки нескольких клиентов удобно использовать на сервере механизм нитей. Каждая нить работает со своим сокетом: слушающая нить принимает соединения со слушающего сокета, при подключении клиента создаётся новая нить, которой передаётся порождённый сокет.

На клиенте и на сервере возможна установка IP адреса и порта.

Основные параметры архитектуры контролируются через заголовочный файл;

Возможность удаления клиентов по автоматическому таймауту позволяет избавиться от «повисших» клиентов.