# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

## Факультет физико-математических и естественных наук

### Кафедра прикладной информатики и теории вероятностей

## ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 15

### *дисциплина: Операционные системы*

Студент: Ким Реачна Группа: НПИбд-02-20

Москва 2021г.

### Цель работы:

Приобретение практических навыков работы с именованными каналами.

### Теоретичекое введение:

Одним из видов взаимодействия между процессами в операционных системах является обмен сообщениями. Под сообщением понимается последовательность байтов, передаваемая от одного процесса другому.

В операционных системах типа UNIX есть 3 вида межпроцессорных взаимодействий: общеюниксные (именованные каналы, сигналы System V Interface Definition (SVID — разделяемая память, очередь сообщений семафоры) и BSD (сокеты).

Для передачи данных между неродственными процессами можно использовать механизм именованных каналов (named pipes). Данные передаются по принципу FIFO (First In First Out) (первым записан — первым прочитан), поэтому они называются также FIFO pipes или просто FIFO. Именованные каналы отличаются от неименованных наличием идентификатора канала, который представлен как специальный файл (соответственно имя именованного канала — это имя файла). Поскольку файл находится на локальной файловой системе, данное IPC используется внутри одной системы.

Файлы именованных каналов создаются функцией mkfifo(3).

#include <sys/types.h>  
#include <sys/stat.h>  
int mkfifo(const char \*pathname, mode\_t mode);

Первый параметр — имя файла, идентифицирующего канал, второй параметр — маска прав доступа к файлу.

После создания файла канала процессы, участвующие в обмене данными, должны открыть этот файл либо для записи, либо для чтения. При закрытии файла сам канал продолжает существовать. Для того чтобы закрыть сам канал, нужно удалить его файл, например с помощью вызова unlink(2).

Вызов функции mkfifo() создаёт файл канала (с именем, заданным макросом FIFO\_NAME):

mkfifo(FIFO\_NAME, 0600);

В качестве маски доступа используется восьмеричное значение 0600, разрешающее процессу с аналогичными реквизитами пользователя чтение и запись. Можно также установить права доступа 0666.

Открываем созданный файл для чтения:

f = fopen(FIFO\_NAME, O\_RDONLY);

Клиент открывает FIFO для записи как обычный файл:

f = fopen(FIFO\_NAME, O\_WRONLY);

Более подробная информация о лаборатории № 15. [1](https://esystem.rudn.ru/pluginfile.php/1142389/mod_resource/content/1/013-ipc-fifo.pdf)

### Задание:

Изучите приведённые в тексте программы server.c и client.c. Взяв данные примеры за образец, напишите аналогичные программы, внеся следующие изменения:

1. Работает не 1 клиент, а несколько (например, два).
2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). Используйте функцию sleep() для приостановки работы клиента.
3. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек). Используйте функцию clock() для определения времени работы сервера. Что будет в случае, если сервер завершит работу, не закрыв канал?

### Выполнение работы:

1. Создайте файлы server.c, client.c common.h и Makefile с помощью редактора команд Vi *(Рисунок 1)*.

*Рисунок 1: Создайте файлы*

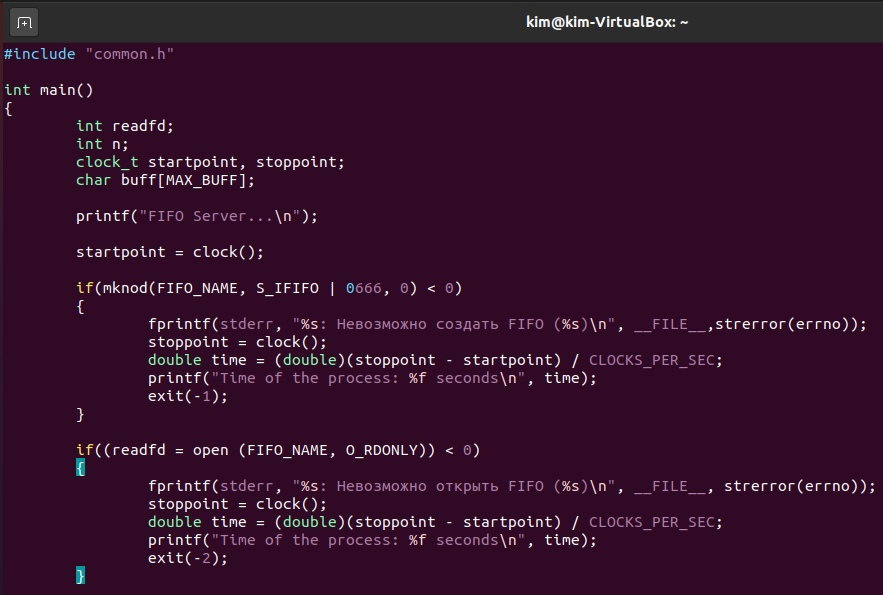


*Рисунок 2: Файл common.h*

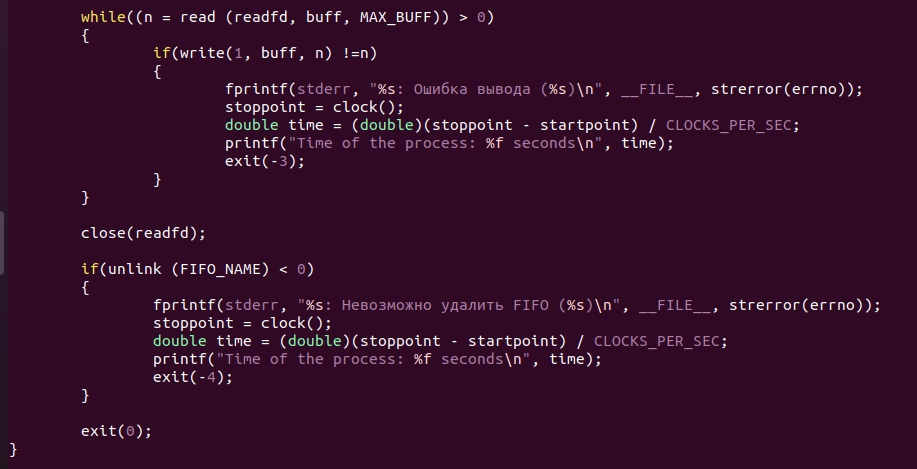
В этом общем файле.h я добавил #include <time.h> и #include <unisted.h>, чтобы мы могли запустить работу командного файла.*(Рисунок 2)*



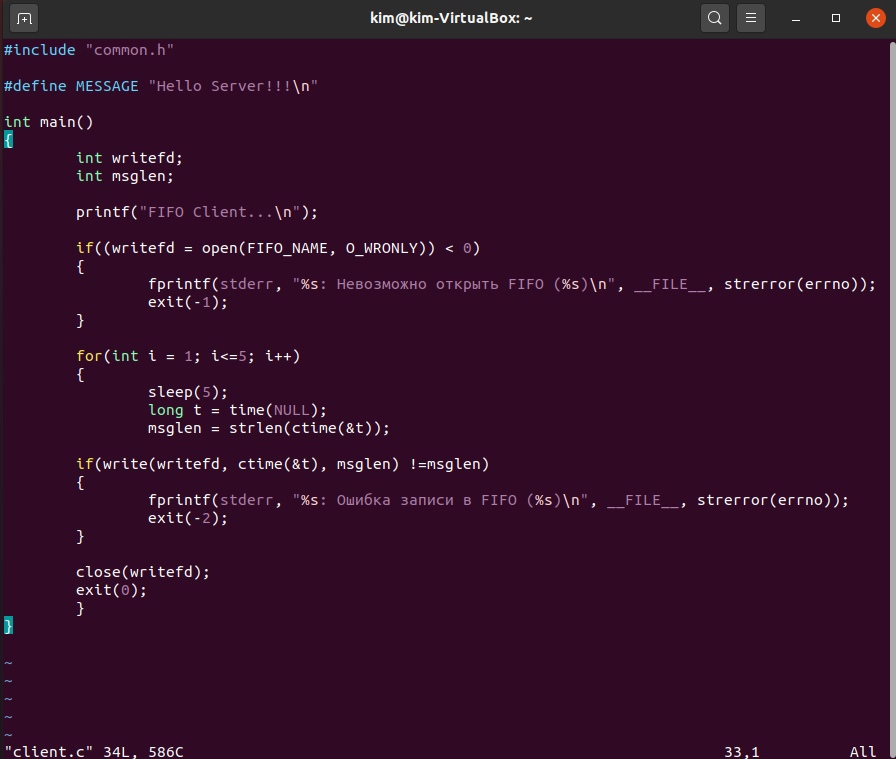
*Рисунок 3: Файл server.c*



*Рисунок 4: Файл server.c*



*Рисунок 5: Файл client.c*



*Рисунок 6: Проверка файлов*

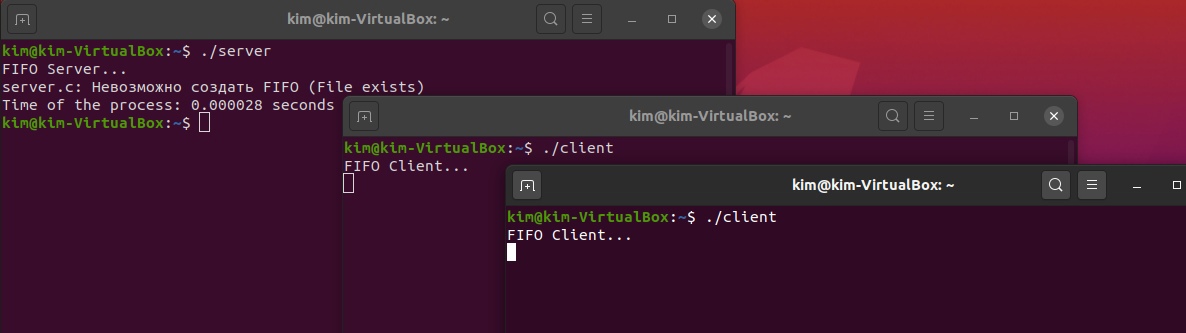
Используя команды gcc-c server.c и gcc -c client.c, чтобы проверить, есть ли какие-либо ошибки в наших командных файлах *(Рисунок 6)*.



Как мы видим, в наших командных файлах нет ошибок.

1. Проверьте нашу работу: после проверки того, что в наших командных файлах нет ошибок, теперь мы проверим нашу целевую работу с помощью команд ./server и ./client *(Рисунок 7)*.

*Рисунок 7: Работа над файлом*



### Вывод:

Я познакомилаль с практическими навыками работы с названными каналами.

#### Библиография:

[1]:[Материал лаборатории 15](https://esystem.rudn.ru/pluginfile.php/1142389/mod_resource/content/1/013-ipc-fifo.pdf)

[2]:[Последствия, объявленные функциями в нашем файле command.h](https://ru.stackoverflow.com/questions/213706/warning-implicit-declaration-of-function-read-write)

[3]:[Именованный канал](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%B5%D0%BD%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB)

### Контрольные вопросы:

1. Именованные каналы, в отличие от неименованных, могут использоваться неродственными процессами. Они дают вам, по сути, те же возможности, что и неименованные каналы, но с некоторыми преимуществами, присущими обычным файлам.
2. Да, для создания неименованного канала используется системный вызов pipe.
3. Да, $ mkfifo [имя\_файла]
4. Опишите функцию языка С, создающую неименованный канал.

int read(int pipe\_fd, void \*area, int cnt);  
int write(int pipe\_fd, void \*area, int cnt);

1. Опишите функцию языка С, создающую именованный канал.

int mkfifo (const char \*pathname, mode\_t\_mode)

1. При чтении большего числа байтов, чем находится в канале или FIFO, возвращается доступное число байтов. Процесс, читающий из канала, должен соответствующим образом обработать ситуацию, когда прочитано меньше, чем заказано.
2. При чтении меньшего числа байтов, чем находится в канале или FIFO, возвращается требуемое число байтов, остаток сохраняется для последующих чтений.
3. орожденные процессы-братья: родительский вызывает pipe для создания канала, затем порождает два или больше процессов-братьев. Порожденные процессы могут сообшаться по каналу посредством своих дескрипторов fifo[0] и fifo[1].
4. Функция записывает length байтов из буфера buffer в файл, определенный дескриптором файла fd. Эта операция реализуется как непосредственный вызов DOS. С помощью функции write мы посылаем сообщение клиенту или серверу.
5. Функция strerror() возвращает строку, описывающую код ошибки, переданный в аргументе errnum, возможно с учетом категории LC\_MESSAGES текущей локали для выбора соответсвующего языка.