**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»**

Темы: Создание и идентификация процессов

|  |  |  |
| --- | --- | --- |
| Студент гр. 8306 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Пеунов В.В. |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Разумовский Г.В. |

Санкт-Петербург,

2021

**Цель работы**

Изучение и использование системных функций обеспечивающих порождение и идентификацию процессов.

**Задание на лабораторную работу**

1. 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.

2. Откомпилировать программу и запустить ее 3 раза с различными сочетаниями задержек

**Описание работы**

После написания программы было проведено 3 эксперимента. С разными значениями задержек.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Главный процесс | Первый ребенок | Второй ребенок |
| Эксперимент 1 | 1 | 3 | 5 |
| Эксперимент 2 | 1 | 5 | 3 |
| Эксперимент 3 | 10 | 5 | 3 |

**Эксперимент 1**

Потомок 1 - задержка: 3

Потомок 1 - идентификатор процесса: 39253

Потомок 1 - идентификатор предка: 39252

Потомок 1 - идентификатор сессии процесса: 3582

Потомок 1 - идентификатор группы процессов: 39252

Потомок 1 - реальный идентификатор пользователя: 1000

Потомок 1 - эффективный идентификатор пользователя: 1000

Потомок 1 - реальный групповой идентификатор: 1000

Потомок 1 - эффективный групповой идентификатор: 1000

Потомок 2 - задержка: 5

Потомок 2 - идентификатор процесса: 39254

Потомок 2 - идентификатор предка: 39252

Потомок 2 - идентификатор сессии процесса: 3582

Потомок 2 - идентификатор группы процессов: 39252

Потомок 2 - реальный идентификатор пользователя: 1000

Потомок 2 - эффективный идентификатор пользователя: 1000

Потомок 2 - реальный групповой идентификатор: 1000

Потомок 2 - эффективный групповой идентификатор: 1000

Родитель - задержка: 1

Родитель - идентификатор процесса: 39252

Родитель - идентификатор предка: 5019

Родитель - идентификатор сессии процесса: 3582

Родитель - идентификатор группы процессов: 39252

Родитель - реальный идентификатор пользователя: 1000

Родитель - эффективный идентификатор пользователя: 1000

Родитель - реальный групповой идентификатор: 1000

Родитель - эффективный групповой идентификатор: 1000

**Эксперимент 2**

Потомок 2 - задержка: 3

Потомок 2 - идентификатор процесса: 39380

Потомок 2 - идентификатор предка: 39377

Потомок 2 - идентификатор сессии процесса: 3582

Потомок 2 - идентификатор группы процессов: 39377

Потомок 2 - реальный идентификатор пользователя: 1000

Потомок 2 - эффективный идентификатор пользователя: 1000

Потомок 2 - реальный групповой идентификатор: 1000

Потомок 2 - эффективный групповой идентификатор: 1000

Родитель - задержка: 1

Родитель - идентификатор процесса: 39377

Родитель - идентификатор предка: 5019

Родитель - идентификатор сессии процесса: 3582

Родитель - идентификатор группы процессов: 39377

Родитель - реальный идентификатор пользователя: 1000

Родитель - эффективный идентификатор пользователя: 1000

Родитель - реальный групповой идентификатор: 1000

Родитель - эффективный групповой идентификатор: 1000

Потомок 1 - задержка: 5

Потомок 1 - идентификатор процесса: 39379

Потомок 1 - идентификатор предка: 1

Потомок 1 - идентификатор сессии процесса: 3582

Потомок 1 - идентификатор группы процессов: 39377

Потомок 1 - реальный идентификатор пользователя: 1000

Потомок 1 - эффективный идентификатор пользователя: 1000

Потомок 1 - реальный групповой идентификатор: 1000

Потомок 1 - эффективный групповой идентификатор: 1000

**Эксперимент 3**

Потомок 2 - задержка: 3

Потомок 2 - идентификатор процесса: 39967

Потомок 2 - идентификатор предка: 39965

Потомок 2 - идентификатор сессии процесса: 3582

Потомок 2 - идентификатор группы процессов: 39965

Потомок 2 - реальный идентификатор пользователя: 1000

Потомок 2 - эффективный идентификатор пользователя: 1000

Потомок 2 - реальный групповой идентификатор: 1000

Потомок 2 - эффективный групповой идентификатор: 1000

Потомок 1 - задержка: 5

Потомок 1 - идентификатор процесса: 39966

Потомок 1 - идентификатор предка: 39965

Потомок 1 - идентификатор сессии процесса: 3582

Потомок 1 - идентификатор группы процессов: 39965

Потомок 1 - реальный идентификатор пользователя: 1000

Потомок 1 - эффективный идентификатор пользователя: 1000

Потомок 1 - реальный групповой идентификатор: 1000

Потомок 1 - эффективный групповой идентификатор: 1000

Родитель - задержка: 10

Родитель - идентификатор процесса: 39965

Родитель - идентификатор предка: 5019

Родитель - идентификатор сессии процесса: 3582

Родитель - идентификатор группы процессов: 39965

Родитель - реальный идентификатор пользователя: 1000

Родитель - эффективный идентификатор пользователя: 1000

Родитель - реальный групповой идентификатор: 1000

Родитель - эффективный групповой идентификатор: 1000

**Вывод:** в ходе лабораторной работы была проделана работа по изучению функций fork и vfork и сопутствующих им для порождения процессов.

**Приложение A. Основная программа**

#include <iostream>

#include <fstream>

#include <unistd.h>

#include <string>

using namespace std;

int mainStop = 10;

int childFirstStop = 5;

int childSecondStop = 3;

const char \*childFirstProgramPath = "/home/peunov/highschool/opp-linux/laba1.2/cmake-build-debug/laba1\_2";

void outputInFile(string processName, string filePath, int stop);

int error();

string getFilePath();

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

if(argc == 4){

mainStop = atoi(argv[1]);

childFirstStop = atoi(argv[2]);

childSecondStop = atoi(argv[3]);

}

string filePath = getFilePath();

pid\_t fork\_process\_id = fork();

if(fork\_process\_id == -1){

return error();

}

if(fork\_process\_id == 0){

sleep(childFirstStop);

outputInFile("Потомок 1", filePath, childFirstStop);

exit(EXIT\_SUCCESS);

}

if(fork\_process\_id > 0){

pid\_t vfork\_process\_id = vfork();

if(vfork\_process\_id == -1){

return error();

}

if(vfork\_process\_id > 0){

sleep(mainStop);

outputInFile("Родитель", filePath, mainStop);

exit(EXIT\_SUCCESS);

}

if(vfork\_process\_id == 0){

sleep(childSecondStop);

outputInFile("Потомок 2", filePath, childSecondStop);

execlp(childFirstProgramPath, nullptr);

exit(EXIT\_SUCCESS);

}

}

return 0;

}

void outputInFile(string processName, string filePath, int stop){

ofstream file;

file.open(filePath, ios::app);

if (file.is\_open()) {

pid\_t process\_id = getpid();

file << processName << " - задержка: " << stop << endl;

file << processName << " - идентификатор процесса: " << process\_id << endl;

file << processName << " - идентификатор предка: " << getppid() << endl;

file << processName << " - идентификатор сессии процесса: " << getsid(process\_id) << endl;

file << processName << " - идентификатор группы процессов: " << getpgid(process\_id) << endl;

file << processName << " - реальный идентификатор пользователя: " << getuid() << endl;

file << processName << " - эффективный идентификатор пользователя: " << geteuid() << endl;

file << processName << " - реальный групповой идентификатор: " << getgid() << endl;

file << processName << " - эффективный групповой идентификатор: " << getegid() << endl;

}

file.close();

}

int error(){

cout << "При создании процесса произошла ошибка";

return 1;

}

string getFilePath(){

string filePath;

cout << "Введите путь к файлу: ";

cin >> filePath;

return filePath;

}

**Приложение B. Заменяемая программа**

#include <string>

int main() {

exit(EXIT\_SUCCESS);

}