Отчет по лабораторной работе №15

Тема:

Именованные каналы

Российский Университет Дружбы Народов

Факультет Физико-Математических и Естественных Наук

Дисциплина: Операционные системы

Студент: Мухамедияр Адиль

Группа: НКНбд-01-20

Москва, 2021г.

Цель работы

Приобретение практических навыков работы с именованными каналами.

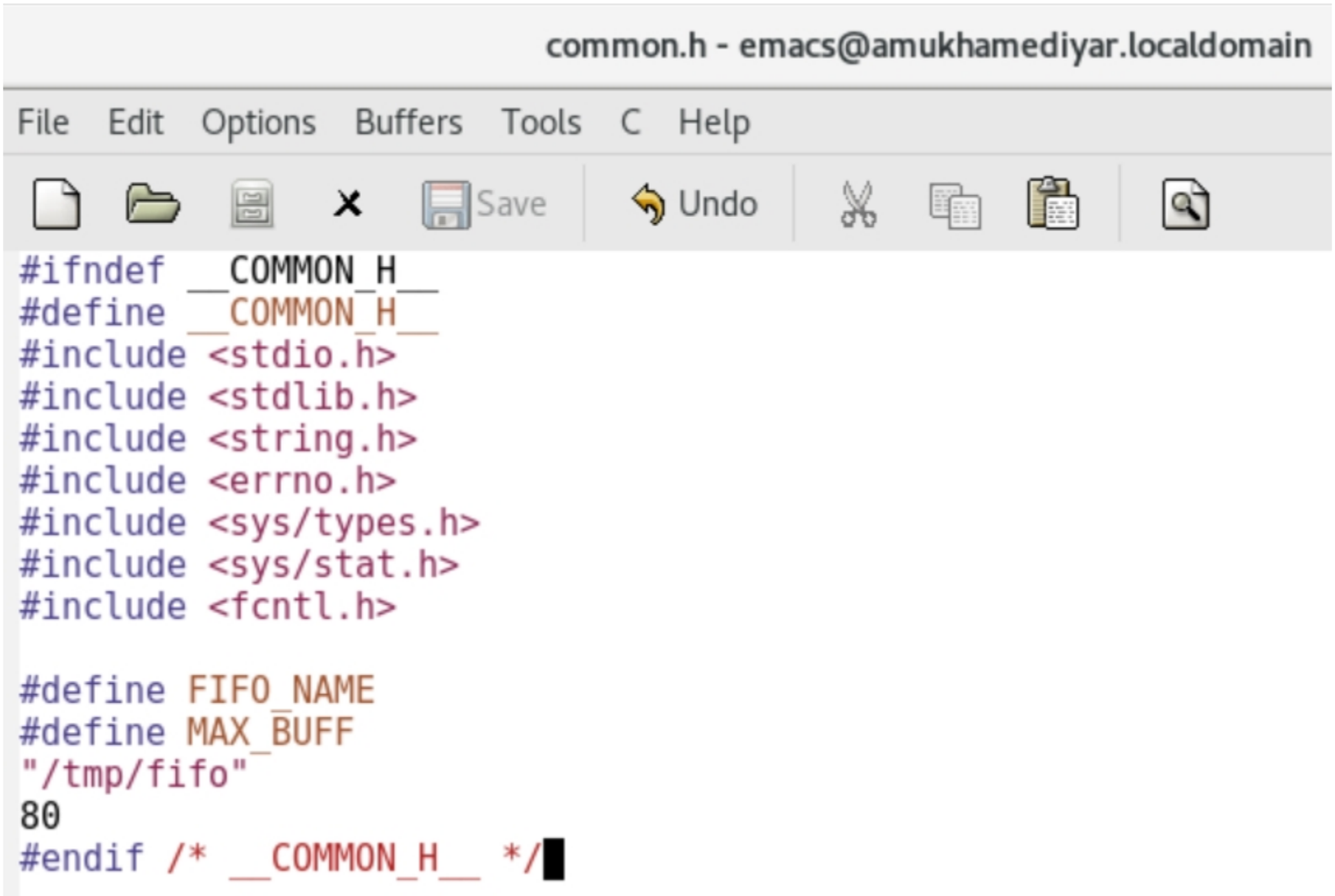
Введение

- Идею программных каналов и значок вертикальной черты как их обозначение придумал Douglas McIlroy, один из авторов ранних командных оболочек. Он обратил внимание на то, сколько времени уходит на обработку вывода одной программы в качестве ввода другой. Его идеи были внедрены в жизнь, когда в 1973 Ken Thompson добавил программные каналы в операционную систему Юникс. Идея была со временем позаимствована другими ОС, такими как DOS, OS/2, Microsoft Windows, и BeOS, часто даже с тем же обозначением.

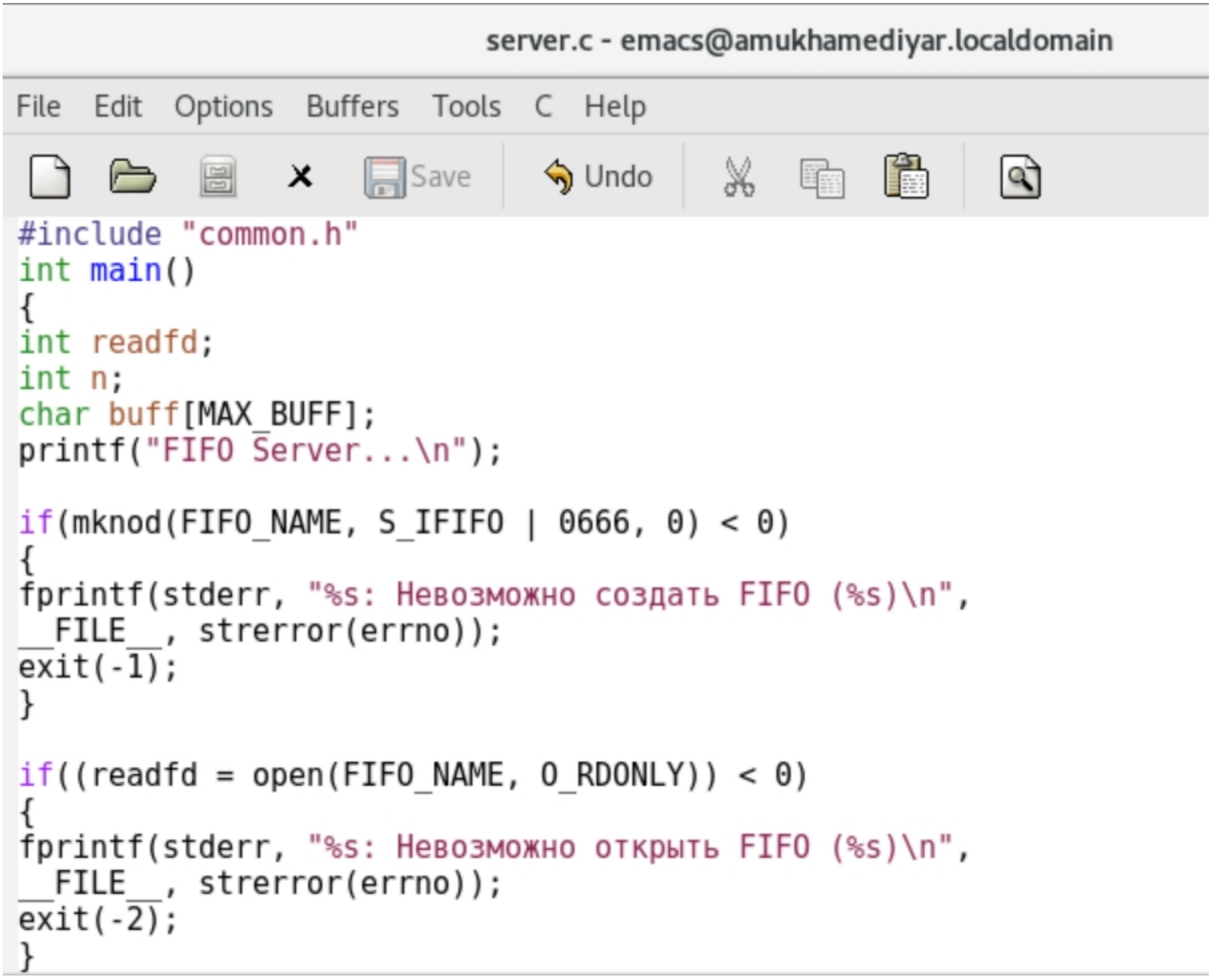
- Английское название именованного канала - named pipe или FIFO (File In, File Out - файл пришел, файл ушел). Именованные каналы служат в основном для межпроцессного взаимодействия, когда различные процессы в системе обмениваются информацией. Тема это сложная и большая, заслуживающая отдельной статьи. Поэтому в данной работе я только вкратце коснусь ее.

Ход работы:

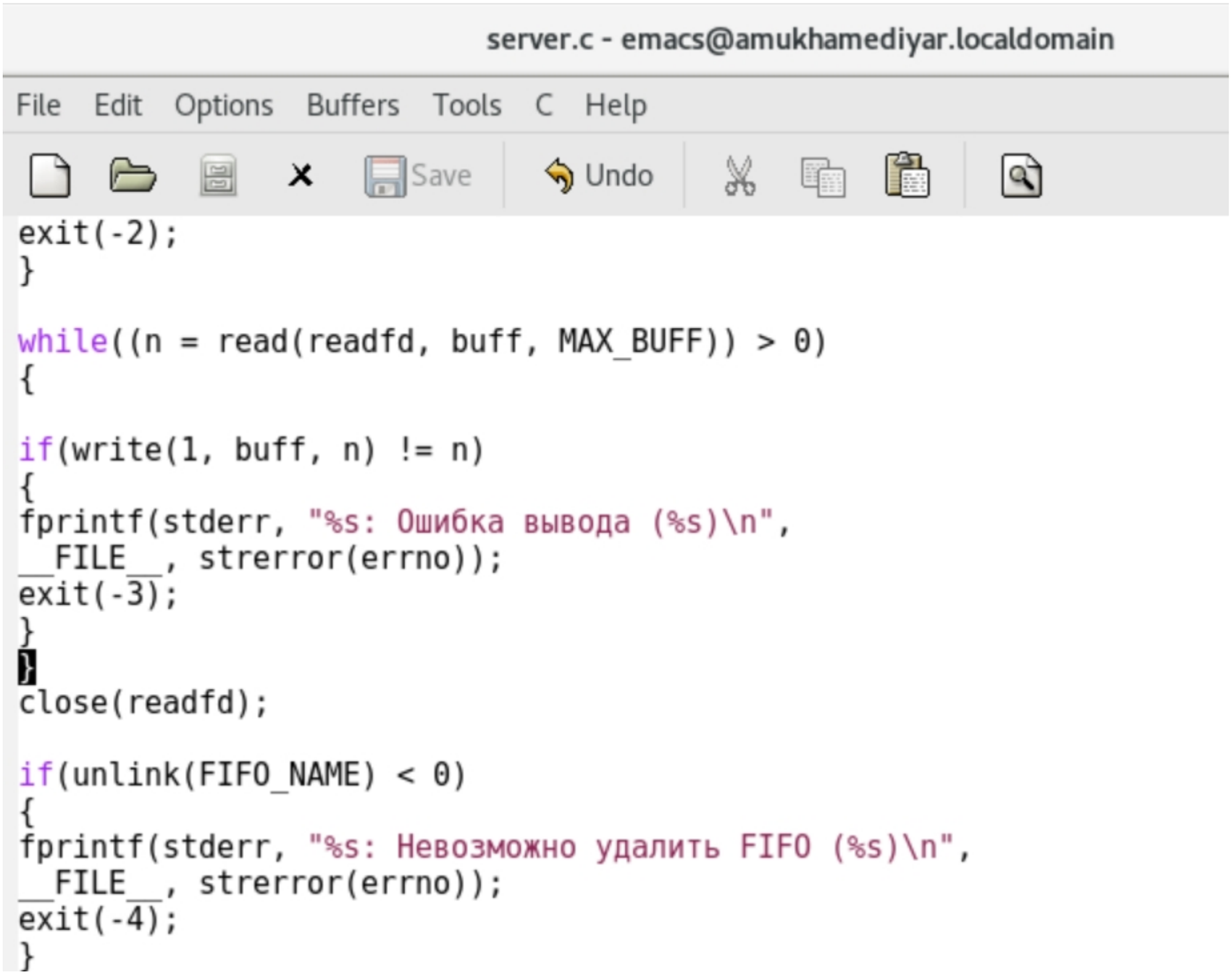
1. Изучил приведённые в тексте программу *common.h* и взял данный пример за образец.



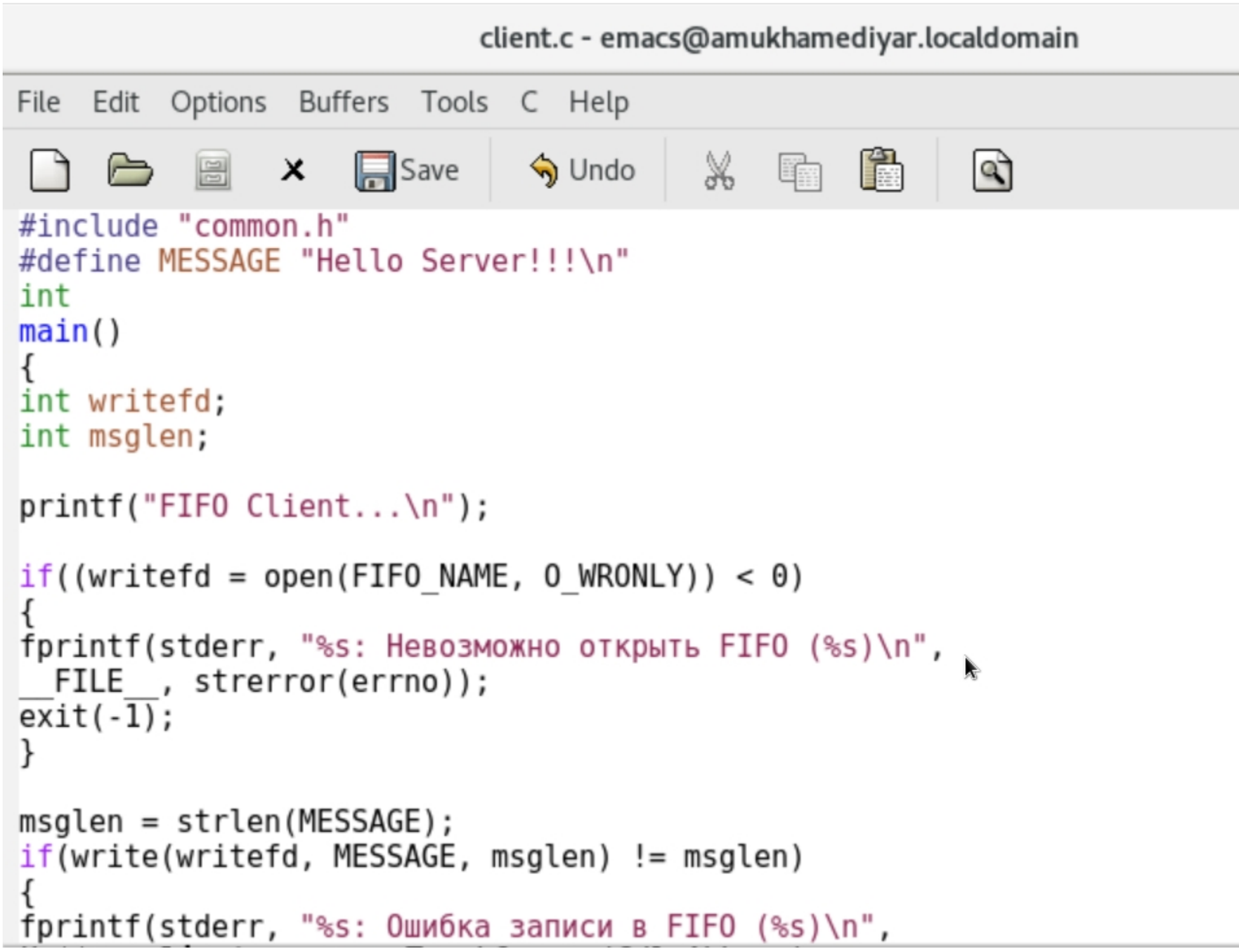
- Изучил приведённые в тексте программу *server.c* и взял данный пример за образец.



- Продолжение кода файла *server.c*.



- Изучил приведённые в тексте программу *client.c* и взял данный пример за образец.



- Продолжение кода файла *client.c*.

client.c - emacs@amukhamediyar.localdomain

File Edit Options Buffers Tools C Help

Save Undo

```
fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
__FILE__, strerror(errno));
exit(-2);
}

close(writefd);
exit(0);
}
```

2. Написал аналогичные программы, внеся следующие изменения:

- В коде файла *common.h* добавил библиотеку *time.h*, для использования таких функций, как *sleep* и *clock*.

common.h - emacs@amukhamediyar.localdomain

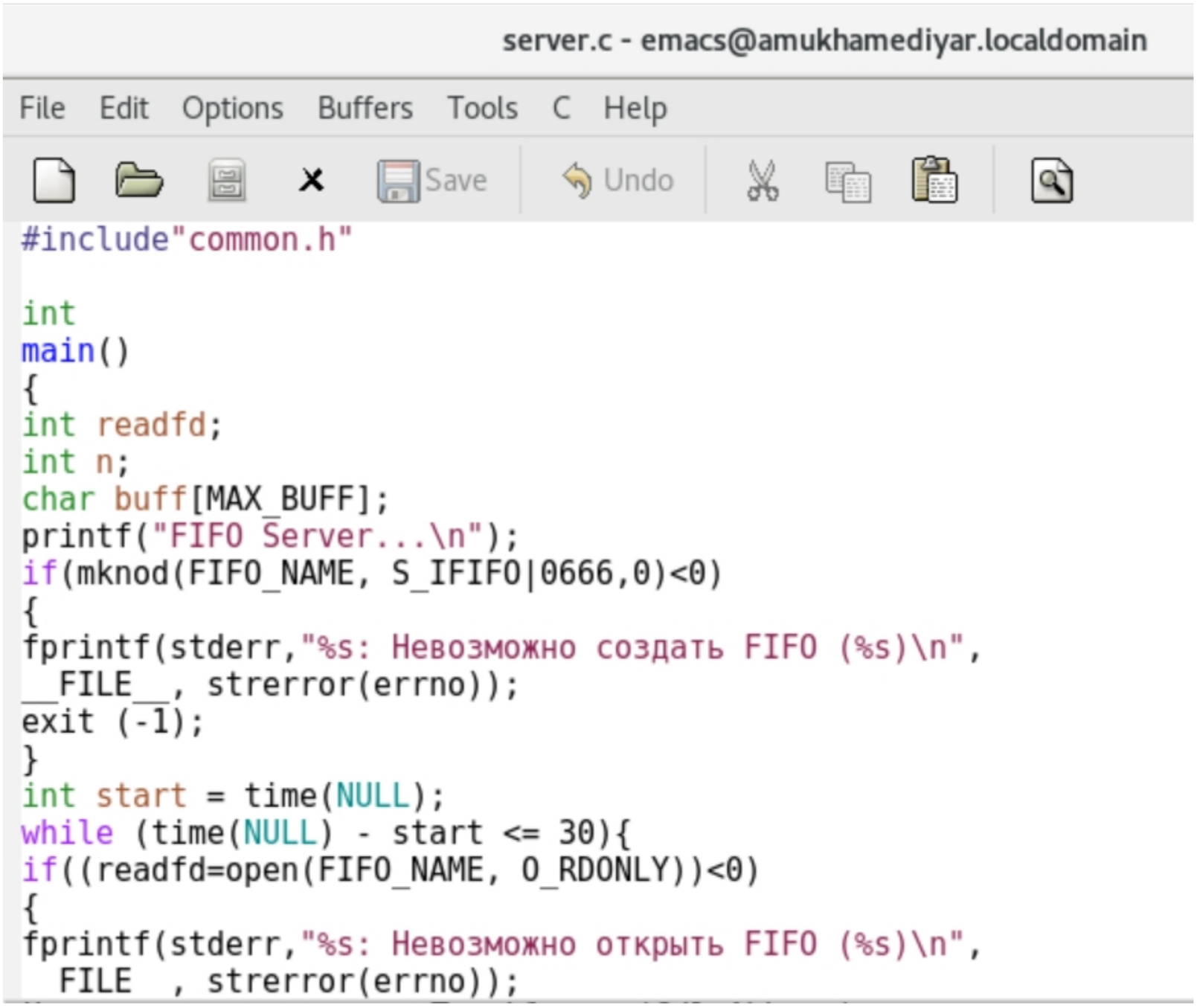
File Edit Options Buffers Tools C Help

Save Undo

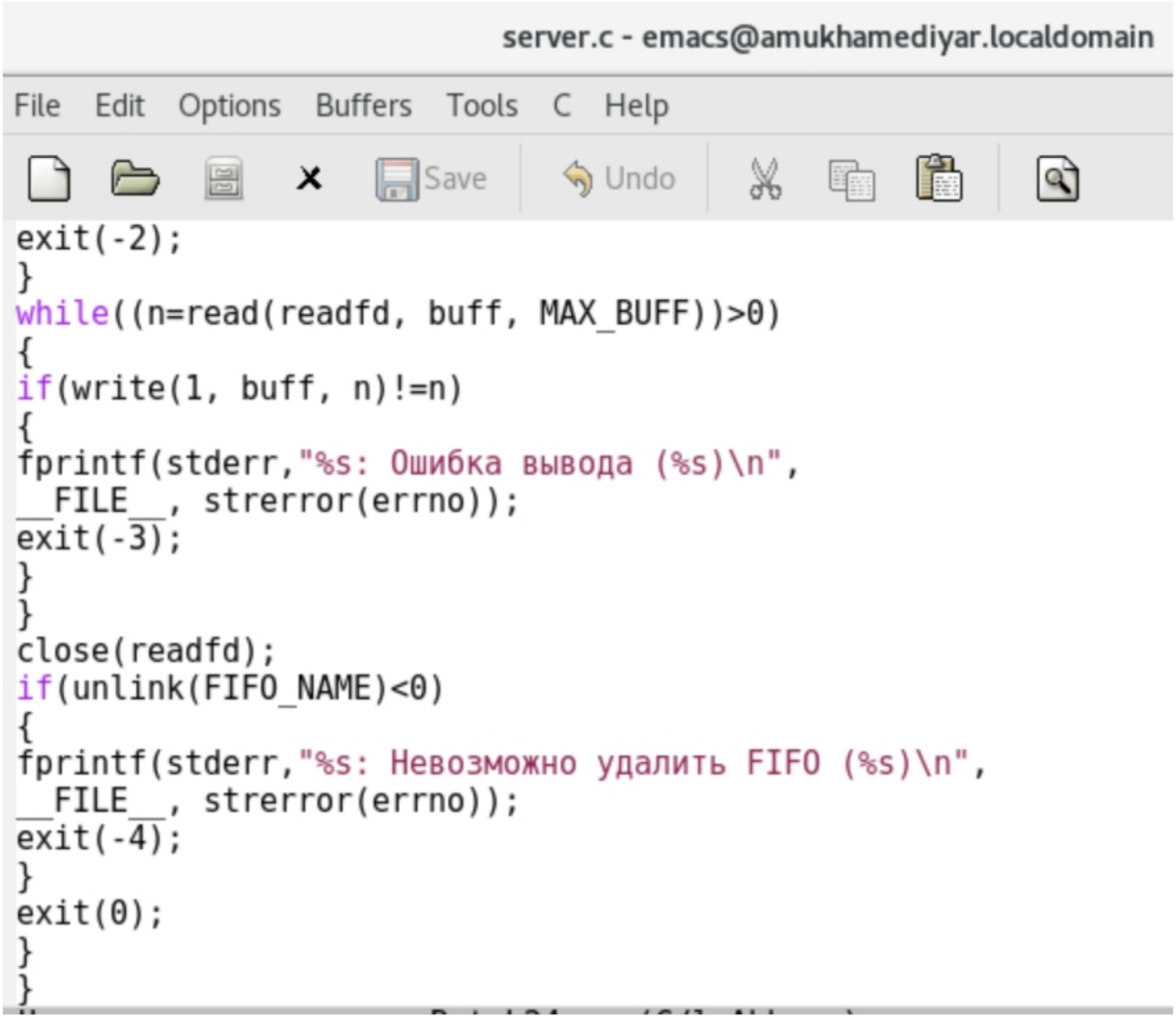
```
#ifndef __COMMON_H__
#define __COMMON_H__
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <time.h>

#define FIFO_NAME "/tmp/fifo"
#define MAX_BUFF 80
#endif /* __COMMON_H__ */
```

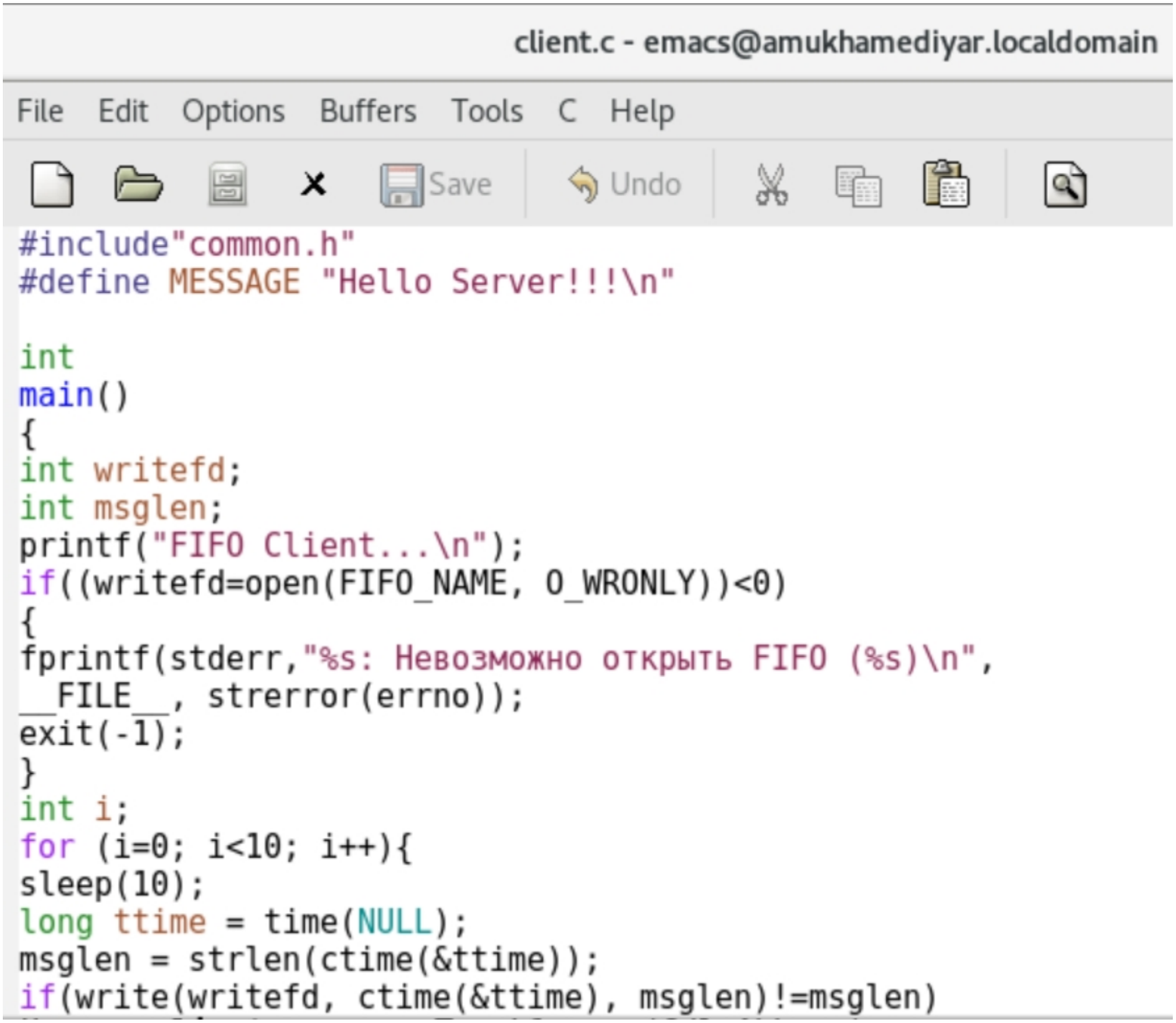
- В код файла *server.c* так же ввел некоторые изменения.



- Продолжение кода файла *server.c*. Сервер работает не бесконечно, а прекращает работу через некоторое время.



- В коде файла *client.c* использовал функцию sleep() для приостановки работы клиента. Так же приравнял переменную *ttime* к нулю, задав его в нутри цикла.



- Продолжение кода файла *client.c*.

client.c - emacs@amukhamediyar.localdomain

File Edit Options Buffers Tools C Help

Save Undo

```
if(write(writefd, ctime(&ttime), msglen)!=msglen)
{
fprintf(stderr,"%s: Ошибка записи в FIFO (%s)\n",
__FILE__, strerror(errno));
exit(-2);
}
}
close(writefd);
exit(0);
}
```

- С помощью команды *make* запустил *makefile*.

makefile - emacs@amukhamediyar.localdomain

File Edit Options Buffers Tools Makefile Help

Save Undo

```
all: server client

server: server.c common.h
	gcc server.c -o server

client: client.c common.h
	gcc client.c -o client

clean:
	-rm server client *.o
```

- Проверяю, запустил коды в файлах *server.c* и *client.c*

```
[amukhamediyar@amukhamediyar lab_15]$ ./server
FIFO Server...
server.c: Невозможно создать FIFO (File exists)
[amukhamediyar@amukhamediyar lab_15]$
```

- В случае, если сервер завершит работу, не закрыв канал, файл FIFO не удалится, поэтому его в следующий раз создать будет нельзя и вылезет ошибка, следовательно, работать ничего не будет.

```
amukhamediyar@amukh...   amukhamediyar@amukh...
[amukhamediyar@amukhamediyar lab_15]$ ./client
FIFO Client...
```

Вывод

Приобрел практические навыки работы с именованными каналами.

Библиография

[Программные каналы в Linux]

[Введение в каналы и именованные каналы в Linux]

Ответы на контрольные вопросы:

1. 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.
2. Создание неименованного канала из командной строки невозможно.
3. Создание именованного канала из командной строки возможно.
4. int read(int pipe_fd, void *area, int cnt); int write(int pipe_fd, void *area, int cnt); Первый аргумент этих вызовов - дескриптор канала, второй - указатель на область памяти, с которой происходит обмен, третий - количество байт. Оба вызова возвращают число переданных байт (или -1 - при ошибке).
5. int mkfifo (const char *pathname, mode_t mode) ; mkfifo(FIFO_NAME, 0600) ; Первый параметр — имя файла, идентифицирующего канал, второй параметр маска прав доступа к файлу. Вызов функции mkfifo() создаёт файл канала (с именем, заданным макросом FIFO_NAME).
6. При чтении меньшего числа байтов, чем находится в канале, возвращается требуемое число байтов, остаток сохраняется для последующих чтений. При чтении большего числа байтов, чем находится в канале или FIFO возвращается доступное число байтов.
7. При записи большего числа байтов, чем это позволяет канал или FIFO, вызов write(2) блокируется до освобождения требуемого места. При этом атомарность операции не гарантируется. Если процесс пытается записать данные в канал, не открытый ни одним процессом на чтение, процессу генерируется сигнал. Запись числа байтов, меньшего емкости канала или FIFO, гарантированно атомарно. Это означает, что в случае, когда несколько процессов одновременно записывают в канал, порции данных от этих процессов не перемешиваются.
8. В общем случае возможна много направленная работа процессов с каналом, т.е. возможна ситуация, когда с одним и тем же каналом взаимодействуют два и более процесса, и каждый из взаимодействующих каналов пишет и читает информацию в канал. Но традиционной схемой организации работы с каналом является однонаправленная организация, когда канал связывает два, в большинстве случаев, или несколько взаимодействующих процесса, каждый из которых может либо читать, либо писать в канал.
9. Write - Функция записывает length байтов из буфера buffer в файл, определенный дескриптором файла fd. Эта операция чисто 'двоичная' и без буферизации. Реализуется как непосредственный вызов DOS. С помощью функции write мы посылаем сообщение

клиенту или серверу.

10. Строковая функция strerror - функция языков C/C++, транслирующая код ошибки, который обычно хранится в глобальной переменной errno, в сообщение об ошибке, понятном человеку. Ошибки эти возникают при вызове функций стандартных Си-библиотек. Возвращенный указатель ссылается на статическую строку с ошибкой, которая не должна быть изменена программой. Дальнейшие вызовы функции strerror перезапишут содержание этой строки. Интерпретированные сообщения об ошибках могут различаться, это зависит от платформы и компилятора.