|  |
| --- |
| «Санкт-Петербургский государственный электротехнический университет |
| «ЛЭТИ» им. В.И.Ульянова(Ленина)» |
| (СПбГЭТУ «ЛЭТИ») Кафедра вычислительной техники |
|  |
| **Отчет**  **по лабораторным работам № 6**  **по дисциплине** **«Организация процессов и программирование в среде Linux»** |
| Тема: «Организация периодических процессов» |
|  |

|  |
| --- |
|  |

|  |  |
| --- | --- |
| Студент гр. 9307 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_Брызгалова Е. А. |
|  |  |
|  |  |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Разумовский Г. В. |

Оглавление

[Цель работы: 2](#_Toc116211715)

[Задание 2](#_Toc116211716)

[Скриншоты экрана результатов работы программы 2](#_Toc116211717)

[Текст программы 3](#_Toc116211718)

[Текст программы main.cpp 3](#_Toc116211719)

[Вывод 3](#_Toc116211720)

Цель работы:

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

Задание

1. Написать периодическую программу, в которой период запуска и количество запусков должны задаваться в качестве ее параметров. При каждом очередном запуске программа должна порождать новый процесс, который выводить 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.
2. Откомпилировать программу и запустить ее несколько раз с разным периодом запуска и количеством повторений.

Скриншот экрана результатов работы периодической программы после всех ее перезапусков.

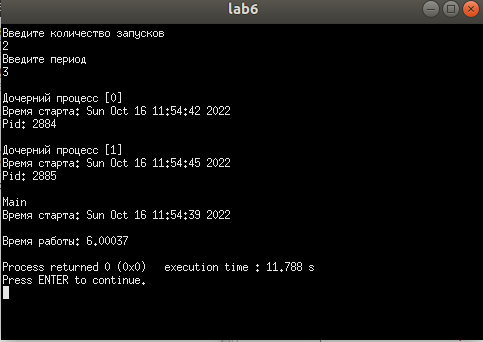


Рисунок 1

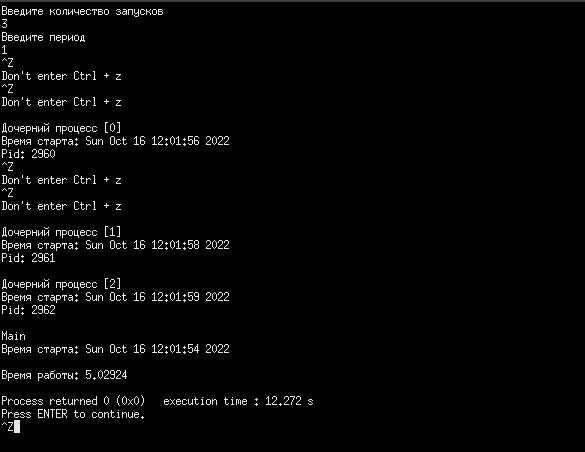


Рисунок 2

Текст периодической программы

### Текст программы main.cpp

#include <stdio.h>

#include <stdlib.h>

#include <sys/wait.h>

#include <unistd.h>

#include <string.h>

#include <signal.h>

#include <sys/time.h>

#include <iostream>

#include <chrono>

#include <ctime>

using namespace std;

int iteration = 0;

int number\_of\_starts;

int period;

void a(int sig)

{

cout<<"\nDon't enter Ctrl + z\n";

}

void catch\_signal(int sig) {

if(vfork() == 0){

auto start = chrono::system\_clock::now();

time\_t start\_time = chrono::system\_clock::to\_time\_t(start);

auto end = std::chrono::system\_clock::now();

chrono::duration<double> elapsed\_seconds = end-start;

cout<<"\nДочерний процесс ["<<iteration<< "]\nВремя старта: "<<ctime(&start\_time)<<"Pid: "<<getpid()<<endl;

exit(0);

}

iteration++;

}

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

struct sigaction s;

sigemptyset(&s.sa\_mask);

s.sa\_flags = SA\_RESTART;

s.sa\_handler = a;

sigaction(20, &s, NULL);

{

cout << "Введите количество запусков\n";

cin >> number\_of\_starts;

cout << "Введите период\n";

cin >> period;

}

chrono::system\_clock::time\_point start = chrono::system\_clock::now();//текущее время

struct sigaction al;

al.sa\_handler = catch\_signal;

sigaction(14, &al, NULL);

struct itimerval it;

memset(&it, 0, sizeof(it));

it.it\_interval.tv\_sec = period;// новое значение таймера

it.it\_value.tv\_sec = period; // текущее значение таймера

for(iteration;iteration<number\_of\_starts;)

{

if(setitimer(ITIMER\_REAL, &it, NULL) == -1)//Установить значения таймеров

cout<<"error";

pause();

}

chrono::system\_clock::time\_point finish= chrono::system\_clock::now();//текущее время

time\_t start\_time = chrono::system\_clock::to\_time\_t(start);

chrono::duration<double> time = finish-start;

cout <<"\nMain \nВремя старта: "<< ctime(&start\_time)<<"\nВремя работы: "<< time.count() << endl;

return 0;

}

Вывод

В процессе выполнения лабораторной работы познакомились с сервисом cron, механизмами сигналов и интервальных таймеров для организации периодических процессов.