Домашнее задание 4. Мусаев Умахан Рашидович, БПИ234.

Программа реализована на 10 баллов.

Условие:

До 8 баллов

Разработать программу использующую для работы с **текстовыми** файлами только системные вызовы. Программа на языке С (или С++ в стиле С) должна прочитать, используя буфер, размер которого **превышает** читаемые файлы и записать прочитанный файл в файл с другим именем. Имена файлов для чтения и записи задавать с использованием **аргументов командной строки**.

Опционально +1 балл

Вместо большого буфера использовать для работы с файлами **буфер ограниченного размера**, требующий циклического использования как для чтения, так и для записи.

Опционально +1 балл

Читать и переписывать не только текстовые, но и исполняемые файлы (**текстовые и бинарные, содержащие признак исполнимости**), включая скрипты, которые сохраняют режим доступа исходных файлов, обеспечивающий их запуск. При этом обычные текстовые файлы запускаться не должны. Для них режим доступа должен оставаться прежним.

Необходимо предоставить только одну программу. В качестве отчета предоставить исходные тексты программы и ее краткое описание с указанием, на какую ожидаемую оценку реализован функционал.

Программа выполнена в виртуальной машине Ubuntu.

Инструкция по запуску:

1. **Прописываем в консоли:**

```
g++ -o dz4 dz4.cpp
```

2. **Запускаем программу, указав входной и выходной файл:**

```
./dz4 compute.cpp output.cpp
```

или

```
./dz4 input.txt output.txt
```

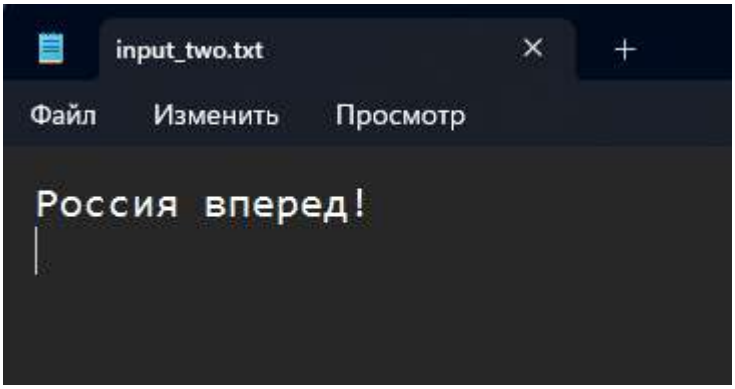
Программа работает как с текстовыми, так и с бинарными файлами, включая исполняемые скрипты и бинарники. Исполняемые права доступа сохраняются для выходного файла, если исходный файл был исполняемым. Для обычных текстовых файлов, которые не являются исполняемыми, исполняемые биты прав доступа удаляются в выходном файле.

```
// Читаем из входного файла и записываем в выходной файл в цикле
while ((bytes_read = read(fd_in, buffer, buffer_size)) > 0) {
    ssize_t bytes_written = 0;
    const char *write_ptr = buffer;

    // Гарантируем, что все прочитанные байты будут записаны
    while (bytes_written < bytes_read) {
        ssize_t result = write(fd_out, write_ptr + bytes_written, bytes_read - bytes_written);
        if (result < 0) {
            std::cerr << "Ошибка при записи в выходной файл '" << output_file << "': " << strerror(errno) << std::e>
            close(fd_in);
            close(fd_out);
            return EXIT_FAILURE;
        }
        bytes_written += result;
    }
}
```

Часть кода, отвечающая за чтение и запись данных. Внутри цикла while (bytes_written < bytes_read) **используется функция** write**, которая отвечает за запись данных из буфера в выходной файл. Переменные fd_out, write_ptr, bytes_written, bytes_read определяют файл и объем данных, которые необходимо записать. После успешной записи обновляется** bytes_written.

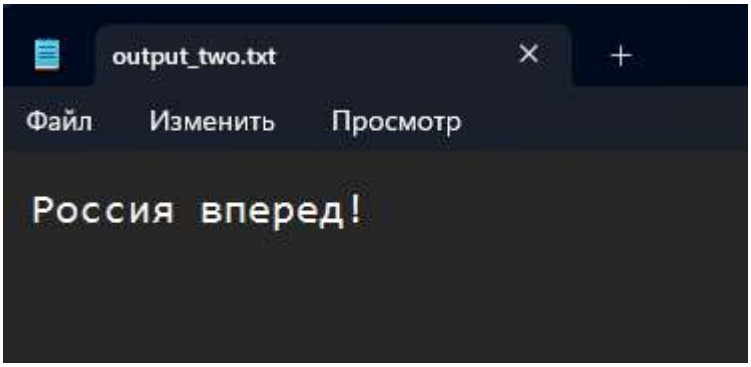
Примеры работы программ: Содержание файла input_two.txt:



Прописываем в консоли:

./dz4 input_two.txt output_two.txt

И создается файл output_two.txt **с аналогичным содержанием:**



Пример работы с исполняемым файлом:

Содержание файла предыдущего дз fibonnachifactorial.cpp:

Прописываем в консоли:

```
./dz4 fibonnachifactorial.cpp output_two.cpp
```

После чего в директиве появляется файл output_two.cpp **с аналогичным содержанием**: