

**Отчет**  
**Итоговое домашнее задание №4**  
**Выполнила: Захарова Наталья Владимировна**  
**БПИ223**

**Сетевые взаимодействия с применением транспортного протокола UDP**

**Вариант № 31**

**Условие:**

Задача о Пути Кулака. На седых склонах Гималаев стоит древний буддистский монастырь: Гуань-Инь-Янь. Каждый год в день сошествия на землю боддисатвы монахи монастыря собираются на совместное празднество и показывают свое совершенствование на Пути Кулака. Всех соревнующихся монахов первоначально разбивают на пары. Бои продолжаются до выявления победителя. Монах который победил в финальном бою, забирает себе на хранение статую боддисатвы.

Реализовать клиент–серверное приложение, определяющего победителя.

Каждый монах — отдельный клиент. Сервер используется для распределения пар и формирования результатов поединка. Проигравший монах–клиент отключается. Победивший — продвигается дальше с новой энергией.

В качестве входных данных используется начальное значение в каждом клиенте, в котором хранится количество энергии Ци каждого монаха. При победе монах забирает энергию Ци своего противника. Новые пары образуются среди победителей других пар в порядке завершения поединков. То есть, возможна ситуация, когда бойцы, участвующие в поединке могут быстро победить и начать биться с другими, в то время как поединки начавшиеся ранее, могут продолжаться. Причем длительное время. Каждый поединок протекает некоторое случайное время, которое пропорционально отношению энергии Ци побежденного к энергии Ци победителя, умноженному на поправочный коэффициент, позволяющий отсле-

живать протекание поединка на экране дисплея (например, путем умножения этого отношения на 1000 миллисекунд или другое более удобное значение).

**Ожидаемая оценка: 10 баллов**

#### **Работа на оценку 4-5**

Разработано 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 системы в целом.

#### **Сценарий решаемой задачи:**

Каждый монах представлен как отдельный клиент, который подключается к серверу, отправляя своё начальное количество энергии Ци. Сервер управляет турниром, формирует пары для поединков и определяет время боя, основываясь на соотношении энергий участников. Поединки продолжаются до тех пор, пока не определится победитель турнира, который заберет статую боддисатвы. Проигравшие монахи отключаются, а победители переходят в следующий раунд с увеличенной энергией. Время каждого поединка зависит от соотношения энергий монахов и может варьироваться в зависимости от исхода боев.

Для этой оценки так же реализованы требования:

- При запуске программ требуемые для их работы IP адреса и порты необходимо задаются в командной строке, чтобы обеспечить гибкую подстройку к любой сети.
- Разработанное приложение работает как на одном компьютере так и в распределенном (сетевом) режиме на нескольких компьютерах, по которым можно будет разнести серверы и клиентов.

Завершение работы клиентов и сервера происходит следующим образом:

Клиенты (монахи):

Клиенты получают от сервера уведомление о победе или поражении. Победители ждут следующего раунда, а проигравшие закрывают соединение и завершают работу. Финальный победитель: получает уведомление о конце турнира, закрывает соединение и завершает работу.

Сервер:

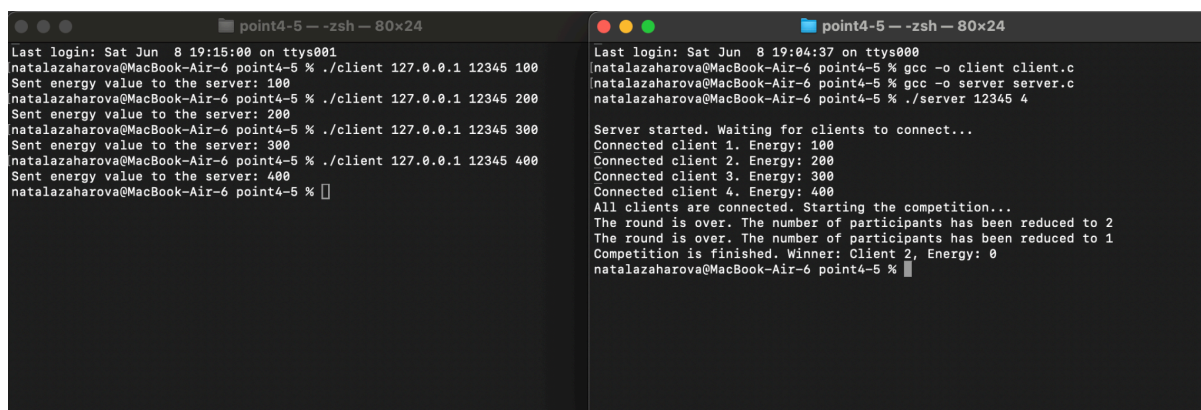
По окончании всех раундов: Отправляет всем оставшимся клиентам (включая победителя) сообщение о завершении турнира. Закрытие сервера: после отправки уведомлений сервер закрывает все активные соединения, освобождает ресурсы и завершает свою работу.

***Более общий и краткий сценарий:***

Сервер запускается с указанием порта и количества ожидаемых участников боя, создает сокет и привязывается к указанному порту, ожидает подключения всех клиентов-игроков. При подключении игроки указывают IP-адрес сервера, порт и количество энергии данного участника. Клиентская программа создает сокет и устанавливает соединение с сервером.

При подключении всех участников проводятся бои, описанные в условии задачи. Раунды проводятся, пока не остается только один игрок, сервер объявляет его победителем и выводит его номер и количество энергии. Сервер закрывает сокет и завершает свою работу. Так как условия отключения клиентов для данной оценки не оговариваются и отсутствует какой-либо общий вывод интегрированной информации, то клиенты отключаются сразу после подключения и передачи числа.

### Пример работы программы:



```
point4-5 -- -zsh -- 80x24
Last login: Sat Jun  8 19:15:00 on ttys001
natalazharova@MacBook-Air-6 point4-5 % ./client 127.0.0.1 12345 100
Sent energy value to the server: 100
natalazharova@MacBook-Air-6 point4-5 % ./client 127.0.0.1 12345 200
Sent energy value to the server: 200
natalazharova@MacBook-Air-6 point4-5 % ./client 127.0.0.1 12345 300
Sent energy value to the server: 300
natalazharova@MacBook-Air-6 point4-5 % ./client 127.0.0.1 12345 400
Sent energy value to the server: 400
natalazharova@MacBook-Air-6 point4-5 % █

point4-5 -- -zsh -- 80x24
Last login: Sat Jun  8 19:04:37 on ttys000
natalazharova@MacBook-Air-6 point4-5 % gcc -o client client.c
natalazharova@MacBook-Air-6 point4-5 % gcc -o server server.c
natalazharova@MacBook-Air-6 point4-5 % ./server 12345 4

Server started. Waiting for clients to connect...
Connected client 1. Energy: 100
Connected client 2. Energy: 200
Connected client 3. Energy: 300
Connected client 4. Energy: 400
All clients are connected. Starting the competition...
The round is over. The number of participants has been reduced to 2
The round is over. The number of participants has been reduced to 1
Competition is finished. Winner: Client 2, Energy: 0
natalazharova@MacBook-Air-6 point4-5 % █
```

### На оценку 6-7 баллов:

В дополнение к программе на предыдущую оценку разработана клиентская программа, подключаемая к серверу, которая предназначена для отображение комплексной информации о выполнении приложения в целом. То есть, данный программный модуль адекватно отображает поведение моделируемой системы, позволяя не пользоваться отдельными видами, предоставляемыми клиентами и серверами по отдельности.

### Изменения, внесенные в программу:

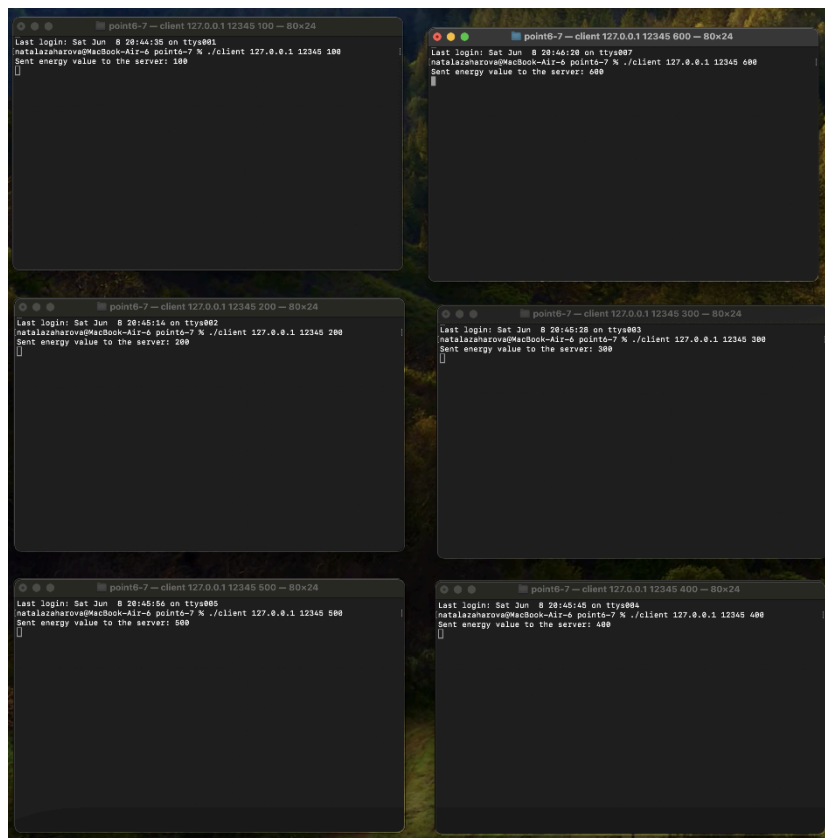
- После того как все участники подключены к серверу, он уведомляет каждого клиента о начале соревнований и присваивает им индивидуальные номера.
- Сервер регулярно информирует клиентов о начале нового раунда и сообщает о количестве участников, которые продолжают борьбу.
- В случае поражения клиента, сервер сообщает ему номер соперника, выигравшего бой.
- По окончании соревнований сервер передает каждому клиенту данные о победителе, включая его номер и суммарную энергию.
- После финала соревнований сервер инициирует закрытие всех клиентских соединений.
- Перед каждым раундом сервер выводит на своей консоли информацию о номерах участников и их текущем уровне энергии.

- Клиенты настроены на прием текстовых сообщений от сервера, что позволяет им получать все необходимые уведомления.
- Клиент закрывает свое соединение после получения последнего сообщения от сервера, которое сигнализирует о завершении соревнований.
- Работа клиентской программы завершается после того, как определен победитель соревнований.

## Пример работы программы:

```
point6-7 — -zsh — 80x24
Last login: Sat Jun  8 20:44:25 on ttys001
natalazaharova@MacBook-Air-6 point6-7 % gcc -o server server.c
natalazaharova@MacBook-Air-6 point6-7 % gcc -o client client.c
natalazaharova@MacBook-Air-6 point6-7 % ./server 12345 6

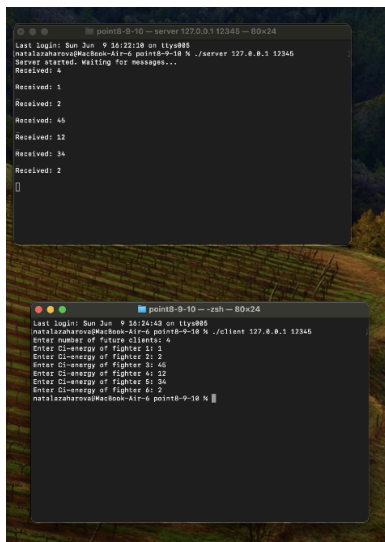
Server started. Waiting for clients to connect...
Connected client 1. Energy: 100
Connected client 2. Energy: 200
Connected client 3. Energy: 300
Connected client 4. Energy: 400
Connected client 5. Energy: 500
Connected client 6. Energy: 600
All clients are connected. Starting the competition...
Failed to send message to a client: Bad file descriptor
natalazaharova@MacBook-Air-6 point6-7 %
```



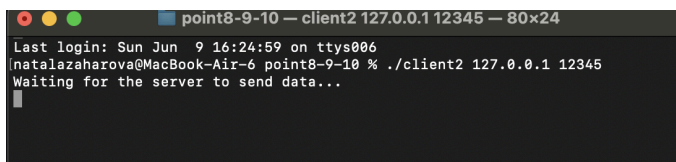
## На 8 баллов

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.

Добавлена возможность ввода будущего количества клиентов



The image shows two terminal windows. The top window is titled 'point8-9-10 - server 127.0.0.1 12345 - 80x24'. It displays the following text: 'Last login: Sun Jun 9 16:22:18 on ttys006', 'natalazharova@MacBook-Air-6 point8-9-10 % ./server 127.0.0.1 12345', 'Server started. Waiting for messages...', 'Received: 4', 'Received: 1', 'Received: 2', 'Received: 46', 'Received: 12', 'Received: 34', 'Received: 2', and a cursor. The bottom window is titled 'point8-9-10 - ssh - 80x24'. It displays: 'Last login: Sun Jun 9 16:24:49 on ttys006', 'natalazharova@MacBook-Air-6 point8-9-10 % ./client 127.0.0.1 12345', 'Enter number of future clients: 4', 'Enter Cl-energy of fighter 1: 1', 'Enter Cl-energy of fighter 2: 2', 'Enter Cl-energy of fighter 3: 46', 'Enter Cl-energy of fighter 4: 12', 'Enter Cl-energy of fighter 5: 34', 'Enter Cl-energy of fighter 6: 2', and a cursor.



The image shows a terminal window titled 'point8-9-10 - client2 127.0.0.1 12345 - 80x24'. It displays the following text: 'Last login: Sun Jun 9 16:24:59 on ttys006', 'natalazharova@MacBook-Air-6 point8-9-10 % ./client2 127.0.0.1 12345', 'Waiting for the server to send data...', and a cursor.