**МИНОБРНАУКИ РОССИИ**

**САНКТ–ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра ВТ**

**ОТЧЕТ**

**по лабораторной работе №6**

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

**Тема: «ОРГАНИЗАЦИЯ ПЕРИОДИЧЕСКИХ**

**ПРОЦЕССОВ»**

Cтудентка гр. 7306 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мищенко А.В.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Разумовский Г.В.

Санкт–Петербург

2020

**Цель работы:**

Использование сервиса cron, механизма сигналов и интервальных таймеров для организации периодических процессов.

**Задание:**

1. Создать пользовательский файл конфигурации сервиса cron, в котором содержатся команды периодического запуска одной из программ, разработанных в предыдущих лабораторных работах. Результаты работы этой программы должны выводиться или переадресоваться в файл.
2. Написать периодическую программу, в которой период запуска и количество запусков должны задавать в качестве ее параметров. При каждом очередного запуске программа должна порождать новый процесс, который выводит на экран свой идентификатор, дату и время старта. Программа и ее дочерний процесс должны быть заблокированы от завершения при нажатии клавиши Ctrl+z. После завершения дочернего процесса программа должна вывести на экран информацию о времени своей работы и дочернего процесса.

**Ход работы:**

1. Для выполнения работы был создан файл с инструкцией \*/1 \* 24 12 \* /home/alina/Документы/test/testProgram >>/home/alina/Документы/test/file.log, при ее запуске с помощью cron программа testProgram будет запускаться через каждую минуту 24 декабря и записывать свое содержмое в файл file.log. На рисунке 1 представлен пример загрузки инструкции в cron.

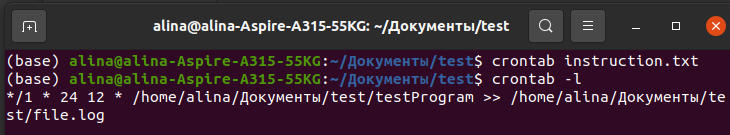


Рисунок 1. Загрузка файла инструкции

Для удаления инструкции используется команда crontab -r.

1. В ходе выполнения работы была написана программа main.cpp, исходный код которой представлен в Приложении 1.

При запуске приложения можно передать в качестве аргументов число запусков и время между запусками. На рисунке 2 представлен пример выполнения программы с заданием 3 запусков с интервалом в 2 секунды. Также было реализован перехват сигнала нажатия Ctrl+Z, чтобы программа при нажатии данного сочетания клавиш не приостанавливалась. Пример представлен на рисунке 3.

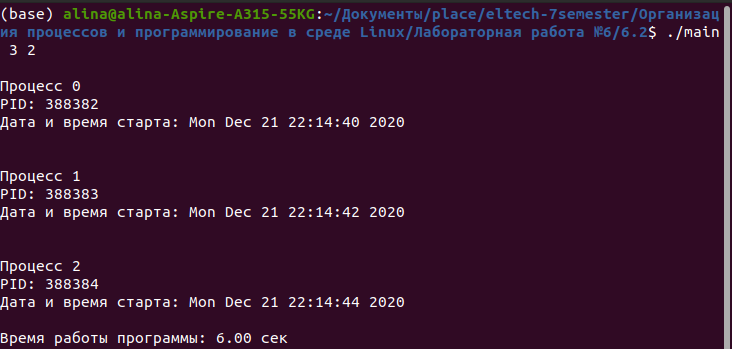


Рисунок 2. Пример работы программы

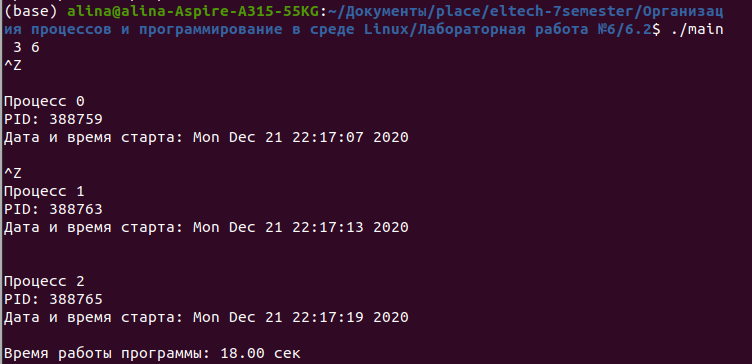


Рисунок 3. Проверка реакции на нажатие Ctrl+Z

**Вывод:**

В ходе лабораторной работы я познакомилась с интервальными таймерами и научилась пользоваться инструментом cron для задания периодического запуска приложений.

**Приложения**

Приложение 1. Файл main.cpp

#include <sys/time.h>

#include <sys/wait.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <time.h>

#include <signal.h>

typedef struct init\_param {

int run\_counts; /\* Кол-во запусков \*/

long tv\_sec; /\* Период повторения в сек \*/

} inparam;

void init\_param(inparam\* param) {

printf("Параметры программы\nКоличество запусков: ");

scanf("%d", &param->run\_counts);

printf("Период повторения (с): ");

scanf("%ld", &param->tv\_sec);

}

int g\_current\_counts = 0;

void signal\_handler(int signum) {

if (signum == SIGALRM) {

if (fork() == 0) {

time\_t t = time(NULL);

tm\* start\_time = localtime(&t);

printf("\nПроцесс %d\nPID: %d\nДата и время старта: %s\n", g\_current\_counts, getpid(), asctime(start\_time));

exit(EXIT\_SUCCESS);

}

++g\_current\_counts;

}

}

//argv[1] - количество запусков; argv[2] - время между запусками

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

signal(SIGALRM,signal\_handler);

signal(SIGTSTP, SIG\_IGN);

time\_t start, end;

inparam param;

if(argc < 3)

init\_param(&param);

else

{

param.run\_counts = atoi(argv[1]);

param.tv\_sec = atoi(argv[2]);

}

time(&start);

struct itimerval value, ovalue;

//устанавливаем период для первого запуска

value.it\_value.tv\_sec = param.tv\_sec;

value.it\_value.tv\_usec = 0;

//Устанавливаем период по умолчанию перезапуска

value.it\_interval.tv\_sec = param.tv\_sec;

value.it\_interval.tv\_usec = 0;

setitimer(ITIMER\_REAL, &value, &ovalue);

int status;

for(int count = 0; count < param.run\_counts; count++)

{

pause();

}

time(&end);

printf("Время работы программы: %.2f сек\n", difftime(end, start));

return EXIT\_SUCCESS;

}