Студент Крамаренко М.К., группа 234, ИДЗ №4, вариант №30

### Задание:

Военная операция. Темной–темной ночью прапорщики Иванов, Петров и Нечепорук занимаются хищением военного имущества со склада родной военной части. Будучи умными людьми и отличниками боевой и строевой подготовки, прапорщики ввели разделение труда. Иванов выносит имущество со склада и передает его в руки Петрову, который грузит его в грузовик. Нечепорук стоит на шухере и заодно подсчитывает рыночную стоимость добычи поле погрузки в грузовик очередной партии похищенного.

Требуется разработать клиент–серверное приложение, моделирующее деятельность прапорщиков.

Каждый прапорщик — отдельный клиент. Сервер используется для передачи информации о продвижении имущества между прапорщиками.

Необходимо учесть случайное время выполнения каждым прапорщиком своей боевой задачи и организовать в программе корректную их синхронизацию.

**Обращаю внимание**, что в каждой программе есть файл launch.sh, который отвечает за компиляцию программ, а также выводит инструкцию по запуску программы.

# **Реализация на 4-5 балла. Программа находится в папке 04-05.**

**Разработать клиент–серверное приложение, в котором сервер (или серверы) и клиенты независимо друг от друга отображают только ту информацию, которая поступает им во время обмена. То есть, отсутствует какой-либо общий вывод интегрированной информации, отображающий поведение системыв целом.**

Требования:

1. **Представить сценарий решаемой задачи поясняющий, каким образом исходные сущности и их поведение отображаются в серверы, клиенты, процессы и как осуществляется их взаимодействие.**

Общий сценарий решения:

1. **Сервер.** Инициализирует udp сокет и ожидает подключение 3 клиентов: Иванова, Петрова и Нечепорука.
2. **Клиент.** Подключается по переданному адресу и порту. После подключения сразу сообщает свой id. Начинает работу при поступлении сообщению по сокету.
3. **Управление и синхронизация.** После подключения всех клиентов на сервере, сервер поочерёдно выполняет действия:
   1. посылает сообщение Иванову, что пора начинать работу и ждёт результат его работы.
   2. посылает сообщение Петрову и ждёт его ответ.
   3. посылает сообщение Нечепоруку и ждёт его ответ.
4. Завершение. Программа завершается, когда Иванов всё украл, и его программа посылает сообщение серверу со специальным id **IVANOV\_DONE\_ID**. Это является сигналом серверу о завершении работы. После чего все программы завершаются.
5. **При запуске программ требуемые для их работы IP адреса и порты необходимо задавать в командной строке, чтобы обеспечить гибкую подстройку к любой сети.**

Для всех программ указываются адрес и порт подключения. Для программы Иванова дополнительным опциональным параметром является количество вещей для кражи, по умолчанию **10**.

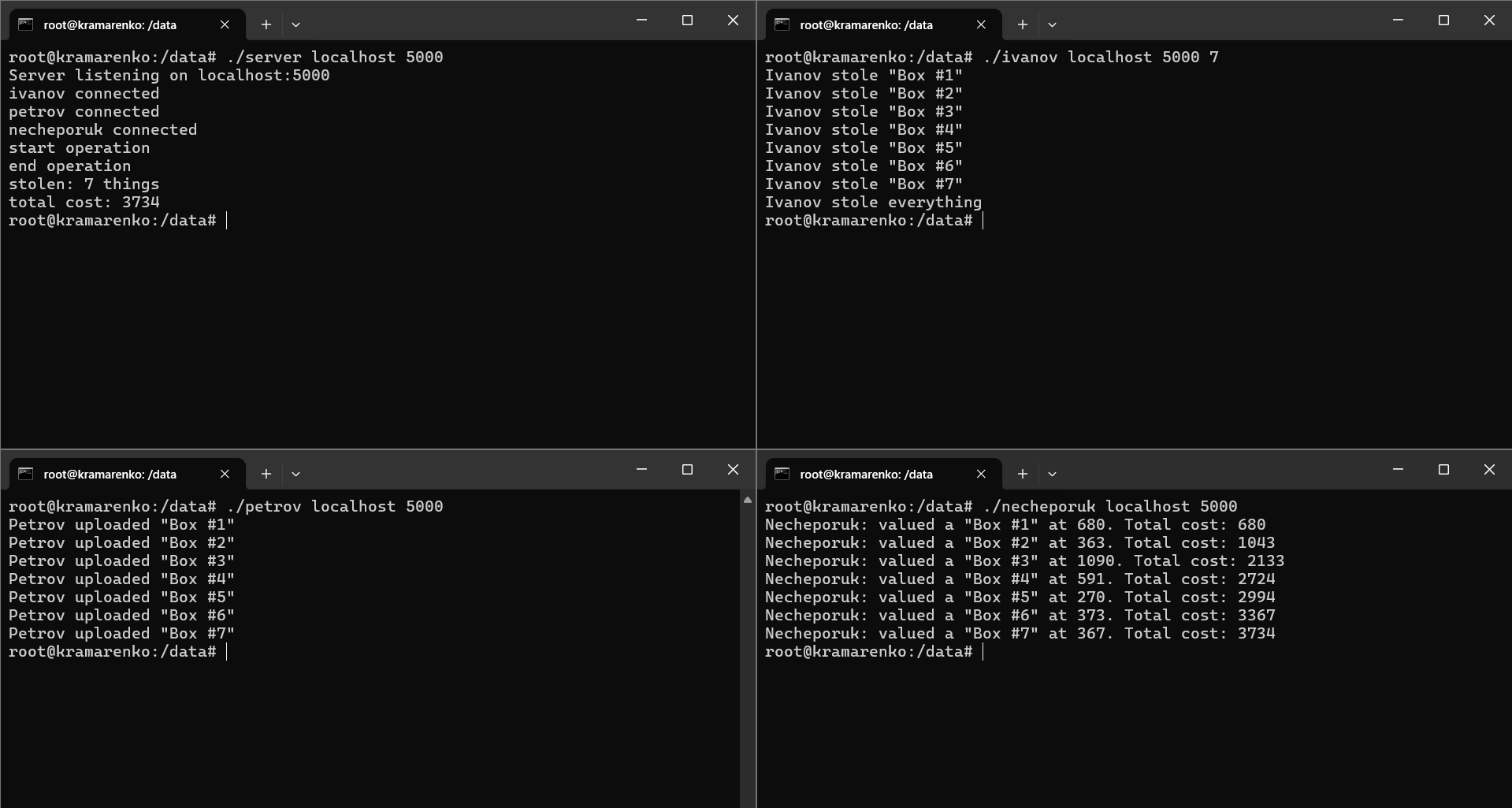
1. **Завершение работы клиентов и серверов на данном этапе не оговаривается. Но оно должно быть представлено в сценарии.**

Завершение. Программа завершается в двух случаях:

Первый. Иванов всё украл, и его программа посылает сообщение серверу со специальным id **IVANOV\_DONE\_ID**. Это является сигналом серверу о завершении работы. После чего все программы завершаются.

Второй. Получение сигнала **SIGINT** на любой программе. При получении такого сигнала, программы завершаются не дожидаясь, пока Иванов всё украдёт.

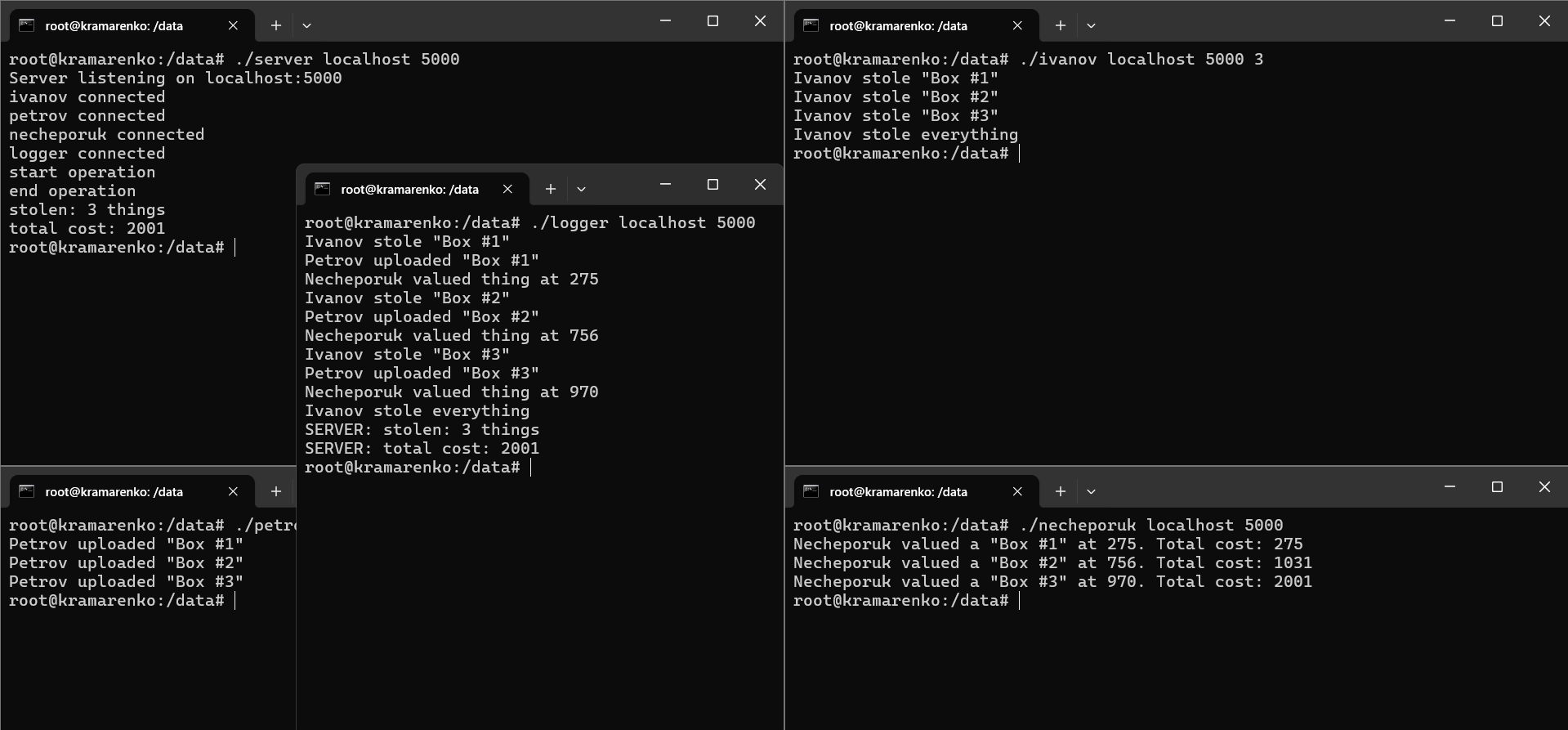
1. **Пример работы программы.**



# **Реализация на 6-7 баллов. Программа находится в папке 06-07.**

**В дополнение к программе на предыдущую оценку необходимо разработать отдельную клиентскую программу, подключаемую к серверу, которая предназначена для отображение комплексной информации о выполнении приложения в целом. То есть, данный программный модуль должен адекватно и в полном виде отображать поведение моделируемой системы (информацию, передаваемую на сервер и информацию, порождаемую сервером), позволяя не пользоваться отдельными видами, предоставляемыми клиентами и серверами по отдельности.**

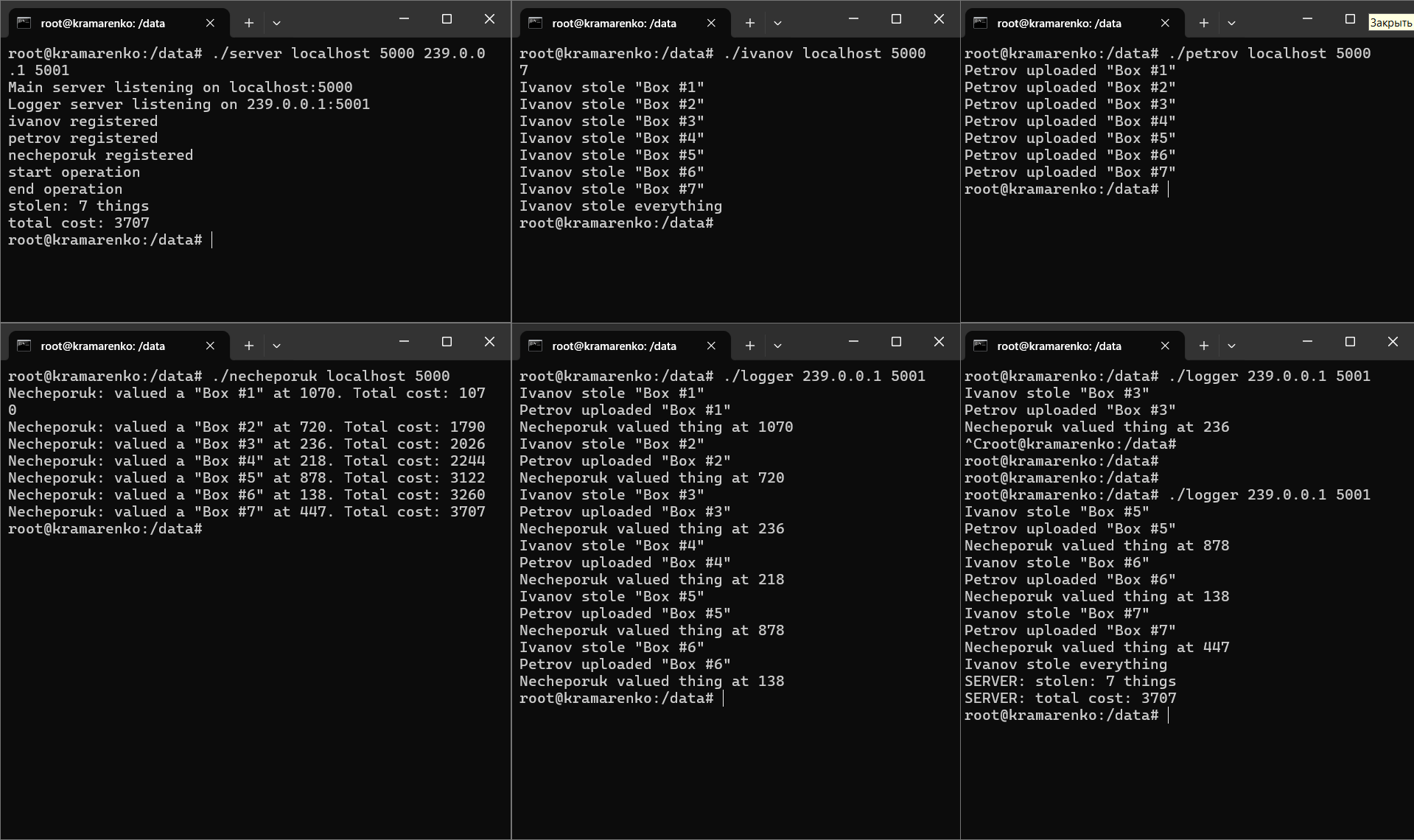
Была написана отдельная программа **logger.c**, которая подключается к серверу также, как и остальные программы. Сервер дублирует сообщения в **logger**, который выводит комплексную информацию о происходящем.



# **Реализация на 8 баллов. Программа находится в папке 08.**

**В дополнение к предыдущей программе реализовать возможность, подключения множества клиентов, обеспечивающих отображение информации о работе приложения. Это должно позволить осуществлять наблюдение за поведением программы с многих независимых компьютеров. При этом клиентов–наблюдателей можно отключать и подключать снова в динамическом режиме без нарушения работы всего приложения.**

Пример работы приложения:



**Ctrl + C**

**6**

**логгер**

**5**

**логгер**

**4**

**Нечепорук**

**3**

**Петров**

**2**

**Иванов**

**1**

**сервер**

**Комментарий к логгерам.**

Логгер в окне №5 подключается в самом начале и наблюдает до конца 6 итерации.

Логгер в окне №6 подключается позже начала работы программы и начинает выводить информацию с 3 итерации. Логгер наблюдает конца 3 итерации, после чего отключается. Затем заново подключается на 5 итерации и наблюдает до конца программы.

**Способ реализации.**

Создание дополнительного UDP (multicast) подключения для отображения информации.

# **Реализация на 9 баллов.**

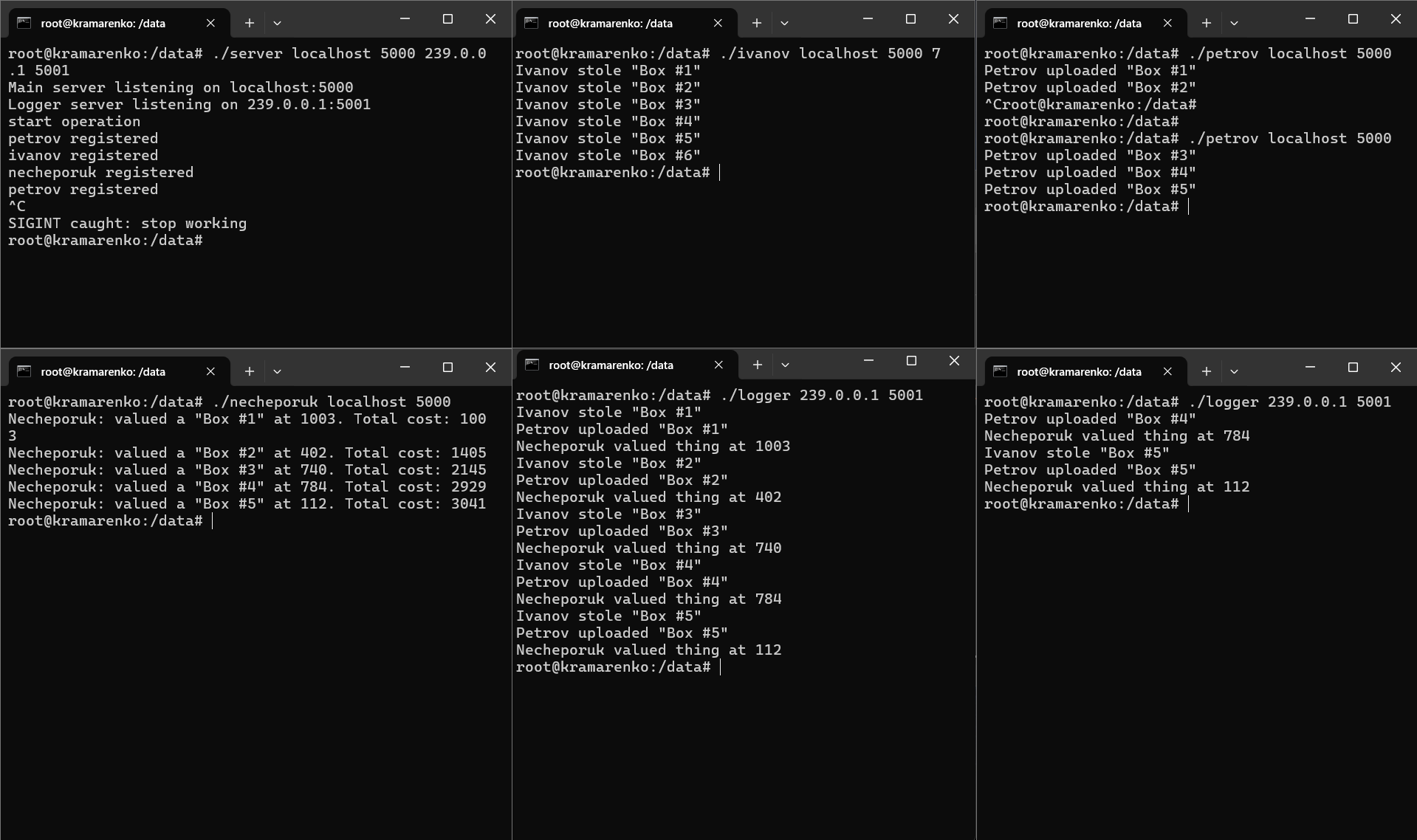
**В дополнение к программам на предыдущие оценки необходимо разработать приложение, позволяющее отключать и подключать различных клиентов с сохранением работоспособности сервера.**

# **Реализация на 10 баллов.**

**Расширить предыдущую программу таким образом, чтобы при завершении работы сервера происходило корректное завершение работы всех подключенных клиентов. То есть, данная программа должна являться модификацией программы на оценку в 9 баллов. Отдельную программу на 9 баллов в этом случае сдавать не нужно.**

**Программа находится в папке 09-10.**

Пример работы приложения:



**6**

**логгер**

**5**

**логгер**

**4**

**Нечепорук**

**3**

**Петров**

**2**

**Иванов**

**1**

**сервер**

Тут было прерывание **Ctrl+C**

После Петров подключился и

продолжил работу

Тут Петров подключился заново

Комментарий:

**Прерывание** клиента произошло на 3 итерации (окно №3). Как только клиент вернулся, сервер отправил ему нужный запрос на обработку. Программа продолжила функционировать.

**Клиенты-логгеры** работают независимо, как в предыдущей программе.

**Прерывание сервера** произошло на сервере на 6 итерации (окно №1), в результате чего программа завершилась, клиенты также завершили свою работу.

Идея решения.

Вместо ожидания ответа по конкретному адресу (**sockaddr\_in**, от конкретного клиента), читаем все запросы, которые поступают на сервер. Все запросы имеют метаданные от кого, и тип запроса. При получении ожидаемого ответа, программа продолжает работу; при получении запроса на регистрацию от ожидаемого клиента, запрос на обработку посылается повторно новому клиенту и ожидается ответ; другие ответы в этот момент игнорируются. Данная схема не работает с логгерами, т.к. у них отдельное multicast подключение.