**Отчет по лабораторной работе №1**

**6.1. Цель работы**  
Получить навыки разработки приложений, реализующих операции с файлами средствами системных вызовов Linux API на языке C.

**6.2. Вариант задания**  
Удалить из текста заданный символ. Входные параметры: имя входного файла, имя выходного файла, заданный символ.

**6.3. Листинг программы**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

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

if (argc != 4) {

fprintf(stderr, "Usage: %s <input\_file> <output\_file> <char>\n", argv[0]);

return -1;

}

char \*input\_path = argv[1];

char \*output\_path = argv[2];

char \*char\_to\_remove = argv[3];

if (strlen(char\_to\_remove) != 1) {

fprintf(stderr, "Error: Third argument must be a single character.\n");

return -1;

}

char target = char\_to\_remove[0];

int input\_fd = open(input\_path, O\_RDONLY);

if (input\_fd == -1) {

perror("Error opening input file");

return -1;

}

int output\_fd = open(output\_path, O\_WRONLY | O\_CREAT | O\_TRUNC, 0644);

if (output\_fd == -1) {

perror("Error opening output file");

close(input\_fd);

return -1;

}

// Блокировка файлов (повышенная сложность)

struct flock lock;

lock.l\_type = F\_RDLCK;

lock.l\_whence = SEEK\_SET;

lock.l\_start = 0;

lock.l\_len = 0;

if (fcntl(input\_fd, F\_SETLK, &lock) == -1) {

perror("Error locking input file");

close(input\_fd);

close(output\_fd);

return -1;

}

lock.l\_type = F\_WRLCK;

if (fcntl(output\_fd, F\_SETLK, &lock) == -1) {

perror("Error locking output file");

close(input\_fd);

close(output\_fd);

return -1;

}

char input\_buf[1024];

char output\_buf[1024];

ssize\_t bytes\_read, bytes\_written;

int total\_removed = 0;

while ((bytes\_read = read(input\_fd, input\_buf, sizeof(input\_buf))) {

if (bytes\_read == -1) {

perror("Error reading input file");

close(input\_fd);

close(output\_fd);

return -1;

}

int output\_index = 0;

for (int i = 0; i < bytes\_read; i++) {

if (input\_buf[i] != target) {

output\_buf[output\_index++] = input\_buf[i];

} else {

total\_removed++;

}

}

bytes\_written = write(output\_fd, output\_buf, output\_index);

if (bytes\_written == -1) {

perror("Error writing to output file");

close(input\_fd);

close(output\_fd);

return -1;

}

if (bytes\_written != output\_index) {

fprintf(stderr, "Error: Incomplete write operation.\n");

close(input\_fd);

close(output\_fd);

return -1;

}

}

close(input\_fd);

close(output\_fd);

printf("Total removed characters: %d\n", total\_removed);

return total\_removed;

}

**Описание программы:**  
Программа обрабатывает аргументы командной строки, открывает файлы, блокирует их, читает входной файл, удаляет указанный символ и записывает результат в выходной файл. Обрабатываются ошибки открытия, чтения, записи и блокировки файлов. Возвращается количество удаленных символов или -1 при ошибке.