

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

Институт искусственного интеллекта

Кафедра проблем управления

**КУРСОВАЯ РАБОТА**

по дисциплине **Основы программирования систем управления**

**Тема курсовой работы:** «Разработка программы для просмотра содержимое указанного файла в текстовом и в шестнадцатеричном виде»

**Студент группы:** Обухова А. М. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Руководитель курсовой работы:** Новоженин М. Б. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Работа представлена к защите: «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Допущен к защите«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г.



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

Институт искусственного интеллекта

Кафедра проблем управления

**ЗАДАНИЕ**

**на выполнение курсовой работы** по дисциплине

«Основы программирования систем управления»

|  |
| --- |
| Утверждаю  Заведующий кафедрой ПУ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Романов М.П.  Подпись  «\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г. |

Студент: Обухова А. М. Группа: КВБО-05-21

**Тема: «**Разработка программы для просмотра содержимого указанного файла в текстовом и в шестнадцатеричном виде»

**Исходные данные:**

1. Разработка программы, позволяющей просмотреть на экране монитора содержимое указанного файла в текстовом и в шестнадцатиричном виде (Hex Dump). Имя просматриваемого файла передается программе в качестве параметра при ее вызове. Пример вызова программы (если имя загрузочного модуля view.exe) для просмотра файла с именем “myfile.doc”: VIEW MYFILE.DOC Размер файла может быть любым (в том числе и превышающем 65535 байт).
2. Язык программирования C++;
3. Библиотека OpenCV;

**Перечень вопросов, подлежащих обработке, и обязательного графического материала:**

1. Определение формата файла
2. Преобразование текстовой информации файла в шестнадцатеричной вид(Hex Dump).
3. Получение доступа к файлу, его текстовой и шестнадцатиричной записи через командную строку

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc74934469)

1. [ОСНОВНАЯ ЧАСТЬ](#_Toc74934470) [6](http://)

2. [ПРОСТРАНСТВО ИМЕН И ЗАГОЛОВКИ](#_Toc74934471) [8](http://)

3. [ПРОЦЕСС РАБОТЫ](#_Toc74934471) [8](http://)

3.1. [FileExists() 8](#_Toc74934471)

3.2. [GetExt() 9](#_Toc74934471)

3.3. [main()](#_Toc74934471) [9](http://)

4. [ЗАКЛЮЧЕНИЕ 12](#_Toc74934472)

5. [СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 12](#_Toc74934473)

[ПРИЛОЖЕНИЕ А 13](#_Toc74934474)

# **ВВЕДЕНИЕ**

Данная курсовая работа представляет собой проект по разработке программы для получения доступа через командную строку к текстовой и шестнадцатеричной информации файлов загрузочного модуля Kursovaya.exe. Для этого была рассмотрена возможность работы как и с текстовыми файлами, так и графическими. В текстовом формате была предусмотрена поддержка русского языка.

Проект является консольным приложением и написан на языке программирования С++. Для реализации проекта была выбрана среда разработки Microsoft Visual Studio 2019, позволяющая с удобством работать с проектами языка С++, их созданием и отладкой.

В процессе разработки были задействованы сторонние библиотеки для расширения функционала и оптимизации работы, упрощая процесс разработки.

# **ОСНОВНАЯ ЧАСТЬ**

Для доступа к корню загрузочного модуля Kursovaya.exe была использована стандартная команда командной строки Windows: cd.

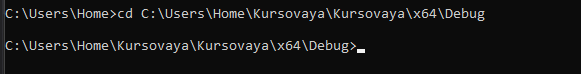


Рис. 1 Получение доступа к корневой папке

Для получения дополнительной информации о файлах в папке есть возможность воспользоваться командой dir.

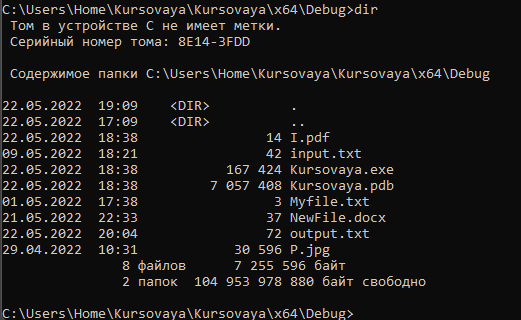


Рис. 2 Просмотр файлов в папке

Результат вызова программы с именем загрузочного модуля Kursovaya.exe для просмотра файла с именем “NewFile.docx” продемонстрирован на рисунке 3.

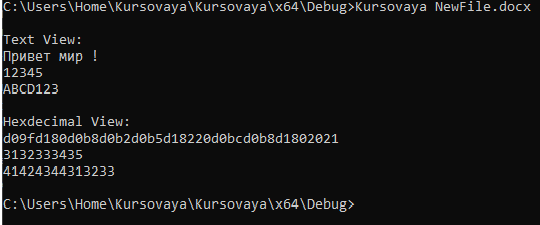


Рис. 3 Пример работы программы

Принципиальная блок-схема работы программы представлена на рисунке 4

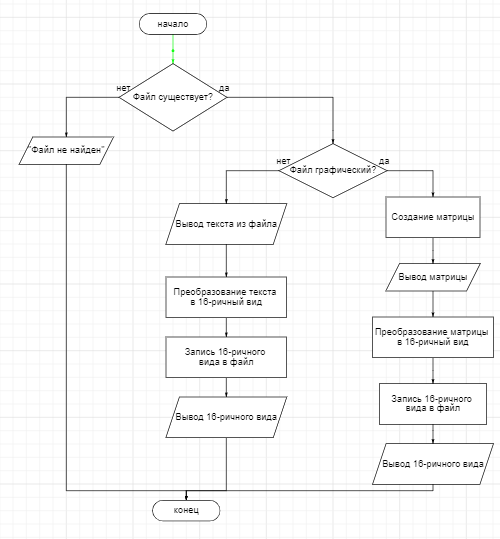


Рис. 4 Принципиальная блок-схема программы

В проекте присутствуют еще 2 функции, не включая main(). Каждая из них является небольшой подпрограммой, вынесенной для удобства, которая выполняет другие операции, необходимые для реализации задания.

В программе было подключено несколько заголовков:

* iostream

Заголовочный файл, включенный в стандартную библиотеку С++. Включает в себя классы, функции и переменные для организации ввода и вывода. Объекты данной библиотеки являются частью пространства имен standart(std).

* fstream

Используется для работы с потоками данных и файлами. Включает в себя 3 класса потоков: ofstream, ifstream, fstream.

* string

Позволяет свободно работать со строками в С++.

* iomanip

Это заголовочный файл ввода-вывода на языке C++. Это функция, которая используется для манипулирования операторами ввода-вывода.

* Windows.h.

Содержит в себе всё необходимое для работы с файлами, директориями и другими составляющими операционной системы Windows.

* io.h

Поиск файлов. Находит файлы по указанному пути, соответствующие маске. Маска - строка, критерий поиска, содержащая символы \* (любая последовательность любых символов) и (любой один символ) path - строка, в которой содержится комбинация пути и маски.

* opencv2/opencv.hpp

По-мимо заголовков стандартных библиотек С++ был подключен заголовок библиотекиOpenCV для работы с графическими файлами. Заголовок определяет как базовые объекты (например, классы) библиотеки OpenCV: изображение, цвет и др, так и функции: преобразование самих изображений и их цветов.Объекты библиотеки OpenCV являются частью пространства имен Computer Vision (cv).

**Пространство имен и заголовки**

#include <iostream>

#include <fstream>

#include <string>

#include<io.h>

#include<iomanip>

#include <Windows.h>

#include <opencv2/opencv.hpp>

using namespace cv;

using namespace std;

Вышеприведенный код находится в начале файла "Kursovaya.cpp" и отвечает за подключение заголовочных файлов и пространств имен.

**Процесс работы**

Как было сказано ранее, для выполнения задачи программа разделена на отдельные функции, а именно:

* FileExists()
* GetExt()
* main()

**FileExists()**

Это булева функция, которая проверяет наличие введенного имя файла в папке

bool FileExists(const char\* fname)

{

return \_access(fname, 0) != -1;

}

Функция принимает имя файла как аргумент, и вызывается через главную функцию.

Функция access() относится к UNIX-подобной файловой системе и не определяется стандартом ANSI С. Она используется для проверки существования файла. Функция access() возвращает 0, если проверяемый вид доступа разрешен; в противном случае она возвращает —1.

**GetExt()**

Данная string-функция отвечает за проверку формата файла.

string GetExt(const string& st) {

size\_t pos = st.rfind('.');

return st.substr(pos + 1, string::npos);

}

Функция принимает имя файла как аргумент, и вызывается через главную функцию.

При запуске выполняется команда rfind к аргументу. Эта команда осуществляет поиск по строке символа '.' в обратном направлении до первого вхождения подстроки, совпадающей с заданной последовательностью символов. Далее функция возвращает команду substr(S.substr (pos, count)), т.е. возвращает подстроку данной строки начиная с символа с индексом pos до конца строки, т.к. pos + count > S.size ().

Таким образом мы получаем строку формата файла.

**main()**

Данная функция добавляется автоматически при создании нового проекта. Именно она запускается после компиляции программы.

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

{

setlocale(0, ".utf8");

if (FileExists(argv[1]) == false) {

cout << endl << "File not found!" << endl;

return 0;

}

string getcontent;

if (GetExt(argv[1]) == "jpg" || GetExt(argv[1]) == "png" || GetExt(argv[1]) == "jpeg" || GetExt(argv[1]) == "bmp" || GetExt(argv[1]) == "gif") {

Mat3d S = imread(argv[1]);

cout << endl << "Text View:" << endl;

cout << S;

cout << endl << "Hexdecimal View:" << endl;

for (int r = 0; r < S.rows; ++r) {

for (int c = 0; c < S.cols; ++c) {

const Vec3b& v = S(r, c);

сout << hex << uppercase << setw(2) << setfill('0') << int(v[0]);

сout << hex << uppercase << setw(2) << setfill('0') << int(v[1]);

сout << hex << uppercase << setw(2) << setfill('0') << int(v[2]);

}

}

}

else {

ifstream fin(argv[1]);

ofstream fout("output.txt");

ifstream fout\_in("output.txt");

fout << hex << setfill('0');

cout << endl << "Text View:" << endl;

while (getline(fin, getcontent))

{

for (int i = 0; i < getcontent.length(); i++)

{

fout << (unsigned int)(unsigned char)getcontent[i];

}

fout << endl;

cout << getcontent << endl;

}

cout << endl << "Hexdecimal View:" << endl;

while (getline(fout\_in, getcontent))

{

cout << getcontent << endl;

}

}

return 0;

}

На вход функции main поступают следующие аргументы:

Первый параметр argc определяет количество параметров, передаваемых в функцию main из командной строки. Второй параметр argv — указатель на массив указателей типа char (массив строк). Каждый элемент массива ссылается на отдельный параметр командной строки. При стандартном запуске программы argc равно 1, argv — массив из одного элемента, этим элементом является имя запускаемого файла.

В нашем случае argc равно 2, argv — массив из двух элементов: имя запускаемого файла и имя файла для обработки.

Setlocale(0, ".utf8") меняет кодировку текста файла. Это необходимо для работы с русским текстом.

После запускается функция проверки существования файла. Если файла не существует, то программа выводит "File not found!" и завершается кодом 0.

Если файл существует, то далее используется hex - манипулятор, для того чтобы переводить всё символы, которые идут после него в шестнадцатиричную систему исчисления. Манипулятор setfill()принимает в качестве аргумента один символ, который замещает все пробелы на незаполненных позициях поля.

После идет разветвлене на графический и текстовый файл при помощи функции GetExt().

Если файл графический, то создается матрица параметров пикселей Mat3d S с помощью команды imread. После вывода на экран, идет преобразование матрицы по строкам и столбцам в шестнадцатеричную систему по 3 элементам цвета пикселя(RGB). Полученное значение выводится на экран для пользователя для просмотра.

Если файл не графический, то создается текстовый файл output.txt для дальнейшей записи шестнадцатиричного вида файла, идет побайтовое считывание символов(getline) и вывод на экран. Символ сразу преобразуется в шестнадцатеричную форму  
и записывается в текстовый файл output.txt. Далее полученный файл выводится на экран для пользователя для просмотра.

**ЗАКЛЮЧЕНИЕ**

Во время выполнения курсовой работы были рассмотрены и изучены следующие вопросы:

* Определение формата файла
* Преобразование текстовой информации файла в шестнадцатеричной вид(Hex Dump).
* Получение доступа к файлу, его текстовой и шестнадцатиричной записи через командную строку

Также была написана программа для просмотра содержимого указанного файла в текстовом и в шестнадцатеричном виде через командную строку.

Процесс разработки произведен на языке программирования С++. Для реализации проекта была выбрана среда разработки Microsoft Visual Studio 2019. Также была задействована сторонняя библиотека OpenCV.

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

* <https://docs.microsoft.com/ru-ru/cpp/>

# 

# **ПРИЛОЖЕНИЕ А**

Листинг исходного кода программы Kursovaya.cpp

#include <iostream>

#include <fstream>

#include <string>

#include<io.h>

#include<iomanip>

#include <Windows.h>

#include <opencv2/opencv.hpp>

using namespace cv;

using namespace std;

string GetExt(const string& st) {

size\_t pos = st.rfind('.');

return st.substr(pos + 1, string::npos);

}

bool FileExists(const char\* fname) //Проверка на существование файла

{

return \_access(fname, 0) != -1;

}

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

{

setlocale(0, ".utf8");

if (FileExists(argv[1]) == false) { //Проверка на существование файла

cout << endl << "File not found!" << endl;

return 0;

}

string getcontent;

if (GetExt(argv[1]) == "jpg" || GetExt(argv[1]) == "png" || GetExt(argv[1]) == "jpeg" || GetExt(argv[1]) == "bmp" || GetExt(argv[1]) == "gif") { //Для графики

Mat3d S = imread(argv[1]);

cout << endl << "Text View:" << endl;

cout << S;

cout << endl << "Hexdecimal View:" << endl;

for (int r = 0; r < S.rows; ++r) {

for (int c = 0; c < S.cols; ++c) {

const Vec3b& v = S(r, c);

сout << hex << uppercase << setw(2) << setfill('0') << int(v[0]);

сout << hex << uppercase << setw(2) << setfill('0') << int(v[1]);

сout << hex << uppercase << setw(2) << setfill('0') << int(v[2]);

}

}

}

else { //Для текста

ifstream fin(argv[1]);

ofstream fout("output.txt");

ifstream fout\_in("output.txt");

fout << hex << setfill('0');

cout << endl << "Text View:" << endl;

while (getline(fin, getcontent))

{

for (int i = 0; i < getcontent.length(); i++)

{

fout << (unsigned int)(unsigned char)getcontent[i];

}

fout << endl;

cout << getcontent << endl;

}

cout << endl << "Hexdecimal View:" << endl;

while (getline(fout\_in, getcontent))

{

cout << getcontent << endl;

}

}

return 0;

}