

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: Сулицкий Богдан Романович

Группа: НФИбд-02-20

МОСКВА

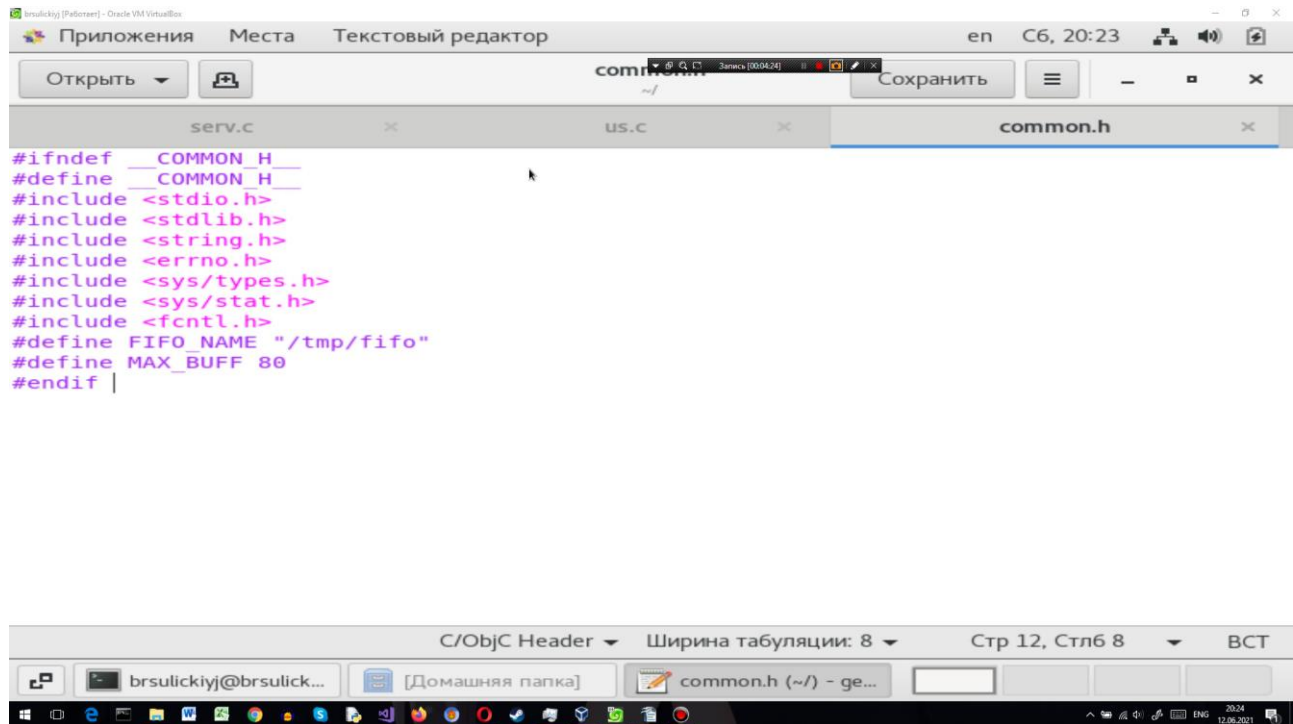
2021 г.

Цель работы: приобретение практических навыков работы с сокетами.

Ход работы:

1. Изучил приведённые в тексте программы server.c и client.c.

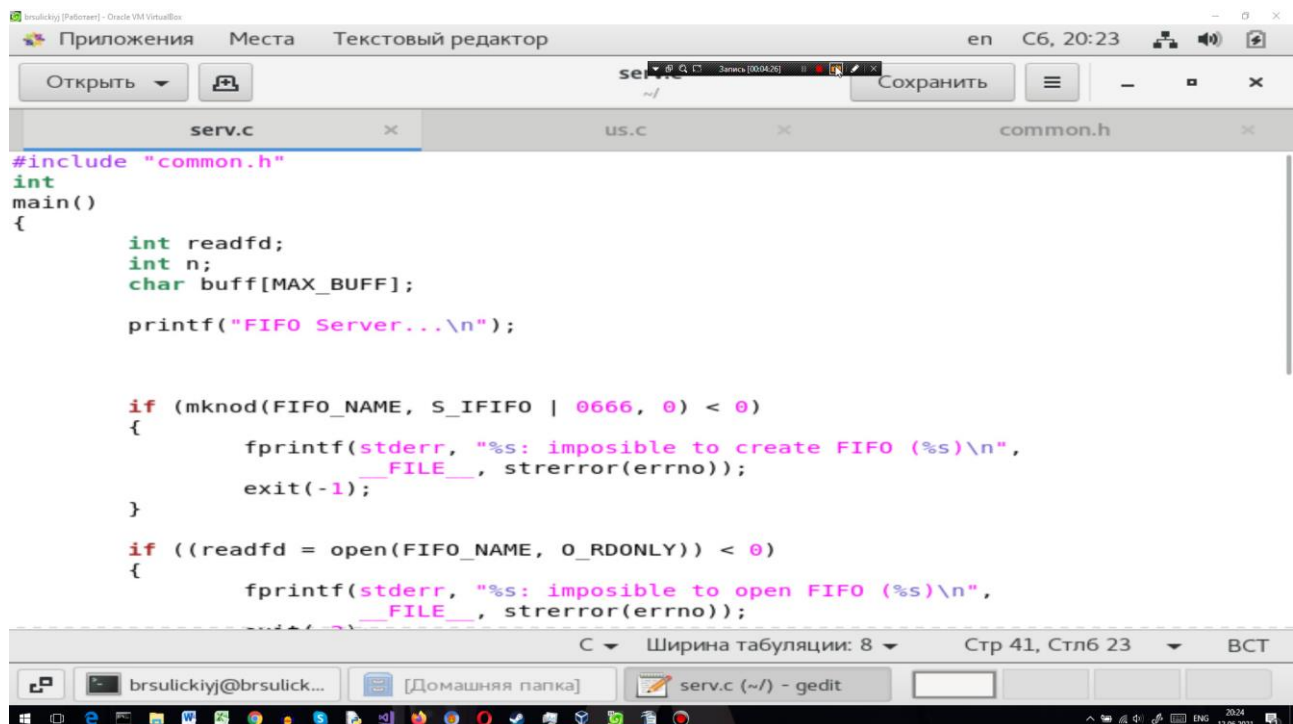
common.h:



The screenshot shows a text editor window titled "common.h" with the following code:

```
#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>
#define FIFO_NAME "/tmp/fifo"
#define MAX_BUFF 80
#endif
```

server.c:



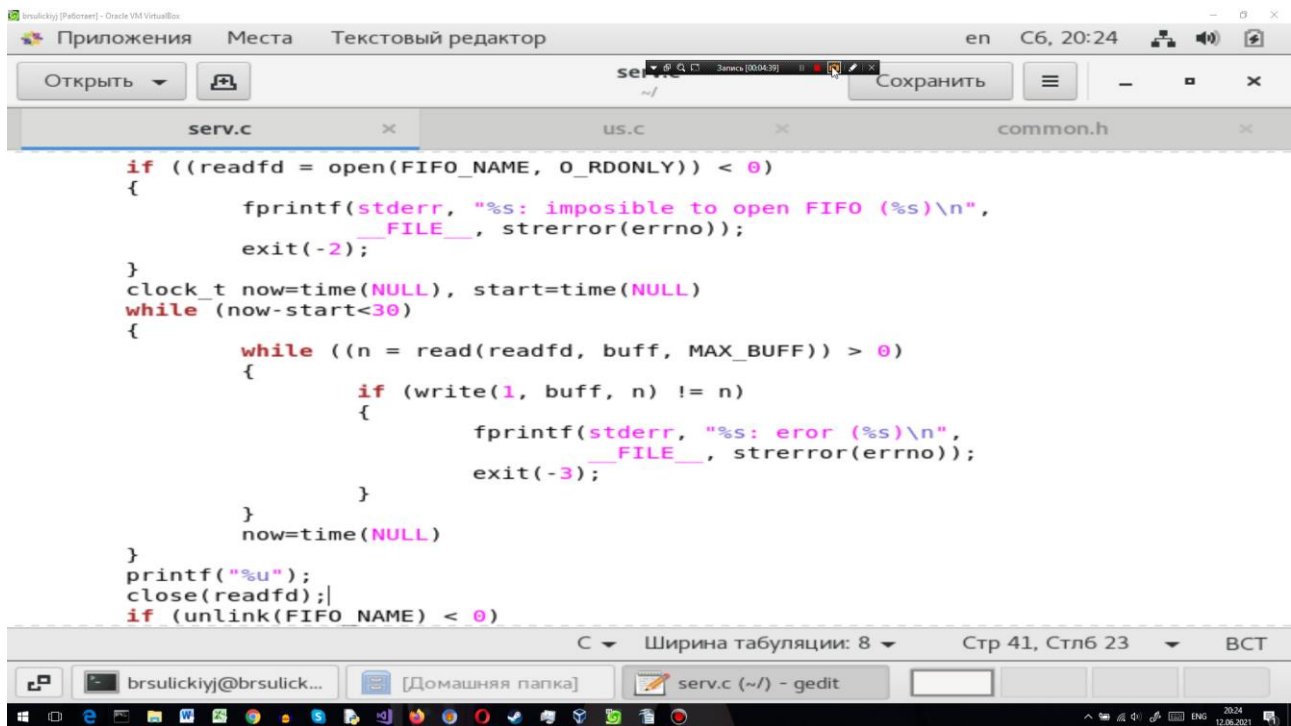
The screenshot shows a text editor window titled "serv.c" with the following code:

```
#include "common.h"
int
main()
{
    int readfd;
    int n;
    char buff[MAX_BUFF];

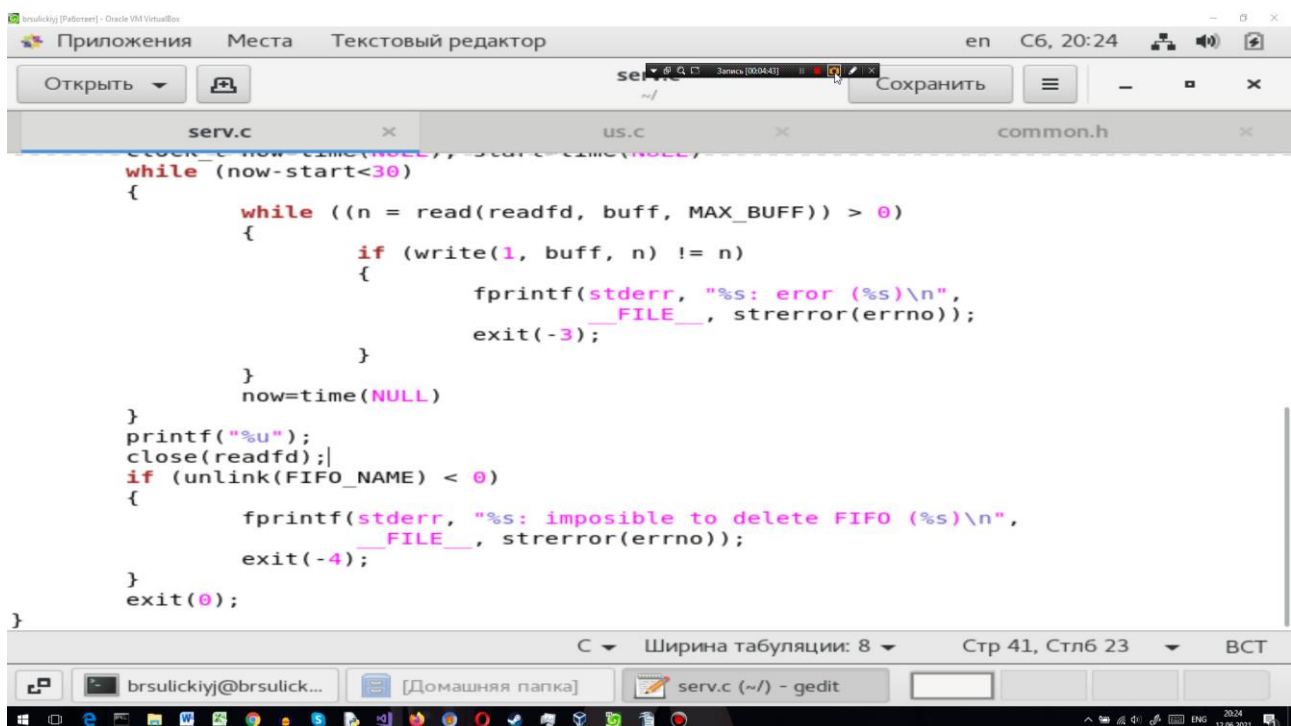
    printf("FIFO Server...\n");

    if (mkfifo(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
    {
        fprintf(stderr, "%s: impossible to create FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-1);
    }

    if ((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
    {
        fprintf(stderr, "%s: impossible to open FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-1);
    }
}
```



```
if ((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
{
    fprintf(stderr, "%s: impossible to open FIFO (%s)\n",
            __FILE__, strerror(errno));
    exit(-2);
}
clock_t now=time(NULL), start=time(NULL)
while (now-start<30)
{
    while ((n = read(readfd, buff, MAX_BUFF)) > 0)
    {
        if (write(1, buff, n) != n)
        {
            fprintf(stderr, "%s: error (%s)\n",
                    __FILE__, strerror(errno));
            exit(-3);
        }
    }
    now=time(NULL)
}
printf("%u");
close(readfd);
if (unlink(FIFO_NAME) < 0)
```



```
while (now-start<30)
{
    while ((n = read(readfd, buff, MAX_BUFF)) > 0)
    {
        if (write(1, buff, n) != n)
        {
            fprintf(stderr, "%s: error (%s)\n",
                    __FILE__, strerror(errno));
            exit(-3);
        }
    }
    now=time(NULL)
}
printf("%u");
close(readfd);
if (unlink(FIFO_NAME) < 0)
{
    fprintf(stderr, "%s: impossible to delete FIFO (%s)\n",
            __FILE__, strerror(errno));
    exit(-4);
}
exit(0);
}
```

client.c:

```
#include "common.h"
#define MESSAGE "Hello Server!!!\n"
int
main()
{
    int writefd;
    int msglen;
    char message[10];
    int i;
    long long int P;
    printf("FIFO Client...\n");
    for(i=0; i<=5; ++i;)
    {
        sleep(5);
        P = (long long int) time(0);
        sprintf(message, "%lli", P);
        message[9]='\n';
        printf("FIFO client... \n");
        if ((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
            fprintf(stderr, "%s: Îââîçîîæîî îðêðòðü FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }

        msglen = strlen(MESSAGE);
        if (write(writefd, MESSAGE, msglen) != msglen)
        {
            fprintf(stderr, "%s: Îðèáèà çàìèñè à FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }
    }
    close(writefd);
    exit(0);
}
```

makefile:

```
CC=gcc
CFLAGS=
programs=server client
all: $(programs)

server: server.c common.h
    $(CC) $(CFLAGS) $< -o $@

client: client.c common.h
    $(CC) $(CFLAGS) $< -o $@

clean:
    -rm $(programs) *.o

[]

U:--- makefile All L14 (GNUmakefile)
```

Вывод: приобрел практические навыки работы с сокетами.

Ответы на контрольные вопросы:

1. BSD является сокращением от 'Berkeley Software Distribution', названия, которое было выбрано Berkeley CSRG (Computer Systems Research Group) для их дистрибутива Unix.

2. Сокет (socket) - это конечная точка сетевых коммуникаций. Он является чем-то вроде "портала", через которое можно отправлять байты во внешний мир.

Приложение просто пишет данные в сокет. Программирование сокетов в Linux , их дальнейшая буферизация, отправка и транспортировка осуществляется используемым стеком протоколов и сетевой аппаратурой. Чтение данных из сокета происходит аналогичным образом. В программе сокет идентифицируется дескриптором - это просто переменная типа int. Программа получает дескриптор от операционной системы при создании сокета, а затем передаёт его сервисам socket API для указания сокета, над которым необходимо выполнить то или иное действие

3. Именованные каналы, описанные в главе 11, очень похожи на сокеты, но в способах их использования имеются значительные различия.

- Именованные каналы могут быть ориентированными на работу с сообщениями, что значительно упрощает программы.
- Именованные каналы требуют использования функций ReadFile и WriteFile, в то время как сокеты могут обращаться также к функциям send и recv.
- В отличие от именованных каналов сокеты настолько гибки, что предоставляют пользователям возможность выбрать протокол для использования с сокетом, например, TCP или UDP. Кроме того, пользователь имеет возможность выбирать

протокол на основании характера предоставляемой услуги или иных факторов.

- Сокеты основаны на промышленном стандарте, что обеспечивает их совместимость с системами, отличными от Windows.

Имеются также различия в моделях программирования сервера и клиента.

4. Коммуникационный домен определяет форматы адресов и правила их интерпретации. Внутри них существуют сокеты.

5. Виды сокетов:

- Сокеты в файловом пространстве имён (file namespace, сокеты Unix) используют в качестве адресов имена файлов специального типа.
- Сокеты в файловом пространстве имён похожи на именованные каналы тем, что для идентификации сокетов используются файлы специального типа. В мире сокетов есть и аналог неименованных каналов — парные сокеты.
- Сетевой сокет — сокет, в котором формат адреса имеет вид ip(7). Поскольку адрес транспортного уровня состоит из пары ip-адрес: порт, то и в структуре под адрес отводится два поля.

6. Когда поддержка BSD сокетов была добавлена в ядро Linux, разработчики решили добавить их единовременно все 17 (на сегодня 20) сокетных вызовов, и добавили для этих вызовов один дополнительный уровень косвенности. Для всей группы этих вызовов введен один новый, редко упоминаемый, системный вызов:

```
int socketcall( int call, unsigned long *args ),
```

где:

— call — численный номер сетевого вызова (SYS_CONNECT, SYS_ACCEPT...);

— args — указатель 6-ти элементного массива (блок параметров), в который последовательно упакованы все параметры любого из системных вызовов этой группы (сетевой), без различия их типа (приведенные к unsigned long)

7. Базовая эталонная модель взаимодействия открытых систем (БЭМВОС) — это концептуальная основа, определяющая характеристики и средства открытых систем.

Она обеспечивает работу в одной сети систем, выпускаемых различными производителями. Разработана ISO (международной организацией стандартов) и широко используется во всём мире как основа концепций информационных сетей и их ассоциаций. На базе этой модели описываются правила и процедуры передачи данных между открытыми системами. Она также описывает структуру открытой системы и

комплекс стандартов, которым она должна удовлетворять. Основными элементами модели являются: уровни, объекты, соединения, физические средства соединений.

Модель информационной системы состоит из трёх основных составляющих:

- прикладные процессы (осуществляют обработку данных);
- область взаимодействия (размещаемые в ней блоки прокладывают в сети логические каналы (пунктирная линия на рисунке) между портами прикладных процессов и обеспечивает их взаимодействие);
- физические средства соединений (обеспечивают физическую связь систем).