## МИНОБРНАУКИ РОССИИ

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

## «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

**Кафедра Вычислительной техники**

## ОТЧЕТ

**по лабораторной работе №3**

## по дисциплине «Организация процессов и программирование в среде Linux»

**Тема: Создание и идентификация процессов**

|  |  |
| --- | --- |
| Студент гр. 8308 | Петров Г.А. |
| Преподаватель | Разумовский Г.В. |

Санкт-Петербург 2021

# Цель работы

Целью лабораторной работы является изучение и использование системных функций, обеспечивающих порождение и идентификацию процессов.

# Задание

Разработать программу, которая порождает 2 потомка. Первый потомок порождается с помощью fork, второй – с помощью vfork с последующей заменой на другую программу. Все 3 процесса должны вывести в один файл свои атрибуты с предварительным указанием имени процесса (например: Предок, Потомок1, Потомок2). Имя выходного файла задается при запуске программы. Порядок вывода атрибутов в файл должен определяться задержками процессов, которые задаются в качестве параметров программы и выводятся в начало файла. Откомпилировать программу и запустить ее 3 раза с различными сочетаниями задержек.

# Примеры выполнения программы

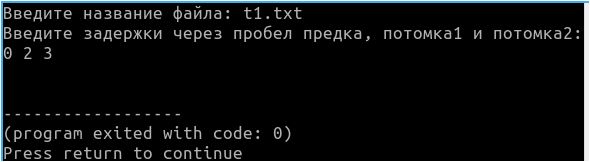
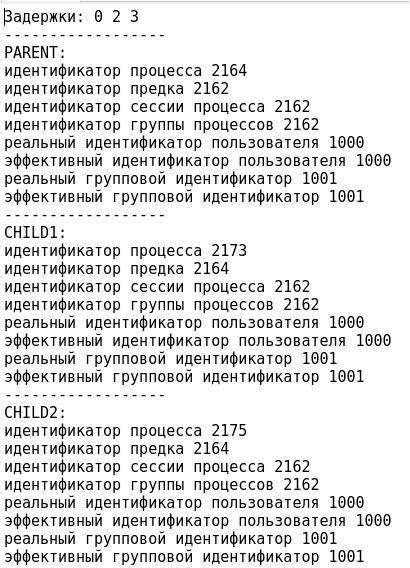
Выполним программу с задержками предка, потомка 1 и потомка 2 равными: 0, 2, 3. Ввод данных пользователем представлен на рис. 1.

Рисунок 1

Распечатка файла:



Выполним программу с задержками: 3, 2, 0. Ввод данных пользователем представлен на рис. 2.

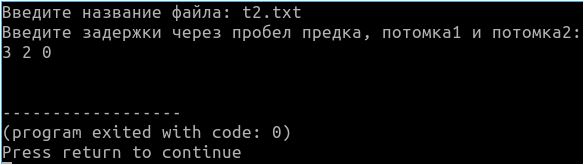
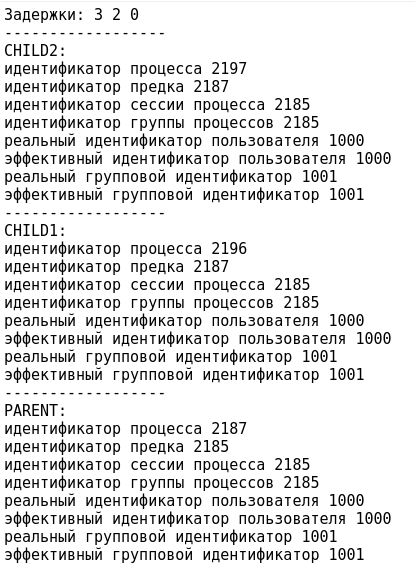


Рисунок 2

Распечатка файла:



Выполним программу с задержками: 2 0 3. Ввод данных пользователем представлен на рис. 3.

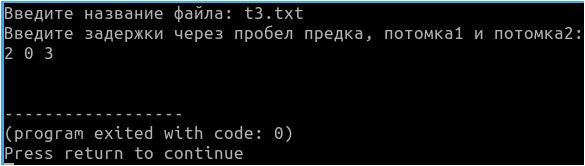
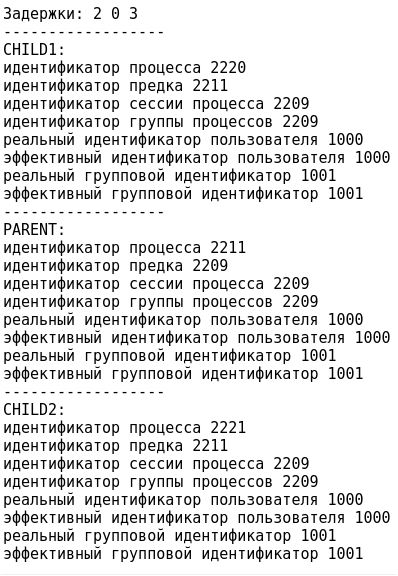


Рисунок 3

Распечатка файла:



# Исходный код программ

## lab3main.cpp

#include <iostream>

#include <fstream>

#include <stdio.h>

#include <stdlib.h>

#include <errno.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

*using* *namespace* std;

int **main**()

{

pid\_t pid1,pid2;

std::ofstream out;

char fname[60];

int t1,t2,status1,status2;

char t3[4];

std::cout<<"Введите название файла: ";

std::cin>>fname;

std::cout<<"Введите задержки через пробел предка, потомка1 и потомка2:\n";

std::cin>>t1>>t2>>t3;

out.open(fname, out.out | out.app);

out<<"Задержки: "<<t1<<" "<<t2<<" "<<t3<<"\n";

out.close();

*//создание* *процесса* *потомка1* *и* *предка*

*switch*(pid1=fork()) {

*case* -1:*//проверка* *ошибок* *при* *создании* *процесса*

perror("fork");

exit(1);

*case* 0:*//код* *выполнения* *процессом* *потомком1*

sleep(t2);*//регулируемая* *задержка* *потомка1*

*//вывод* *в* *файл* *информации* *о* *потомке1*

out.open(fname, out.out | out.app);

out<<"------------------\nCHILD1:"<<

"\nидентификатор процесса "<<getpid()<<

"\nидентификатор предка "<<getppid()<<

"\nидентификатор сессии процесса "<<getsid(getpid())<<

"\nидентификатор группы процессов "<<getpgid(getpid())<<

"\nреальный идентификатор пользователя "<<getuid()<<

"\nэффективный идентификатор пользователя "<<geteuid()<<

"\nреальный групповой идентификатор "<<getgid()<<

"\nэффективный групповой идентификатор "<<getegid()<<"\n";

out.close();

exit(EXIT\_SUCCESS);*//успешное* *завершение* *потомка1*

*default*:*//ко* *двыполнения* *предка*

*switch*(pid2=vfork()) {*//создание* *потомка2*

*case* -1:*//обработка* *ошибки* *создания* *потомка2*

perror("vfork");

exit(2);

*case* 0:

*//переход* *к* *выполнению* *кода* *другой* *программы* *с* *передачей* *параметров*

execl("child"," ",fname,t3,NULL);

exit(EXIT\_SUCCESS);*//успешное* *завершение* *потомка2*

*default*:*//код* *программы* *предка*

sleep(t1);*//задержка* *предка*

out.open(fname, out.out | out.app);

out<<"------------------\nPARENT:"<<

"\nидентификатор процесса "<<getpid()<<

"\nидентификатор предка "<<getppid()<<

"\nидентификатор сессии процесса "<<getsid(getpid())<<

"\nидентификатор группы процессов "<<getpgid(getpid())<<

"\nреальный идентификатор пользователя "<<getuid()<<

"\nэффективный идентификатор пользователя "<<geteuid()<<

"\nреальный групповой идентификатор "<<getgid()<<

"\nэффективный групповой идентификатор "<<getegid()<<"\n";

out.close();

waitpid(pid1,&status1,0);*//ожидание* *завершения* *потомка1*

waitpid(pid2,&status2,0);*//ожидание* *завершения* *потомка2*

}

}

*return* 0;

}

## child.cpp

#include <iostream>

#include <fstream>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

*//код* *программы* *потомка2*

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

std::ofstream out;*//поом* *вывода* *в* *файл*

out.open(argv[1], out.out | out.app);

sleep(atoi(argv[2]));*//задержка* *потомка2*

out<<"------------------\nCHILD2:"<<

"\nидентификатор процесса "<<getpid()<<

"\nидентификатор предка "<<getppid()<<

"\nидентификатор сессии процесса "<<getsid(getpid())<<

"\nидентификатор группы процессов "<<getpgid(getpid())<<

"\nреальный идентификатор пользователя "<<getuid()<<

"\nэффективный идентификатор пользователя "<<geteuid()<<

"\nреальный групповой идентификатор "<<getgid()<<

"\nэффективный групповой идентификатор "<<getegid()<<"\n";

out.close();

*return* 0;

}

# Вывод

При выполнении лабораторной работы изучены и использованы системные функции, обеспечивающие порождение и идентификацию процессов; разработана программа, порождающая 2 потомков и выводящая в файл информацию о процессах в очередности, заданной пользователей и обеспеченной функциями задержек процессов.