Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Кафедра информационных технологий и систем

**Отчет по лабораторной работе №4**

«Операционные системы»

Разработал:

Студент группы 9091

Манилов Т.В.\_\_\_\_\_\_\_

«\_\_».\_\_\_\_\_.2021г.

Проверил:

Ананьев В.В\_\_\_\_\_\_

«\_\_».\_\_\_\_\_.2021г.

**Великий Новгород**

**2021**

#include <sys/shm.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

void \*allocateSharedMemory(size\_t memSize, int &memId)

{

memId = shmget(IPC\_PRIVATE, memSize, 0600|IPC\_CREAT|IPC\_EXCL);

//доступ к разд памяти

if (memId <= 0)

{

perror("error with memId");

return NULL;

}

void \*mem = shmat(memId, 0, 0);

if (NULL == mem)

{

perror("error with shmat");

}

return mem;

}

void print()

{

for(int i=0;i<20;i++)

printf(\*mem[i])

}

void childMainCode(int \*sharedMem)

{

|  |
| --- |
| qsort(sharedMem, 20, 4,c\_sort);сортировка элементов в разделяемой памяти |
| print (sharedMem); выводятся на экран |
| exit(0);//завершение процесса |

}

int main()

{

int memId;

int \*sharedMem = (int \*)allocateSharedMemory(1024, memId);

printf("memId = %d\n", memId);

|  |  |
| --- | --- |
| srand(time(0)); | //генерируем 20 чисел |
| for(int i=0;i<20;i++) |
|  | \*(sharedMem +i)=rand() % 100;  print(sharedMem); |
| //размещаем в разделяемой памяти и выводим на экран |  |

pid\_t childId = fork();

//порождение процесса ребенка

и он сам делает сортировку элементов через функцию

if (childId < 0) не породился

{

perror("error with fork()\n");

ошибка }

else if (childId > 0) породился

{

childMainCode(sharedMem); }

else

{

sprintf(sharedMem, "init value for shared memory");

printf("Parent: This is main process. Child id = %d\n", childId);

printf("Parent: awaiting for child.\n");

waitpid(childId, NULL, 0); ожидает только прекращения работы потомка

printf("%6d proc: Process is finished.\n", curId);

процесс завершен }

char shared\_mem\_delete\_command[124];// очищение разд памяти

sprintf(shared\_mem\_delete\_command, "ipcrm -m %i", mem\_id);

system(shared\_mem\_delete\_command);

return 0;

}