Задание на лабораторную работу №6.

Общие моменты для задачи, справедливые для всех вариантов:

При вызове программы клиента задается hostname (или IP) и порт для связи с сервером (можно в самой программе). Передача информации между клиентом и сервером происходит через потоковые сокеты (TCP). Для хранения главной структуры (struct) программы использовать разделяемую память, доступ к которой есть у всех процессов сервера. Для блокирования доступа к разделяемой памяти использовать семафоры. Сервер оформить в виде демона с логированием действий сервера. На сервере всегда существует master-процесс, который создает и уничтожает дочерние процессы-worker’ы. Из конфигурационного файла сервер читает порт, по которому происходит взаимодействие, а также имя файла для лога (к прочитанному имени файла в конец добавляется временная метка, полученная во время чтения (или перечитывания) конфигурационного файла). Сервер всегда принимает SIGTERM для корректного завершения работы (необходимо отдать результат работы клиенту, после этого только завершать работу сервера), при получении SIGHUP сервер перечитывает конфигурационный файл, обработка SIGHUP блокируется на время активных действий сервера. Для генерации любых случайных чисел использовать /dev/urandom.

Вариант 3\*\*\*. Односторонний «морской бой» с сервером. (Абдулкин)

Общение клиента и сервера проходит посредством диалога. Клиент отправляет серверу команду типа «battle {quantity}», где {quantity} – уровень сложности, означающий максимально допустимое количество промахов, за которое клиент должен уничтожить все корабли. На сервере мастер создает воркера, который случайным образом генерирует положение кораблей, после чего должно быть сформировано корректно заполненное игровое поле, затем клиент отправляет клиенту сообщение о начале игры. Клиент отправляет команды серверу по типу «shot (x,y)», где х и у – координаты строки и столбца. На сервере мастером производится проверка корректности команды (при некорректности клиенту отправляется сообщение и счетчик попыток не меняется). Если все нормально, то в очередь сообщений пишутся координаты для проверки и текущее значение счетчика, создается воркер, который читает сообщение из очереди, определяет промах/попадание и отправляет сообщение в очередь с текущим значением счетчика оставшихся промахов (счетчик уменьшается только в случае промаха). После каждой отработки воркера мастер проверяет игровое поле и значение счетчика для определения конца игры. Если игра продолжается, то сервер отправляет строкой текущее поле клиента с пометками выстрелов, а также значение счетчика оставшихся попыток. Если наступил конец игры, сервер оповещает об этом клиента и блокирует прием любых команд, кроме «battle {quantity}».

Для всех вариантов с 2 клиентами алгоритма старта игры следующий: 1-ый клиент будет создавать игру, для этого он пишет команду «start {password}», где {password} – пароль от игры, которая будет создана. Второй клиент подключается к игре, вводя команду «join {password}». Считаем, что сервер может держать только 1 активную игру. Если игр на сервере не существует клиент ввел неправильный пароль или присоединилось максимальное число игроков (2), сервер должен уведомить об этом клиента. Общий алгоритм всех таких игр: в начале возможна генерация сервером некоторых начальных условий (положение кораблей, загаданное слово, число и т.д.), мастер для этого создает воркера, который генерирует то что нужно и записывает в главную структуру, после чего начинается фаза «перестрелки» - клиент 1 делает какое-то действие (вводит команду), затем клиент 2 делает какое-то действие, после этого на сервере происходит обработка текущей ситуации (это цикличная фаза). Предусмотреть, чтобы клиент, который пытается сделать ход в не в свой черед, получил соответствующее уведомление от сервера (такие проверки делает отдельный воркер). Соответственно, когда воркер проверил команду на синтаксическую и логическую корректность, через очередь сообщений он отсылает мастеру сообщение, в зависимости от которого мастер либо шлет клиенту сообщение об ошибке, либо создает воркера, которому через очередь сообщений посылает команду. Воркер не удаляется, пока не обработает обе команды (обрабатывает он их и отдает результат по отдельности, кроме игр «Угадай число» и «Тривиадор», в которых результат хода оба клиента должны получить только после того, как оба сделают ход). Соответственно после обработки либо 1 команды, либо сразу 2 (в зависимости от игры) воркер обновляет необходимые структуры данных, кладет в очередь сообщений сообщения с текстом для клиентов, мастер вычитывает их и посылает клиентам. Если воркер определил конец игры, он удаляет игру и посылает клиентам результат игры.

Можно реализовать команду disconnect (по желанию), которая позволит отключиться от игры, которую сервер автоматически расформирует.

Вариант 6\*\*\*. Реализация двусторонней игры (2 клиента) «Угадай число». (Махмутов)

Логика игры: сервер загадывает число, оба клиента вводят ответы, не видя ответы друг друга. Верхний диапазон загаданного числа задается константой в программе. На каждую попытку сервер ведет счетчик и отправляет подсказку клиенту «больше/меньше». Тот, кто быстрее по попыткам (подсчет времени не нужен) отгадывает число, тот и победил, если они делают это в одну и ту же попытку, то победила дружба.

Вариант 9\*\*\*. Реализация двусторонней игры (2 клиента) «Серия пенальти» (Фетисов)

Считаем, что ворота разделены на 10 зон. Все зоны пронумерованы и указаны в таблице. Зона 0 – зона реагирования вратаря. Если у зоны рядом с номером стоит %, это процент попасть в створ при выборе этой зоны для удара. Считаем, что шанс попадания в зоны 0, 2, 3 при их выборе равен 100%. Но для баланса вводится следующее правило: если вратарь выбрал зону 0 (то есть реагировать на удар), а мяч летит в зону 2 или 3 у него есть 70%-й шанс отбить мяч. Если мяч летит в зоны 1 или 4, а вратарь выбрал зону 0, то шанс отбить мяч 50%. Принцип игры следующий: 1-й игрок выбирает зону для удара, 2-й игрок после этого выбирает зону для отбивания. Считаем, что если вратарь выбрал правильную зону, то он со 95% вероятностью отобьет мяч, если он туда полетит.