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»

**Процессы в ОС UNIX**

Лабораторная работа №4 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:

\_\_\_\_\_\_\_ Ефимов А.С.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Великий Новгород

2020

**Цель работы:** познакомиться с механизмами разделённой памяти и многопоточности.

**Задание:**

Объединить программы первой части предыдущей лабораторной в одну программу.

Вначале программа должна сгенерировать 20 случайных чисел, разместить их в разделяемой памяти и вывести на экран.

Затем требуется породить новый процесс, который выполнит сортировку самостоятельно, без запуска других программ (в отличие от предыдущей лабораторной).

Второй процесс должен отсортировать числа и также вывести их на экран.

Первый процесс должен дождаться завершения работы второго, после чего самостоятельно выполнить освобождение выделенной разделяемой памяти и всех прочих выделенных ресурсов (при наличии таковых).

**Содержание файла**

#include <stdio.h>

#include <sys/wait.h>

#include <sys/shm.h>

#include <stdlib.h>

#include <unistd.h>

#include <time.h>

const int AMOUNT = 20;

int compare\_ints(const void \*a, const void \*b)

{

return (\*((int \*)a) - \*((int \*)b));

}

void sortAndPrint(int \*mem)

{

qsort(mem, AMOUNT, sizeof(int), compare\_ints);

for (size\_t i = 0; i < AMOUNT; i++)

{

printf("%i ", mem[i]);

printf("\n");

}

}

int main()

{

srand((unsigned)(time(0)));

int memId = shmget(IPC\_PRIVATE, sizeof(int) \* AMOUNT, 0600 | IPC\_CREAT | IPC\_EXCL);

int \*numbers = (int \*)shmat(memId, 0, 0);

for (size\_t i = 0; i < AMOUNT; i++)

{

numbers[i] = rand() % 1000;

}

for (size\_t i = 0; i < AMOUNT; i++)

{

printf("%d ", numbers[i]);

printf("\n");

}

printf("\n");

int child\_id = fork();

if (child\_id == 0)

{

sortAndPrint(numbers);

}

else

{

waitpid(child\_id, NULL, 0);

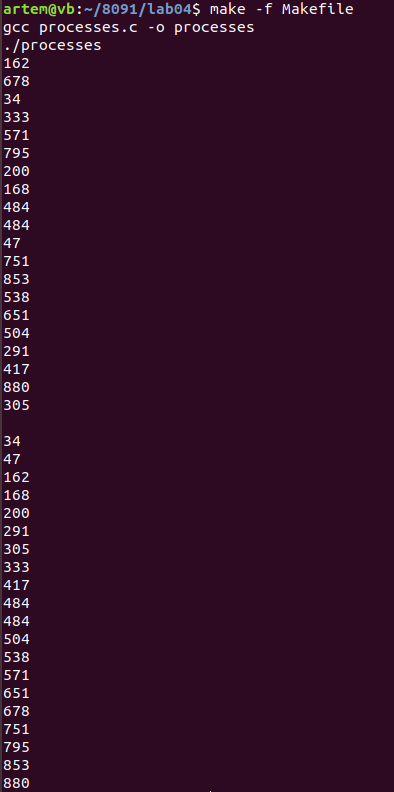
}

shmdt(numbers);

return 0;

}

**Результат:**



**Вывод:** В процессе выполнения лабораторной работы я на практике познакомился с механизмами запуска новых процессов в UNIX.