**Отчёт о выполнении индивидуального домашнего задания №3**

**Вариант №5**

**(Задание на 10 баллов)**

Отчёт подготовлен студентом ФКН БПИ228

Лысиным Кириллом

**Условие задания**

Задача о читателях и писателях. Базу данных, представленную массивом целых положительных чисел, разделяют два типа процессов: N читателей и K писателей. Читатели периодически просматривают случайные записи базы данных и выводя номер свой номер (например, PID), индекс записи, ее значение, а также 7 вычисленное значение числа Фибоначчи. Писатели изменяют случайные записи на случайное число и также выводят информацию о своем номере, индексе записи, старом значении и новом значении. Предполагается, что в начале БД находится в непротиворечивом состоянии (все числа отсортированы, например, по возрастанию). Каждая отдельная новая запись переводит БД из одного непротиворечивого состояния в другое (то есть, новая сортировка может поменять индексы записей или переставить числа). Для предотвращения взаимного влияния транзакций процесс–писатель должен иметь исключительный доступ к БД. Если к БД не обращается ни один из процессов–писателей, то выполнять транзакции могут одновременно сколько угодно читателей. Создать клиент–серверное приложение с процессами–писателями и процессами–читателями. Сервер моделирует базу данных. Все писатели и все читатели — два разных клиента, в каждом из которых возможна конкуренция параллельных процессов или потоков. Каждый процесс — это отдельный писатель или отдельный читатель внутри сервера.

**Общая схема решения**

Задача заключается в создании клиент-серверного приложения для моделирования базы данных с множеством читателей и писателей.

**Сценарий работы системы:**

**Сущности:**

* Сервер: Центральный узел, управляющий базой данных и обрабатывающий запросы от клиентов и мониторов. Сервер отвечает за целостность данных и координирует взаимодействие между клиентами и мониторами.
* Клиенты (Читатели, писатели и мониторы): Подключаются к серверу для чтения или записи данных. Читатели считывают значения из базы данных, а писатели изменяют значения в базе данных. Мониторы подключаются к серверу для мониторинга и отслеживания состояния сервера и базы данных.

**Поведение:**

* Сервер: Принимает входящие соединения от клиентов и мониторов, обрабатывает их запросы и отправляет соответствующие ответы. Он также управляет синхронизацией доступа к базе данных, чтобы избежать конфликтов при одновременном чтении и записи. Кроме того, сервер отправляет данные мониторам для отображения текущего состояния системы.
* Клиенты (Читатели и Писатели): Подключаются к серверу и отправляют запросы на чтение или запись данных, взаимодействуя с базой данных через сервер. Читатели периодически считывают случайные значения из базы данных, вычисляют для них числа Фибоначчи и выводят результаты. Писатели периодически изменяют случайные значения в базе данных. Клиенты могут подключаться к серверу и отключаться от него. **Программа позволяет подключать и отключать клиенты без утраты работоспособности. При завершении работы все подключенные клиенты автоматически завершают свою работу.**
* Мониторы: Подключаются к серверу для отслеживания и мониторинга его работы. Они получают информацию о состоянии сервера и базы данных, чтобы отображать текущую нагрузку, количество активных подключений, а также обнаруживать и реагировать на возможные проблемы.

**Взаимодействие:**

Клиенты, мониторы и сервер взаимодействуют через сетевые соединения.

Клиенты и мониторы подключаются к серверу и отправляют запросы на чтение, запись данных или мониторинг состояния сервера.

Сервер принимает запросы от клиентов и мониторов, обрабатывает их и отправляет соответствующие ответы. Он также отправляет данные мониторам для отображения текущего состояния системы.

В случае параллельной работы читателей и писателей сервер обеспечивает эксклюзивный доступ к базе данных во время операций записи. Таким образом, клиент-серверное приложение с мониторами моделирует взаимодействие множества пользователей с базой данных, при этом сервер управляет этим взаимодействием и обеспечивает правильное выполнение операций, а мониторы отслеживают и отображают текущее состояние системы.

**Завершение программы:**

Программа написана с применением подхода “graceful termination”: при завершении программы происходит корректное закрытие всех активных соединений, освобождение ресурсов и завершение процессов без потери данных или состояния.

**Запуск программы:**

./server <IP > <port>

./writer\_client <IP> <port> <WRITERS\_COUNT>

./reader\_client <IP> <port> <READERS\_COUNT>

./monitor\_client <IP> <port>

Результаты работы программы находятся в папке **Results.**