

1 Задание

Разработать на языке C для ОС Linux программу, позволяющую выполнять рекурсивный поиск файлов, начиная с указанного каталога, в соответствии с условием из Табл. 3 и вариантом из Табл. 4.

Программа должна представлять собой консольную утилиту, настройка работы которой осуществляется путем передачи аргументов в строке запуска и/или с помощью переменных окружения (опции необязательны, аргументы каталог и цель_поиска — обязательны):

```
lab11abcNXXXXXX [опции] каталог цель_поиска
```

Программа должна выполнять рекурсивный поиск файлов, отвечающих критерию, который задается аргументом цель_поиска в соответствии с условием из Табл. 3. При обнаружении файла, отвечающего заданным критериям поиска, программа должна вывести в стандартный поток вывода полный путь к этому файлу.

При указании опций -h или -v (или их "длинных" аналогов --help или --version) выполняется вывод информации, заданной опцией, и работа программы завершается. Опции, которые должны поддерживаться программой, приведены в Табл. 1.

При определении переменной окружения LAB11DEBUG в стандартный поток ошибок должна выводиться информация о том, что и в каком месте файла нашлось (чтобы было легче понять, почему файл отвечает критериям поиска), а также может выводиться любая дополнительная отладочная информация. Переменные окружения, которые должны поддерживаться программой, приведены в Табл. 2.

Обход осуществляется в соответствии с вариантом 3 – ftw().

Выполняется поиск заданной последовательности байтов. Аргумент цель_поиска имеет формат строки в кодировке UTF-8.

2 Make-file

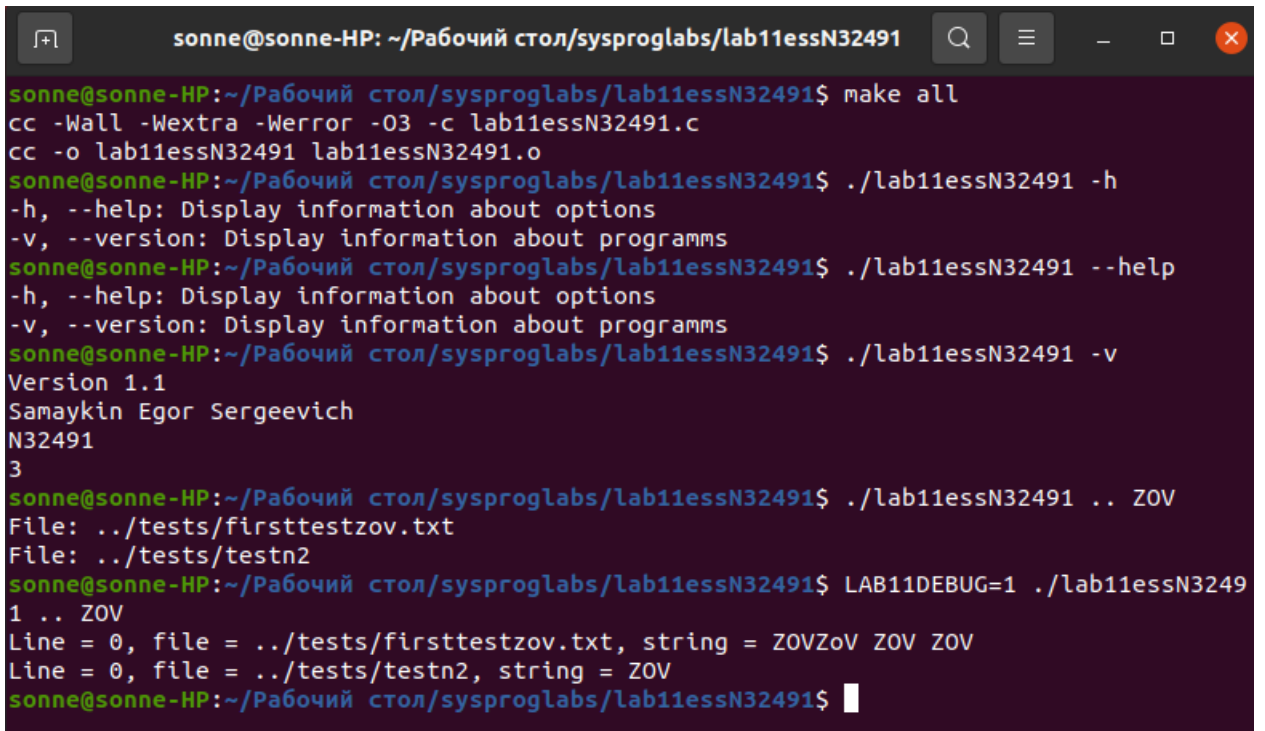
```
APP=lab11essN32491
SRCS=lab11essN32491.c
OBS=$(SRCS:.c=.o)
CC=gcc
CFLAGS=-Wall -Wextra -Werror -O3
LDFLAGS=-lm

.PHONY: all clean
all: $(APP)
clean:
    rm -rf *.o $(APP)
$(APP): $(OBS)
    $(CC) -o $@ $^
.c.o:
    $(CC) $(CFLAGS) -c $<
```

3 Отчет valgrind

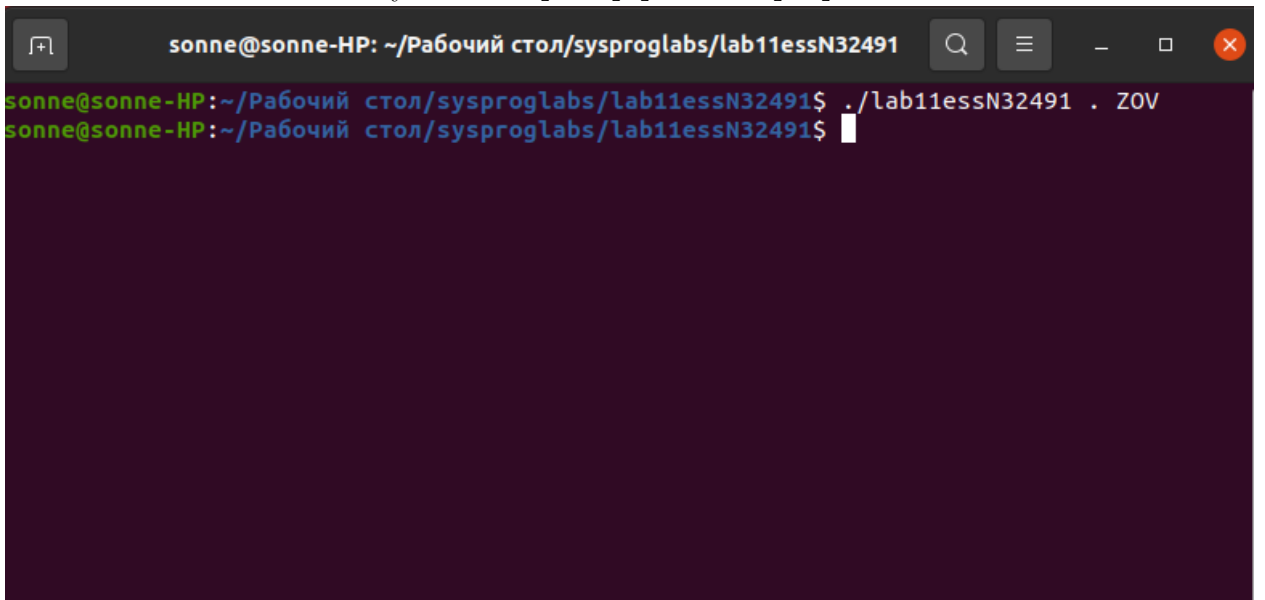
```
==54948== HEAP SUMMARY:
==54948==      in use at exit: 0 bytes in 0 blocks
==54948==    total heap usage: 150,072 allocs, 150,072 frees,
718,488,322 bytes allocated
==54948==
==54948== All heap blocks were freed -- no leaks are possible
==54948==
==54948== For lists of detected and suppressed errors, rerun with: -s
==54948== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from
0)
```

4 Работа программ



```
sonne@sonne-HP: ~/Рабочий стол/sysproglabs/lab11essN32491
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ make all
cc -Wall -Wextra -Werror -O3 -c lab11essN32491.c
cc -o lab11essN32491 lab11essN32491.o
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ ./lab11essN32491 -h
-h, --help: Display information about options
-v, --version: Display information about programmes
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ ./lab11essN32491 --help
-h, --help: Display information about options
-v, --version: Display information about programmes
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ ./lab11essN32491 -v
Version 1.1
Samaykin Egor Sergeevich
N32491
3
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ ./lab11essN32491 .. ZOV
File: ../tests/firsttestzov.txt
File: ../tests/testn2
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ LAB11DEBUG=1 ./lab11essN3249
1 .. ZOV
Line = 0, file = ../tests/firsttestzov.txt, string = ZOVZoV ZOV ZOV
Line = 0, file = ../tests/testn2, string = ZOV
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$
```

Рисунок 1 – пример работы программы



```
sonne@sonne-HP: ~/Рабочий стол/sysproglabs/lab11essN32491
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$ ./lab11essN32491 . ZOV
sonne@sonne-HP:~/Рабочий стол/sysproglabs/lab11essN32491$
```

Рисунок 2 – пример работы программы

5 Тексты программ

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <dirent.h>
#include <string.h>
#include <getopt.h>
#include <errno.h>
#include <ftw.h>    //Хед для ftw обхода
#include <sys/param.h>
#include <malloc.h>

#define _GNU_SOURCE //Макрос для поддержки ф-ции getline

//гп для хранения директории поиска и строки соответственно
char* dir_search;
char* targ_search;

//Флаг отладки
int Debug_FLAG = 0;

//Поиск целевой строки
void CheckFileLine(const char* fpath)
{
    FILE* fp = fopen(fpath, "r");
    if (!fp && Debug_FLAG) {
        fprintf(stderr, "Unable to open! %s\n", fpath);
    }

    char* buffer = NULL;
    size_t buffer_length = 0;
    ssize_t nread;
    size_t Nline = 0;
    while ((nread = getline(&buffer, &buffer_length, fp)) != -1) {
        if (strstr(buffer, targ_search) != NULL) {
            if (Debug_FLAG)
                fprintf(stdout, "Line = %ld, file = %s, string = %s", Nline, fpath,
buffer);
            else
                fprintf(stdout, "File: %s\n", fpath);
        }
        Nline++;
    }
    free(buffer);
    fclose(fp);
}

/*Принимает инфу о текущем пути файла fpath, структуру stat и флаг typeflag, указывающий
тип файла. Если файл является обычным файлом FTW_F, то функция вызывает CheckFileLine()
для анализа содержимого файла */
int CheckFile(const char* fpath, const struct stat* sb, int typeflag)
{
    (void)sb;
    if (typeflag == FTW_F)
        CheckFileLine(fpath);
    return 0;
}

/*Начинает обход ФС, начиная с данной директории dir_search. Используется ftw() для
рекурсивного обхода дерева каталогов и вызывает CheckFile() для каждого файла*/
```

```

void WalkDirectory()
{
    int result = ftw(dir_search, CheckFile, FTW_NS);
    if (result < 0 && Debug_FLAG) {
        fprintf(stderr, "ftw() failed: %s\n", strerror(errno));
    }
}

/*Функция для разбора аргументов ком строки, getopt нужен для обработки опций*/
void OptionDo(int argc, char* const* argv)
{
    int value = 0;
    opterr = 0;
    static struct option long_options[] = {
        { "help", no_argument, 0, 'h' },
        { "version", no_argument, 0, 'v' }
    };

    while (1) {
        int optindex = 0;
        value = getopt_long(argc, argv, "hv", long_options, &optindex);
        if (value == -1)
            break;
        //Выводимая инфа
        switch (value) {
            case 'h':
                fprintf(stdout, "-h, --help: Display information about options\
\n-v, --version: Display information about programmes\n");
                exit(EXIT_SUCCESS);

            case 'v':
                fprintf(stdout, "Version 1.1\
\nSamaykin Egor Sergeevich\
\nN32491\
\n3\n");
                exit(EXIT_SUCCESS);

            case '?':
                if (Debug_FLAG)
                    fprintf(stderr, "Inputed unknown option: %c\
\nUsage: lab1lessN32491 [options] directory target\n",
                        optopt);
                exit(EXIT_FAILURE);

            default:
                break;
        }
    }

    if (argc - optind <= 1) {
        if (Debug_FLAG)
            fprintf(stderr, "Usage: lab1lessN32491 [options] directory target\n");
        exit(EXIT_FAILURE);
    }
    dir_search = argv[optind];
    targ_search = argv[optind + 1];
}

int main(int argc, char* const* argv)
{
    //(int argc, char **argv)
    //Проверка флага переменной окружения
    if (getenv("LAB11DEBUG")) {
        Debug_FLAG = 1;
    }
}

```

```
OptionDo(argc, argv);  
WalkDirectory();  
  
return 0;  
}
```

Листинг 1 – lab11essN32491.c