Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования "Новгородский государственный университет имени Ярослава Мудрого"

Кафедра «Информационных технологий и систем»

Дисциплина «Операционные системы»

Отчет по лабораторной работе

«Огранизация FS в UNIX. Работа с файлами. Понятие о memory mapped files»

Выполнил студент группы 9091

\_\_\_\_\_\_\_\_\_\_\_\_\_/Федотов Дмитрий Борисович/

Подпись ФИО

Принял преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_/Ананьев Владислав Валерьевич/

Подпись ФИО

Великий Новгород

2021

**Цель лабораторной работы**

Цель работы: познакомиться с огранизацией FS в UNIX и понятием memory mapped files, научиться работать с файлами.

**Исходный текст программы**

|  |
| --- |
| main.c |
| #include "stdio.h"  #include "fcntl.h"  #include "stdlib.h"  #include "unistd.h"  #include "sys/stat.h"  #include "sys/mman.h"  #include "sys/types.h"  int main(int argv, char\* argc[])  {  if (argv <= 2)  {  printf("Error! Not enough arguments! Required (2): input file, output file\n");  return -1;  }  const char\* input\_file\_name = argc[1];  const char\* output\_file\_name = argc[2];  int input\_file = open(input\_file\_name, O\_RDONLY, 0600);  if (input\_file < 0)  {  perror("Error! Cant open input file!\n");  return -1;  }  struct stat input\_file\_stat;  if (fstat(input\_file, &input\_file\_stat) == -1)  {  perror("Error! Cant get input file stat!\n");  return -1;  }  int output\_file = open(output\_file\_name, O\_RDWR | O\_CREAT, 0600);  if (output\_file < 0)  {  perror("Error! Cant get output file disc!\n");  return -1;  }  int file\_length = input\_file\_stat.st\_size;  ftruncate(output\_file, file\_length);  char\* output\_file\_ptr = (char\*) mmap(NULL, file\_length, PROT\_WRITE | PROT\_READ, MAP\_SHARED, output\_file, 0);  if (output\_file\_ptr == MAP\_FAILED)  {  perror("Output file mapping failed!\n");  return -1;  }  ssize\_t input\_file\_read\_bytes = read(input\_file, output\_file\_ptr, file\_length);  printf("Need to write: %i\n", file\_length);  printf("Write in file: %i\n", (int) input\_file\_read\_bytes);  close(input\_file);  close(output\_file);  munmap((void\*) output\_file\_ptr, file\_length);  return 0;  } |

|  |
| --- |
| Содержимое входного файла input\_file |
| ipcs shows information on System V inter-process communication facilities. By default it  shows information about all three resources: shared memory segments, message queues, and  semaphore arrays. |

**Результат выполнения программы**

|  |
| --- |
| Вывод в терминале |
| >> make  gcc main.c -o main  ./main input\_file output\_file  Need to write: 198  Write in file: 198 |

|  |
| --- |
| Содержимое выходного файла output\_file |
| ipcs shows information on System V inter-process communication facilities. By default it  shows information about all three resources: shared memory segments, message queues, and  semaphore arrays. |

Размер входного файла: 198 байт

Размер выходного файла: 198 байт

**Вывод**

Вывод: выполняя лабораторную работу, я познакомился с огранизацией FS в UNIX и понятием memory mapped files, научился работать с файлами.