Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

за 4 семестр

По дисциплине: «ОСиСП»

Тема: «GCC. Процессы»

Выполнил:

Студент 2 курса

Группы ПО-6(1)

Мартынович Д. М.

Проверил:

Давидюк Ю. И.

2022

Лабораторная работа №

GCC. Процессы

Вариант 8

Цель работы: изучить работу с процессами в ОС Linux.

**Задание для выполнения**

Написать программу, которая будет реализовывать следующие функции:

• сразу после запуска получает и сообщает свой ID и ID родительского процесса;

• перед каждым выводом сообщения об ID процесса и родительского процесса эта

информация получается заново;

• порождает процессы, формируя генеалогическое дерево согласно варианту, сообщая,

что "процесс с ID таким-то породил процесс с таким-то ID";

• перед завершением процесса сообщить, что "процесс с таким-то ID и таким-то ID

родителя завершает работу";

• один из процессов должен вместо себя запустить программу, указанную в варианте

задания.

На основании выходной информации программы предыдущего пункта изобразить

генеалогическое дерево процессов (с указанием идентификаторов процессов). Объяснить

каждое выведенное сообщение и их порядок в предыдущем пункте.



#include <unistd.h>

#include <sys/types.h>

#include <stdio.h>

#include <stdlib.h>

int **main**(){

*//1 process*

    pid\_t pid;

**printf("Создание процесса 1: PID=%d, PPID=%d\n", getpid(),getppid())**;

*//2 process*

    if((pid=**fork()**)==-1){

**printf("Error!\n")**;

    }

    else if(pid==0){

**printf("Создание процессса 2: PID=%d, PPID=%d\n", getpid(), getppid())**;

*//3 process*

        if((pid=**fork()**)==-1){

**printf("Error!\n")**;}

        else if(pid==0){

**printf("Создание процессса 3: PID=%d, PPID=%d\n", getpid(), getppid())**;

*//5 process*

            if((pid=**fork()**)==-1){

**printf("Error!\n")**;}

                    else if(pid==0){

**printf("Создание процессса 5: PID=%d, PPID=%d\n", getpid(), getppid())**;

*//7 process*

                if((pid=**fork()**)==-1){

**printf("Error!\n")**;}

                            else if(pid==0){

**printf("Создание процессса 7: PID=%d, PPID=%d\n", getpid(), getppid())**;

**printf("Завершение процесса 7: PID=%d, PPID=%d\n", getpid(), getppid())**;

**exit(0)**;

                        }else **sleep(1)**;

**printf("Завершение процесса 5: PID=%d, PPID=%d\n", getpid(), getppid())**;

**exit(0)**;

                }else **sleep(2)**;

**printf("Завершение процесса 3: PID=%d, PPID=%d\n", getpid(), getppid())**;

**exit(0)**;

        }else **sleep(1)**;

**sleep(3)**;

        if((pid=**fork()**)==-1){

**printf("Error!\n")**;}

        else if(pid==0){

**printf("Создание процессса 4: PID=%d, PPID=%d\n", getpid(), getppid())**;

*//6 process*

            if((pid=**fork()**)==-1){

**printf("Error!\n")**;}

            else if(pid==0){

**printf("Создание процессса 6: PID=%d, PPID=%d\n", getpid(), getppid())**;

**printf("Завершение процесса 6: PID=%d, PPID=%d\n", getpid(), getppid())**;

**exit(0)**;

            }else **sleep(1)**;

**printf("Завершение процесса 4: PID=%d, PPID=%d\n", getpid(), getppid())**;

**exit(0)**;

        }else **sleep(2)**;

**printf("Завершение процесса 2: PID=%d, PPID=%d\n", getpid(), getppid())**;

**exit(0)**;

}else **sleep(3)**;

**sleep(4)**;

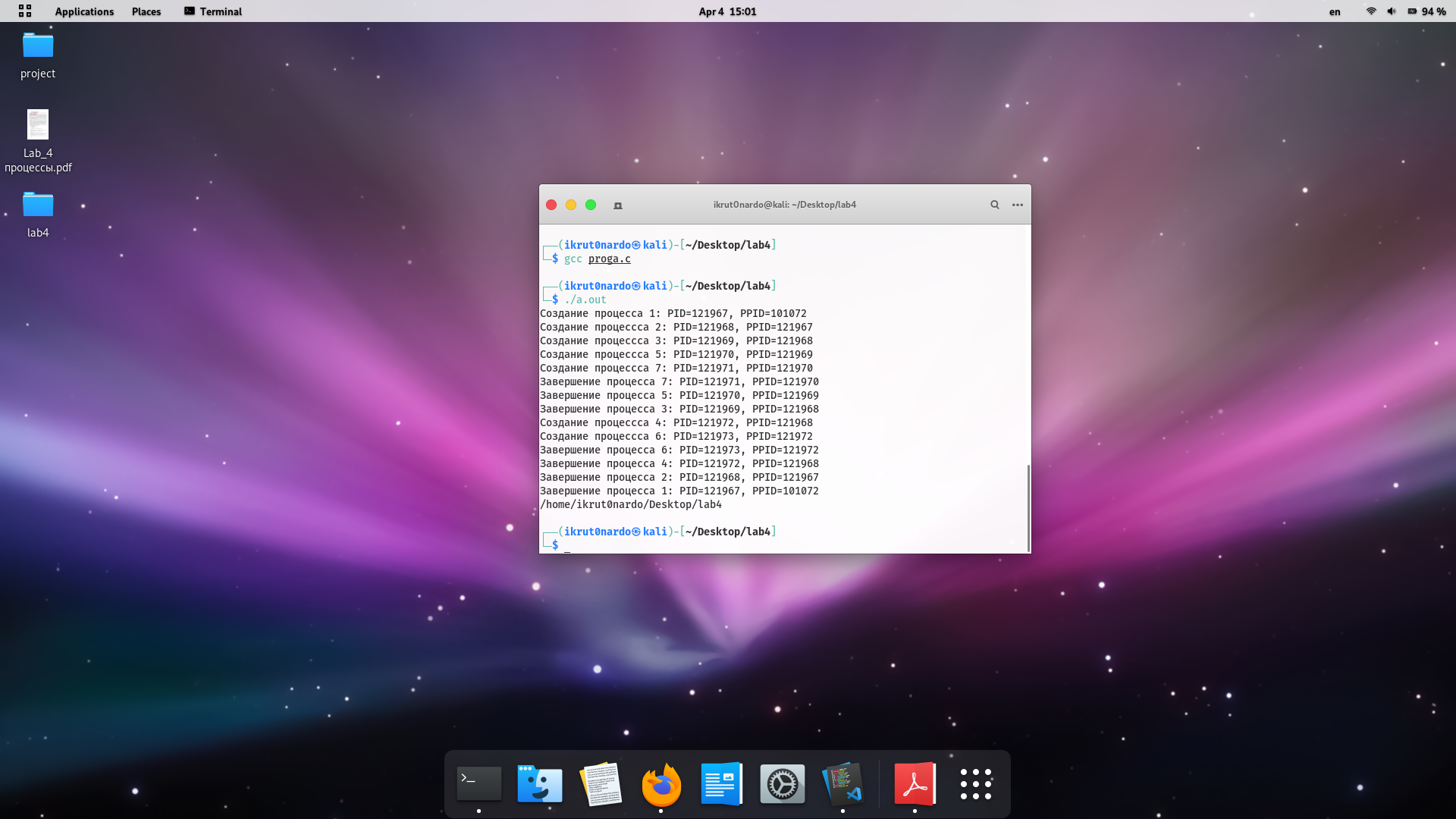
**printf("Завершение процесса 1: PID=%d, PPID=%d\n", getpid(), getppid())**;

**execl("/bin/pwd", "pwd", NULL)**;

**exit(0)**;

return 1;

}



Первый процесс порождает второй процесс, второй процесс порождает третий процесс, третий процесс порождает пятый, пятый процесс порождает седьмой, завершаем седьмой процесс, завершаем пятый процесс, завершаем третий процесс, второй процесс порождает четвертый процесс, четвертый процесс порождает шестой процесс, завершаем шестой процесс, завершаем четвертый процесс, завершаем второй процесс, завершаем первый процесс.

Вывод: изучил работу с процессами.