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

«Организация взаимодействия процессов через pipe и FIFO в UNIX»

Выполнил студент группы 9091

\_\_\_\_\_\_\_\_\_\_\_\_\_/Прокофьев Максим Юрьевич/

Подпись ФИО

Принял преподаватель

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

Подпись ФИО

Великий Новгород

2021

**Цель лабораторной работы**

Цель работы: познакомиться с механизмами организации взаимодействия процессов через pipe и FIFO в UNIX.

**Исходный текст программы**

|  |
| --- |
| Lab05.c |
| #include <stdio.h>  #include <sys/stat.h>  #include <time.h>  #include <stdlib.h>  #include <unistd.h>  #include <fcntl.h>  const char\* NAME = "stroka";  int comparator(const void\* a, const void\* b)  {  return (\*((int\*)b) - \*((int\*)a));  }  int\* randomazing(int N)  {  srand((unsigned) (time(0)));  int \*array = malloc(sizeof(int) \* N);  for (int i = 0; i < N; ++i)  {  array[i] = rand() % 300;  }    return array;  }  void Out\_numbers(int\* array, int N)  {  for (int i = 0; i < N; ++i)  {  printf("%d ", array[i]);  }  printf("\n");  }  int main(int argv, char \*argc[])  {  int N = atoi(argc[1]);  int \*array = randomazing(N);  Out\_numbers(array, N);  mknod(NAME, S\_IFIFO | 0666, 0);  int p[2];  pipe(p);  int childId = fork();  if (childId == 0)  {  close(p[1]);  int fifo = open(NAME, O\_WRONLY);  int \*mem = malloc(sizeof(int) \* N);  read(p[0], mem, sizeof(int) \* N);  close(p[0]);  qsort(mem, N, sizeof(int), comparator);  write(fifo, mem, sizeof(int) \* N);  }  else if (childId > 0)  {  close(p[0]);  int fifo = open(NAME, O\_RDONLY);  write(p[1], array, sizeof(int) \* N);  close(p[1]);  int \*sortirovka = malloc(sizeof(int) \* N);  read(fifo, sortirovka, sizeof(int) \* N);  Out\_numbers(sortirovka, N);  unlink(NAME);  }  else if (childId < 0)  {  perror("error with fork()\n");  }  } |

**Результат выполнения программы**

|  |
| --- |
| Вывод в терминале |
| >> make  gcc lab05.c -o lab05  ./lab05 10  199 266 221 116 0 6 160 190 99 45  266 221 199 190 116 99 45 6 0 |
|  |

**Вывод**

Вывод: выполняя лабораторную работу, я познакомился с механизмами организации взаимодействия процессов через pipe и FIFO в UNIX.