

**«Санкт-Петербургский государственный электротехнический
университет «ЛЭТИ» им. В.И.Ульянова (Ленина)»
СПбГЭТУ «ЛЭТИ»**

Кафедра вычислительной техники

Отчет по по лабораторной работе №6 по дисциплине
«Организация процессов и программирование в среде Linux»
Тема: «ОРГАНИЗАЦИЯ ПЕРИОДИЧЕСКИХ ПРОЦЕССОВ»

Студент гр.8306

Преподаватель

Слепов А.Э.

Разумосвский Г.В.

Санкт-Петербург

2021

Цель работы

Использование механизма сигналов и интервальных таймеров для организации периодических процессов.

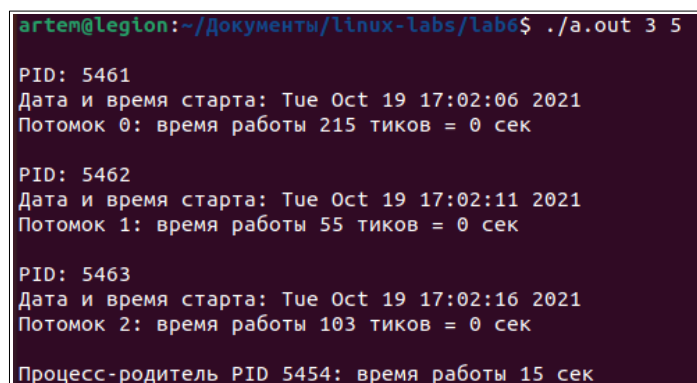
Задание

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.

Порядок выполнения работы

Результат работы программы для 3 запусков с периодом 5 секунд представлен на рисунке 1. Процессы потомки только выводят строку на экран, поэтому работают меньше секунды, поэтому их время работы считалось в тиках процессора с помощью `clock()`. Из времени запуска видно, что процессы стартуют согласно заданному периоду 5 секунд. Процесс родитель в общей сложности проработал 5 секунд, так как было запланировано 3 запуска с периодом 5 секунд.



```
artem@legion:~/документы/linux-labs/lab6$ ./a.out 3 5
PID: 5461
Дата и время старта: Tue Oct 19 17:02:06 2021
Потомок 0: время работы 215 тиков = 0 сек

PID: 5462
Дата и время старта: Tue Oct 19 17:02:11 2021
Потомок 1: время работы 55 тиков = 0 сек

PID: 5463
Дата и время старта: Tue Oct 19 17:02:16 2021
Потомок 2: время работы 103 тиков = 0 сек

Процесс-родитель PID 5454: время работы 15 сек
```

Рисунок 1. Результаты работы программы для 3 запусков с периодом 5 секунд

Результат работы программы для 5 запусков с периодом 1 секунду представлен на рисунке 2.

```
artem@legion:~/Документы/linux-labs/lab6$ ./a.out 5 1
PID: 5467
Дата и время старта: Tue Oct 19 17:02:41 2021
Потомок 0: время работы 128 тиков = 0 сек

PID: 5469
Дата и время старта: Tue Oct 19 17:02:42 2021
Потомок 1: время работы 130 тиков = 0 сек

PID: 5470
Дата и время старта: Tue Oct 19 17:02:43 2021
Потомок 2: время работы 79 тиков = 0 сек

PID: 5471
Дата и время старта: Tue Oct 19 17:02:44 2021
Потомок 3: время работы 119 тиков = 0 сек

PID: 5472
Дата и время старта: Tue Oct 19 17:02:45 2021
Потомок 4: время работы 120 тиков = 0 сек

Процесс-родитель PID 5466: время работы 5 сек
```

Рисунок 2. Результаты работы программы для 5 запусков с периодом 1 секунду

Результат работы программы для 0 запусков с периодом 5 секунду представлен на рисунке 3. Можно наблюдать, что ни один процесс не был порожден, так как было задано 0 запусков. Процесс родитель никого не ждал и работал 0 секунд.

```
artem@legion:~/Документы/linux-labs/lab6$ ./a.out 0 5
Процесс-родитель PID 5465: время работы 0 сек
```

Рисунок 3. Результаты работы программы для 0 запусков с периодом 5 секунду

Выводы

В ходе работы были изучены механизмы сигналов и интервальных таймеров для организации периодических процессов в операционной системе Ubuntu.

ПРИЛОЖЕНИЕ А

Текст программы

```
#include <sys/time.h>
#include <sys/wait.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <time.h>
#include <signal.h>

void signal_handler(int sig){
    if (sig == SIGALRM) {
        if (fork() == 0) {
            time_t start_time = time(NULL);
            printf("\nPID: %d\nДата и время старта: %s", getpid(),
ctime(&start_time));
            exit(EXIT_SUCCESS);
        }
    }
}

void main(int argc, char** argv){
    if(argc == 3){
        int numStarts = atoi(argv[1]);
        int period = atoi(argv[2]);
        signal(SIGTSTP, SIG_IGN);
        signal(SIGALRM, signal_handler);

        struct itimerval timer_value;
        timerclear(&timer_value.it_interval);
        timerclear(&timer_value.it_value);
        timer_value.it_interval.tv_sec = period;
        timer_value.it_value.tv_sec = period;
        setitimer(ITIMER_REAL, &timer_value, NULL);

        time_t pt1,pt2;
        clock_t p1, p2;
        clock_t pt3, pt4;
        pt1 = time(NULL);
        p1 = clock();
        for(int i = 0; i < numStarts; i++){
            pause();
            pt3 = clock();
            wait(0);
            pt4 = clock();
            printf("Потомок %d: время работы %lu тиков = %ld сек\n", i, pt4-pt3,
(pt4-pt3)/ CLOCKS_PER_SEC);
        }
        p2 = clock();
        pt2 = time(NULL);
        printf("\nПроцесс-родитель PID %d: время работы %lu сек \n",
getpid(), pt2-pt1);
    }
    else
        printf("Недостаточно аргументов для запуска\n");
    exit(EXIT_SUCCESS);
}
```