

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

Кафедра вычислительной техники

Отчет по по лабораторной работе №3 по дисциплине
«Организация процессов и программирование в среде Linux»
Тема: «ОБРАБОТКА СИГНАЛОВ»

Студент гр.8306

Преподаватель

Слепов А.Э.

Разумосвский Г.В.

Санкт-Петербург

2021

Цель работы

Знакомство с механизмом сигналов и способами их обработки.

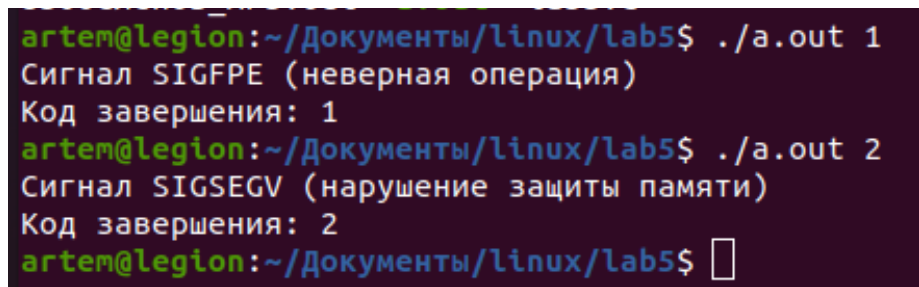
Задание

1. Написать программу, которая реагирует на ошибки при выполнении операции деления и неверном использовании указателя (деление на ноль, нарушение защиты памяти). При обнаружении ошибки программа должна передать управление функции, которая выведет сообщение и завершит работу программы с кодом ошибки (1 или 2). Тип ошибки, который должна зафиксировать программа, задается как параметр при ее запуске.

2. Откомпилировать программу и дважды запустить ее с разными значениями типа ошибки.

Порядок выполнения работы

Результат обработки заданных сигналов представлен на рисунке 1. В качестве ошибки №1 представлено деление числа на 0. В качестве ошибки номер №2 используется разыменование указателя со значением NULL.



```
artem@legion:~/Документы/linux/lab5$ ./a.out 1
Сигнал SIGFPE (неверная операция)
Код завершения: 1
artem@legion:~/Документы/linux/lab5$ ./a.out 2
Сигнал SIGSEGV (нарушение защиты памяти)
Код завершения: 2
artem@legion:~/Документы/linux/lab5$
```

Рисунок 1. Результаты запуска программы

Выводы

В ходе работы были изучены механизмы отправки и обработки сигналов в программах под операционную систему Ubuntu.

ПРИЛОЖЕНИЕ А

Текст программы-родителя

```
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>

void signal_handler(int sig) {
    switch (sig)
    {
        case SIGFPE:
            printf("Сигнал SIGFPE (неверная операция) \nКод завершения: 1\n");
            exit(1);
            break;
        case SIGSEGV:
            printf("Сигнал SIGSEGV (нарушение защиты памяти) \nКод завершения: 2\n");
            exit(2);
            break;
        default:
            printf("Необработанный сигнал");
            break;
    }
}

int main(int argc, char* argv[]){
    signal(SIGFPE, signal_handler);
    signal(SIGSEGV, signal_handler);
    int option, a = 10;
    int *pointer = NULL;
    if(argc == 2){
        option = atoi(argv[1]);
        if(option == 1){
            a = a / 0;
        }
        if(option == 2){
            printf("%d", *pointer);
        }
    }
    else
        printf("Нет аргументов для запуска");

    return 0;
}
```