МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

Факультет информационных технологий и компьютерных систем

Кафедра «Прикладная математика и фундаментальная информатика»

**Домашнее задание**

|  |  |  |
| --- | --- | --- |
| по дисциплине | Операционные системы |  |

|  |  |
| --- | --- |
| Студента | Иванютин Сергей Александрович |
|  | фамилия, имя, отчество полностью |
| Курс | 2 Группа ФИТ-221 |
| Направление | 02.03.02 Фундаментальная информатика |
|  | и информационные технологии |
|  | код, наименование |
| Руководитель | ассистент |
|  | должность, ученая степень, звание |
|  | Карабцов Р.Д. |
|  | фамилия, инициалы |
| Выполнил | 04.06.2024 |
|  | дата, подпись студента |
| баллы |  |
|  | дата, подпись руководителя |

Омск-2024

**Задание**

Разработать программу для Linux, которая должна запускаться в двух экземплярах, каждый со своей виртуальной консоли. Программа использует заранее подготовленный текстовый файл. Она открывает этот файл, при невозможности этого действия выдается сообщение и прекращает выполнение. После успешного открытия файла делается попытка установить на весь файл многопользовательскую блокировку по записи. По результатам попытки выполнения блокировки – на экран выдается сообщение о его реализации или текущей невозможности это сделать. При невозможности установить блокировку сразу, программа задает блокировку с ожиданием ее выполнения. По установлении блокировки доступа программа читает из этого файла все находящиеся в нем данные и выводит их на экран. Затем программа делает задержку выполнения («засыпает») на 7 или 8 секунд, после чего снимает блокировку. Сообщения должны выводиться цветные и в середине экрана. (Базовый вариант)

1. Чем блокировка файла по записи в Windows отличается от блокировка по записи в Linux.

2. Чем блокировка файла по чтению в Windows отличается от блокировка по чтению в Linux.

3. Сформулируйте наиболее существенное отличие управлением цветом вывода на консоль в Windows и Linux, оцените в каких условиях какое из этих решений кажется вам предпочтительней.

**Решение**

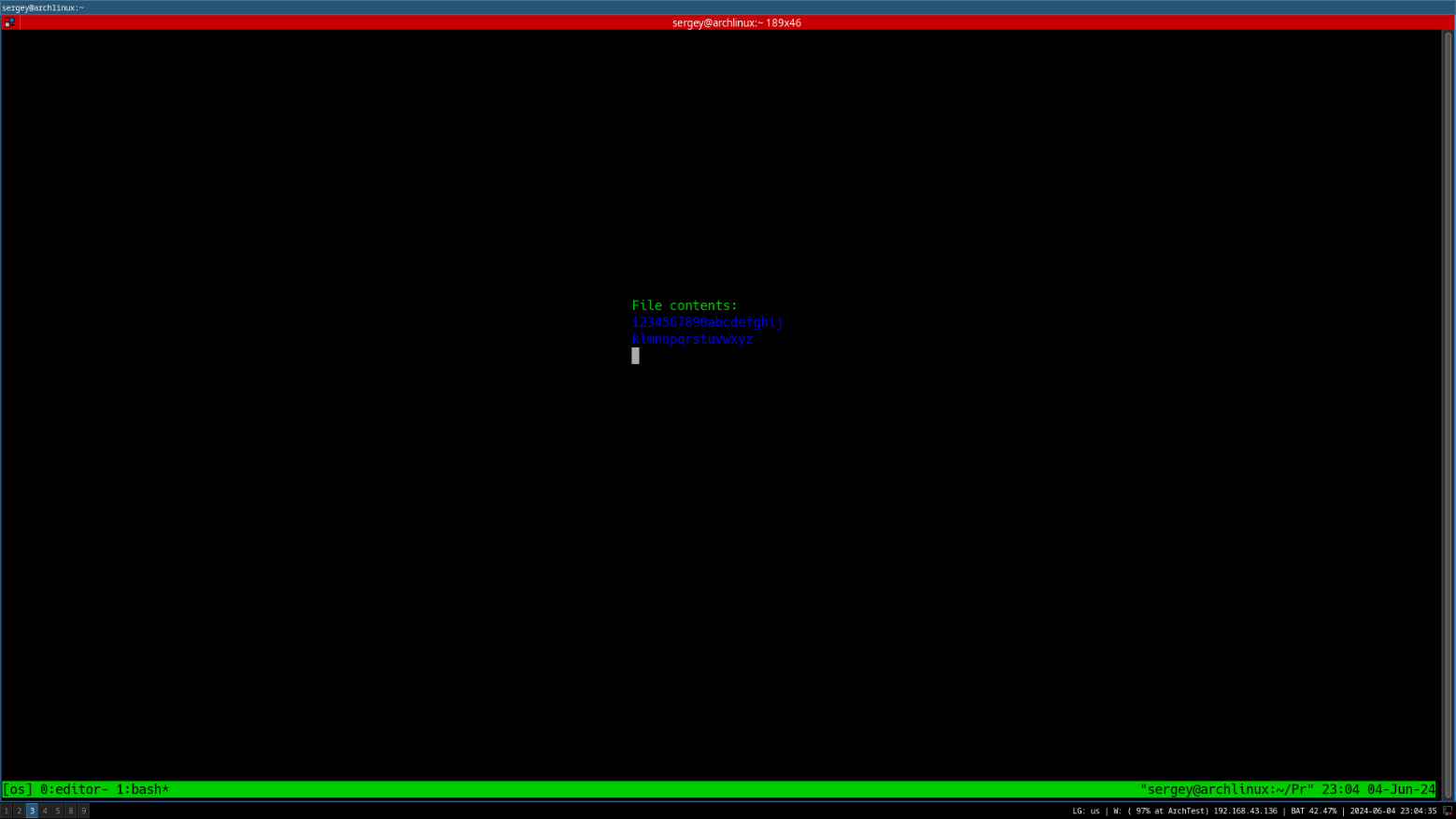
****

Рисунок 1. Успешное выполнение программы



Рисунок 2. Блокировка и ожидание выполнения программы

1. Блокировка файла по записи в Windows и Linux отличается в основном в реализации и подходе к работе с файлами. В Windows для блокировки файла по записи обычно используется функция LockFile, которая позволяет заблокировать определенный диапазон байтов в файле. В Linux же для блокировки файла по записи используется системный вызов fcntl с аргументом F\_SETLK, который позволяет устанавливать блокировку на файл целиком или на определенный диапазон байтов.

2. Блокировка файла по чтению в Windows и Linux также имеет различия в реализации. В Windows для блокировки файла по чтению используется функция LockFileEx, которая позволяет установить блокировку только на чтение. В Linux же аналогичную функциональность можно достичь с помощью системного вызова fcntl с аргументом F\_SETLK и указанием типа блокировки как F\_RDLCK.

3. Отличие управления цветом вывода на консоль в Windows и Linux заключается в использовании различных механизмов и кодов для установки цвета текста и фона. Например, в Windows для управления цветом консоли используются специальные функции из библиотеки Windows API, такие как SetConsoleTextAttribute. В Linux же для управления цветом вывода на консоль можно использовать ANSI Escape-коды, которые добавляются к тексту для изменения его цвета.

В зависимости от конкретной задачи и целевой платформы можно определить, какое из решений предпочтительнее. Например, если необходимо написать кроссплатформенное приложение, то использование ANSI Escape-кодов для управления цветом в Linux может быть более предпочтительным, так как они работают и в Windows (хотя не всегда корректно). В случае, если приложение предназначено только для одной платформы, то использование специфических функций API может быть более удобным и эффективным.

**Листинг программы**

#include <errno.h>

#include <sys/ioctl.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

#include <stdio.h>

#define RED "\033[31m"

#define GREEN "\033[32m"

#define BLUE "\033[34m"

#define CLEAR "\033[2J"

#define RESET "\033[0;0H"

void display(int h, char \*text, int len, char \*color);

int main()

{

int fd, td, len;

char buffer[300];

struct flock check\_lock, lock = {F\_WRLCK, SEEK\_SET, 0, 0};

fd = open("file", O\_RDWR);

td = open("/dev/tty", O\_WRONLY);

write(td, CLEAR, strlen(CLEAR));

if (td < 0)

{

display(td, "Cannot open terminal file", 25, RED);

return 1;

}

if (fd < 0)

{

switch (errno)

{

case ENOENT:

display(td, "File doesnt exist", 17, RED);

break;

default:

display(td, "Error", 5, RED);

break;

}

}

if (fd > 0)

{

fcntl(fd, F\_GETLK, &check\_lock);

if (check\_lock.l\_type != F\_UNLCK)

{

display(td, "File is locked", 14, RED);

}

fcntl(fd, F\_SETLKW, &lock);

len = read(fd, buffer, 300);

display(td, "File contents:", 14, GREEN);

display(td, buffer, len, BLUE);

sleep(7);

lock.l\_type = F\_UNLCK;

fcntl(fd, F\_SETLK, &lock);

close(fd);

}

write(td, CLEAR, strlen(CLEAR));

write(td, RESET, strlen(RESET));

close(td);

}

static int row, col;

void display(int fd, char \*text, int len, char \*color)

{

int td, voffset, hoffset, row\_len, actlen;

char buffer[300];

struct winsize ws;

voffset = 5;

hoffset = 10;

if (row + col == 0)

{

td = open("/dev/tty", O\_RDONLY);

if (td < 0)

{

printf("Cannot open tty\n");

fflush(stdout);

}

ioctl(td, TIOCGWINSZ, &ws);

row = ws.ws\_row/2-voffset;

col = ws.ws\_col/2-hoffset;

}

row\_len = 2\*hoffset;

sprintf(buffer, "\033[%d;%dH", row, col);

write(fd, buffer, strlen(buffer));

write(fd, color, strlen(color));

while (len > 0)

{

if (len < row\_len)

{

actlen = len;

}

else

{

actlen = row\_len;

}

len -= row\_len;

write(fd, text, actlen);

text += actlen;

row++;

sprintf(buffer, "\033[%d;%dH", row, col);

write(fd, buffer, strlen(buffer));

}

close(td);

}