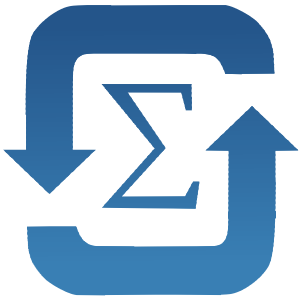
Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра теоретической и прикладной информатики

Управление ресурсами в вычислительных системах

Лабораторная работа № 3



ФАКУЛЬТЕТ:	ПМИ
ГРУППА:	ПМ-71
СТУДЕНТЫ:	Баштовой П.А. Востриков В.А.
ВАРИАНТ:	4
ПРЕПОДАВАТЕЛЬ:	Стасышин В.М, Сивак М.А.

Новосибирск 2020

1) ***Цель работы***

Ознакомиться с системой управления вводом-выводом в ОС UNIX и основными структурами данных, используемыми этой системой. Исследовать механизм работы системы управления вводом-выводом.

2) ***Условие задачи:***

Пусть N процессов осуществляют доступ к одному и тому же файлу на диске (но с разными режимами доступа). Разработать программу, демонстрирующую динамику формирования таблицы файлов и изменения ее элементов (при перемещении указателей чтения-записи, например). Сценарий программы может быть следующим:

– открытие файла процессом 0 для чтения;

– открытие файла процессом 1 для записи;

– открытие файла процессом 2 для добавления;

– чтение указанного числа байт файла процессом 0;

– запись указанного числа байт в файл процессом 1;

– добавление указанного числа байт в файл процессом 2.

После каждого из этапов печатаются таблицы файлов всех процессов.

3) ***Анализ задачи:***

a. Считываем имя файла, с которым будем работать.
b. В случае успеха считывания информации о файле запускаем процессы.
c. Распечатываем таблицы.
d. Взаимодействуем с этими процессами.
e. Распечатываем таблицы

4) ***Программные компоненты:***

int open(pathname, oflag [[, pmode]]) - функция open открывает файл, определяемый по path-имени, и подготавливает его к последующему чтению или записи, что определяется посредством oflag;

*int read(int handle, void *buf, unsigned len)* - функция read делает попытку считать len байт из файла, связанного с handle, в буфер, адресуемый параметром buf;

*int write(int fd, void *buffer, unsigned length)* - функция записывает length байтов из буфера buffer в файл, определенный дескриптором файла fd;

*int stat(char *path, struct stat *buf)* – Функция stat берет информацию о файле или каталоге, определенном параметром path, и помещает ее в структуру, на которую указывает buf.

5) ***Спецификация программы:***

Программа лежит в **/home/NSTU/pmi-b7104/lab_3/**

Сборка осуществлялась командой **gcc -std=c99 -Wall -o lab3 lab3.c**

Программа lab3 принимает на вход один параметр: имя открываемого файла. Если параметр не был передан, в этом случае программа опрашивает пользователя о файле, с которым она будет работать. В случае неверного ввода программа выдаст ошибку.

Дальше программа попытается считать информацию о файле. В случае успеха появится меню с выбором в каком режиме открывать файл, в случае неудачи - вывод ошибки. Максимальное количество процессов, которое может быть открыто программой, задается в MAX_COUNT_PROCESS. После открытия файла будут распечатаны таблицы файлов для каждого процесса. После этого идет опрос: «Хотите ли вы провзаимодействовать?». В случае положительного ответа программа опрашивает сколько символов использовать (считать в буффер, записать из буффера, добавить из буфера). После чего распечатываются таблицы файлов.

6) ***Текст программы:***

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <errno.h>

#define SIZE_FNAME 30 // Максимальный размер имени файла
#define SIZE_BUFF 512 // максимальный размер буффера
#define MAX_COUNT_PROCESS 10 // максимальное количество процессов


struct table // таблица
{
	char file_name[SIZE_FNAME];
	int inode; // индексный дескриптор
	int local_point; // указатель
} f_tables[MAX_COUNT_PROCESS];

struct stat st; // stat предоставляет информацию по имени файла
int f_descr[MAX_COUNT_PROCESS]; // дескрипторы файла
char f_name[SIZE_FNAME]; // имя файла


void write_into_table(int i, char file[SIZE_FNAME], int point) // заполнение таблицы файла
{
	strcpy(f_tables[i].file_name, file);
	f_tables[i].inode = st.st_ino;
```

```
	f_tables[i].local_point = point;
}

void print_table(int i) // распечатывание таблицы
{
	printf("\nfile name \t\t number inode \t\t local point\n");
	printf("%s \t\t %d \t\t %d\n\n", f_tables[i].file_name, f_tables[i].inode,
f_tables[i].local_point);
}

void change_lpoint(int i, int point) // изменение указателя
{
	f_tables[i].local_point += point;
}

int func_actions(int i, int choice)
{
	char buff[SIZE_BUFF];
	int val, counts;

	// в зависимости от процесса, срабатывает соответствующее действие
	switch (choice)
	{
	case 1:
		// Читаем в буффер
		printf("How many characters do you want to read to buffer? ");
		if (!scanf("%d", &counts))
			return -1;

		printf("The process read %d characters!\n", counts);
		val = read(f_descr[i], buff, counts * sizeof(char));
		break;
	case 2:
		// Записываем из буффера
		printf("How many characters do you want to write from buffer? ");
		if (!scanf("%d", &counts))
			return -1;

		printf("The process write %d characters!\n", counts);
		val = write(f_descr[i], buff, counts * sizeof(char));
		break;
	case 3:
		// Записываем из буфера в конец
		printf("How many characters do you want to add from buffer? ");
		if (!scanf("%d", &counts))
			return -1;

		printf("The process adds %d characters!\n", counts);
		val = write(f_descr[i], buff, counts * sizeof(char));
		break;
	default:
		break;
	}

	change_lpoint(i, val); // изменяем положение указателя
	return 1;
}


int main(int argc, char* argv[])
{
	// Получаем имя файла из переданного аргумента
	if (argc == 1) {
		printf("Enter the name file: ");
		if (!scanf("%s", f_name)) {
```

```
                perror("Failed to read file name!\n");
                return -1;
            }
        }
        else { // Получаем имя файла из stdin
            strcpy(f_name, argv[1]);
            if (f_name[0] == ' ') {
                perror("Failed to read file name!\n");
                return -1;
            }
        }

        // Пытаемся считать информацию о файле
        if (stat(f_name, &st) != -1)
        {
            int choice = 4, choice_2;
            int flag = 1;

            // открываем процессы (до MAX_COUNT_PROCESS)
            for (int i = 0; i < MAX_COUNT_PROCESS; i++)
            {
                printf("Open for:\n1 - read only,\n2 - write only,\n3 - append,\n4 -
Exit\n");

                flag = 1;

                // Если не удалось считать выбор пользователя
                if (!scanf("%d", &choice)) {
                    perror("Input Error "); // вывод ошибок
                    return -2;
                }

                switch (choice)
                {
                case 1:
                    f_descr[i] = open(f_name, O_RDONLY); // открываем в режиме чтения
                    break;
                case 2:
                    f_descr[i] = open(f_name, O_WRONLY); // открываем в режиме записи
                    break;
                case 3:
                    f_descr[i] = open(f_name, O_WRONLY | O_APPEND); // открываем в
режиме добавления
                    break;
                case 4:
                    return 0;
                    break;
                default:
                    printf("Try again!\n");
                    flag = 0;
                }

                if (flag != 1) // в случае неверного выбора
                {
                    i--;
                    continue;
                }

                // пытаемся получить информацию об открытом файле
                if (fstat(f_descr[i], &st) != -1)
                {
                    printf("The process opens %d the file.\n", i);
                    write_into_table(i, f_name, 0);  // записываем информацию в
таблицу
```

```
                        for (int j = 0; j <= i; j++) // вывод таблиц перед действием
процесса
                                print_table(j);

                        printf("Do you want to make some action with this process?\n1 -
Yes, 2 - No\n");
                        if (!scanf("%d", &choice_2)) {
                                perror("Input Error "); // вывод ошибок
                                return -2;
                        }

                        if (choice_2 == 1)
                        {
                                if (func_actions(i, choice) == -1) // взаимодействуем с
процессом
                                        return -1;

                                for (int j = 0; j <= i; j++) // вывод таблиц после
действием процесса
                                        print_table(j);
                        }
                }
                else
                {
                        perror("Failed to return information about the open file! "); //
вывод ошибок
                        return -1;
                }
        }
    }
    else
    {
        perror("Failed to read file information! "); // вывод ошибок
        return -3;
    }
}
```

7) ***Тест программы:***

1) **Передача программе несуществующего имени файла.**

```
[pmi-b7104@students lab_3]$ ./lab3 text
Failed to read file information! : No such file or directory
[pmi-b7104@students lab_3]$
```

2) **Ввод несуществующего имени файла в программе.**

```
[pmi-b7104@students lab_3]$ ./lab3
Enter the name file: text
Failed to read file information! : No such file or directory
[pmi-b7104@students lab_3]$
```

3) **Корректная работа программы (создаем процесс «readonly», читаем 15 символов, создаем процесс «writeonly», записываем 12 символов, и в режиме «добавления» добавляем 10 символов в конец).**

Содержание Text.txt до операции:

«Паника глобальная с коронавирусом... — злится муж. — И не говори, милый! А давай на всякий случай закупимся макаронами и тушенкой? — На какой такой случай, дорогая? — На случай, если нам захочется макарон с тушенкой...»

```
[pmi-b7104@students lab_3]$ ./lab3
Enter the name file: text.txt
Open for:
1 - read only,
2 - write only,
3 - append,
4 - Exit
1
The process opens 0 the file.

file name               number inode            local point
text.txt                1709460                 0

Do you want to make some action with this process?
1 - Yes, 2 - No
1
How many characters do you want to read to buffer? 15
The process read 15 characters!

file name               number inode            local point
text.txt                1709460                 15

Open for:
1 - read only,
2 - write only,
3 - append,
4 - Exit
2
The process opens 1 the file.

file name               number inode            local point
text.txt                1709460                 15


file name               number inode            local point
text.txt                1709460                 0

Do you want to make some action with this process?
1 - Yes, 2 - No
1
How many characters do you want to write from buffer? 12
The process write 12 characters!

file name               number inode            local point
text.txt                1709460                 15


file name               number inode            local point
text.txt                1709460                 12

Open for:
1 - read only,
2 - write only,
3 - append,
4 - Exit
3
The process opens 2 the file.
```

```
file name               number inode          local point
text.txt                1709460               15


file name               number inode          local point
text.txt                1709460               12


file name               number inode          local point
text.txt                1709460               0

Do you want to make some action with this process?
1 - Yes, 2 - No
1
How many characters do you want to add from buffer? 10
The process adds 10 characters!

file name               number inode          local point
text.txt                1709460               15


file name               number inode          local point
text.txt                1709460               12


file name               number inode          local point
text.txt                1709460               10

Open for:
1 - read only,
2 - write only,
3 - append,
4 - Exit
4
[pmi-b7104@students lab_3]$
```

Содержание Text.txt после операции:

*«Паника глобальная с коронавирусом... — злится муж. — И не говори, милый! А давай на всякий случай закупимся макаронами и тушенкой? — На какой такой случай, дорогая? — На случай, если нам захочется макарон с тушенкой...**Паника гло**»*

4) Попытка открыть файл, к которому нет доступа.

Изменяем доступ:

```
[pmi-b7104@students lab_3]$ chmod 000 text.txt
[pmi-b7104@students lab_3]$ ls -l
total 28
-rwxr-xr-x. 1 pmi-b7104 пользователи домена 13467 Mar 19 14:19 lab3
-rw-r--r--. 1 pmi-b7104 пользователи домена  4413 Mar 19 14:19 lab3.c
----------. 1 pmi-b7104 пользователи домена   228 Mar 19 15:21 text.txt
```

```
[pmi-b7104@students lab_3]$ ./lab3 text.txt
Open for:
1 - read only,
2 - write only,
3 - append,
4 - Exit
1
Failed to return information about the open file! : Bad file descriptor
[pmi-b7104@students lab_3]$
```