

Лабораторная работа №14

Дисциплина - операционные системы

Пронякова О.М.

9 мая 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Пронякова Ольга Максимовна
- студент НКАбд-02-22
- факультет физико-математических и естественных наук
- Российский университет дружбы народов

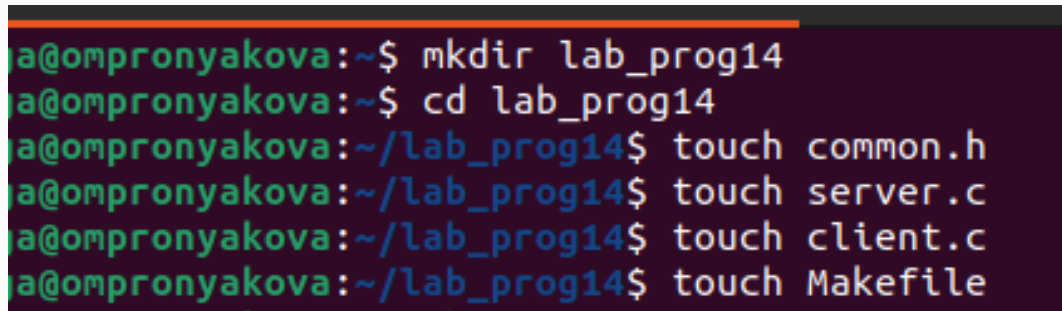
Создание презентации

Приобретение практических навыков работы с именованными каналами.

Изучите приведённые в тексте программы `server.c` и `client.c`. Взяв данные примеры за образец, напишите аналогичные программы, внося 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 лабораторной работы

В терминале создаю каталог и соответствующие файлы, которые заполняю определенными данными(рис.1).

A terminal window with a dark purple background and a green title bar. The prompt is 'a@ompronyakova:~\$'. The commands and their outputs are: 'mkdir lab_prog14', 'cd lab_prog14', 'touch common.h', 'touch server.c', 'touch client.c', and 'touch Makefile'.

```
a@ompronyakova:~$ mkdir lab_prog14
a@ompronyakova:~$ cd lab_prog14
a@ompronyakova:~/lab_prog14$ touch common.h
a@ompronyakova:~/lab_prog14$ touch server.c
a@ompronyakova:~/lab_prog14$ touch client.c
a@ompronyakova:~/lab_prog14$ touch Makefile
```

Рис. 1: Создание подкаталога и файлов в нем

Пишу аналогичные программы, внося некоторые изменения - работает не 1 клиент, а несколько, клиенты передают текущее время с некоторой периодичностью(использую функцию `sleep()` для приостановки работы клиента), сервер работает не бесконечно, а прекращает работу через некоторое время(используйте функцию `clock()` для определения времени работы сервера)(рис. 2) (рис. 3).

Выполнение лабораторной работы

```
client.c  x      common.h  x      Makefile  x
1
2 #include "common.h"
3
4 #define MESSAGE "Hello Server!!!\n"
5
6 int
7 main()
8 {
9     int writefd; /* дескриптор для записи в FIFO */
10    int msglen;
11
12    for(int i=0; i<20; i++)
13    {
14        sleep(3);
15        t=time(NULL);
16
17        printf("FIFO Client...\n");
18
19        /* получим доступ к FIFO */
20        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
21        {
22            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
23                __FILE__, strerror(errno));
24            exit(-1);
25        }
26
27        /* передадим сообщение серверу */
28        msglen = strlen(MESSAGE);
29        if(write(writefd, MESSAGE, msglen) != msglen)
30        {
31            fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
32                __FILE__, strerror(errno));
```

Выполнение лабораторной работы

```
-
2
3 #include "common.h"
4
5 int
6 main()
7 {
8     int readfd; /* дескриптор для чтения из FIFO */
9     int n;
10    char buff[MAX_BUFF]; /* буфер для чтения данных из FIFO */
11
12    /* баннер */
13    printf("FIFO Server...\n");
14
15    /* создаем файл FIFO с открытыми для всех
16     * правами доступа на чтение и запись
17     */
18    if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
19    {
20        fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
21            __FILE__, strerror(errno));
22        exit(-1);
23    }
24
25    /* откроем FIFO на чтение */
26    if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
27    {
28        fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
29            __FILE__, strerror(errno));
30        exit(-2);
31    }
32
33    /* читаем данные из FIFO и выводим на экран */
34    ccclock_t now=time(NULL), start=time(NULL);
35    while(1)
36    {
37        n = read(readfd, buff, MAX_BUFF);
38        if(n < 0)
39            break;
40        printf("%s", buff);
41    }
42}
```

Приобрела практические навыки работы с именованными каналами.