Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных систем и технологий»

**Организация FS в UNIX. Работа с файлами. Понятие о memory mapped files.**

Лабораторная работа №8 по учебной дисциплине «Операционные системы»

По направлению 09.03.01 «Информатика и вычислительная техника»

Отчёт

Принял преподаватель:

\_\_\_\_\_\_\_\_\_ Ананьев В. В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Выполнил студент группы 8091:

\_\_\_\_\_\_\_ Шаклеин В. В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Великий Новгород

2020

**Цель работы:** познакомиться с механизмами работы с файлами в UNIX.

**Задание:**

Написать программу для работы с файлами, проецируемыми в память.

В параметрах командной строки программа должна получить имена двух файлов.

Первый файл должен содержать входные данные - короткий текст на английском языке. Достаточно нескольких предложений. (Например, их можно скопировать со страницы справки man mmap, или с любой страницы из интернета.)

Второй файл (выходной) требуется открыть, если надо, то создать, и спроецировать в память текущего процесса.

Программа должна прочитать входной файл, размещая получаемые данные в памяти, в которую спроецирован выходной файл.

В конце работы требуется освободить все выделенные ресурсы и закрыть оба файла.

В отчете привести исходный код программы, а также содержимое и размеры обоих файлов после окончания работы программы.

Примечание.

Для того, чтобы создать файл достаточного размера требуется сначала узнать размер входного файла с текстом. Если файл создается заведомо большего размера, без проверки требуемого количества памяти, то количество баллов за лабораторную снижается на 1.

**Содержание файла msg\_queue.c:**

#include <stdio.h>

#include <sys/mman.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <unistd.h>

int main(int argc, char \*argv[])

{

const char \*input\_file\_name = argv[1];

const char \*output\_file\_name = argv[2];

int input\_file = open(input\_file\_name, O\_RDONLY, 0600);

int output\_file = open(output\_file\_name, O\_RDWR | O\_CREAT, 0600);

struct stat st;

stat(input\_file\_name, &st);

int file\_size = st.st\_size;

ftruncate(output\_file, file\_size);

char \*output\_file\_data = (char \*)mmap(NULL, file\_size, PROT\_WRITE | PROT\_READ, MAP\_SHARED, output\_file, 0);

read(input\_file, output\_file\_data, file\_size);

munmap(output\_file\_data, file\_size);

close(input\_file);

close(output\_file);

}

**Содержание файла test\_input\_file.dat:**

mmap() creates a new mapping in the virtual address space of the calling process. The starting address for

the new mapping is specified in addr. The length argument specifies the length of the mapping (which must be

greater than 0).

**Содержание файла test\_output\_file.dat:**

mmap() creates a new mapping in the virtual address space of the calling process. The starting address for

the new mapping is specified in addr. The length argument specifies the length of the mapping (which must be

greater than 0).

**Результат, выведенный на экран:**

sevskii@MBP-sevskii 8 % make

gcc mem\_mapping.c -o mem\_mapping

./mem\_mapping test\_input\_file.dat test\_output\_file.dat

**Вывод:** В процессе выполнения лабораторной работы я на практике познакомился с работой с файлами в UNIX.