# 

Частное учреждение образования

«Колледж бизнеса и права»

ПРОГРАММА ДЛЯ ОБРАБОТКИ BMP-ФАЙЛОВ С 24-ХБИТНОЙ ГЛУБИНОЙ ЦВЕТА ФИЛЬТРАМИ «СЕПИЯ», «НЕГАТИВ», ДОБАВЛЕНИЕМ В ИЗОБРАЖЕНИЕ «ПОВРЕЖДЕННЫХ» ПИКСЕЛЕЙ, ИСПРАВЛЕНИЕМ В ИЗОБРАЖЕНИИ «ПОВРЕЖДЕННЫХ» ПИКСЕЛЕЙ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

«Основы алгоритмизации и программирование»

КП Т.992002.401 ПЗ

Руководитель проекта ( Ю.В. Шаляпин )

Учащийся (М.М. Быковский)

2021

Содержание

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | *КП Т.992002.401ПЗ* | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № докум | Подпись | Дата |
| Разраб | | Быковский М.М. |  |  | *ПРОГРАММА ДЛЯ ОБРАБОТКИ BMP-ФАЙЛОВ ФИЛЬТРАМИ «СЕПИЯ» «НЕГАТИВ», ДОБАВЛЕНИЕМ В ИЗОБРАЖЕНИЕ «ПОВРЕЖДЕННЫХ» ПИКСЕЛЕЙ, ИСПРАВЛЕНИЕМ В ИЗОБРАЖЕНИИ «ПОВРЕЖДЕННЫХ* | Литера | | | Лист | Листов |
| Пров | | Шаляпин Ю.В. |  |  |  | y |  | 3 | 59 |
| Н. Контр. | |  |  |  | КБиП | | | | |
| Утв | |  |  |  |
|  | |  |  |  |

Введение 3

1 Постановка задачи 5

1.1 Сущность задачи 5

1.2 Проектирование модели 5

2 Вычислительная система 6

2.1 Требования к аппаратным и операционным ресурсам 6

2.2 Инструменты разработки 6

3 Проектирование задачи 7

3.1 Требования к приложению 7

3.2 Организация данных 7

3.3 Функции:Логическая и физическая организация и элементы управления 9

3.4 Проектирование справочной системы приложения 11

4 Описание программного средства 13

4.1 Общие сведения 13

4.2 Функциональное назначение 13

4.3 Входные данные 13

4.4 Выходные данные 13

5 Методика испытаний 14

5.1 Технические требования 14

5.2 Порядок проведения испытаний 14

5.2.1 Функциональное тестирование 14

6 Применение 21

6.1 Назначение программы 21

6.2 Условия применения 21

Заключение 22

Список информационных источников 23

Приложение А. Код программы 24

Введение

Целью разработки проекта на тему «Программа для обработки BMP-файлов с 24-хбитной глубиной цвета фильтрами «сепия», «негатив», добавлением в изображение «поврежденных» пикселей, исправлением в изображении «поврежденных» пикселей» является создание про­граммного средства, которое предоставит пользователям возможность работы и обработки bmp-изображений с помощью фильтров «сепия», «негатив», а так же добавления в изображение «поврежденных» пикселей и их исправление.

Программа состоит из пяти MS Windows форм.

Пояснительная записка состоит из шести разделов, содержащих необходимую информа­цию по организации эксплуатации программного приложения.

В первом разделе «Постановка задачи» описывается организационная сущность задачи, предметная область и круг задач, которые должны быть автоматизированы. Описывается за­дача, перечисляются основные функции программы. Строится информационная модель, отра­жающая сущности задачи, их свойства и взаимосвязи.

Во втором разделе «Вычислительная система» описываются требования к разрабатывае­мому программному приложению, описание структуры приложения, проектирование модели и описание средств разработки.

В третьем разделе «Проектирование задачи» описываются используемые функции и лис­тинг этих функций.

В четвертом разделе «Описание программного средства» описываются общие сведения приложения, функциональное назначение и описание главного экрана.

В пятом разделе описывается «Методика испытаний» описываются технические требо­вания приложения, а также функциональное тестирование с проверкой функций приложения.

В шестом разделе «Применение» описывается назначение программы и условия её при­менения. В этом разделе приводится структура справочной системы, а также методика ее ис­пользования.

В заключении описывается цель создания программного приложения, какие функции и задачи были реализованы.

В приложении А содержится код программы.

Для редактирования BMP изображений можно применить почти любой графический ре­дактор, но я выбрал один, достаточно похожий, удобный и функциональный редактор, подхо­дящий для домашнего использования. Universal Viewer – универсальное средство для про­смотра файлов разных форматов, включая текст, изображения, видео, аудио, веб-страницы, а также шестнадцатеричные и двоичные коды. Говоря про обработку bmp-изображений, функ­ционал моей программы не сильно отличается от функционала программы-аналога. В Universal Viewer присутствует ряд дополнительных функций: Поворот на 90 градусов в обе стороны и поворот на 180 градусов, отражение по вертикали и по горизонтали, зум изображения, обра­ботка изображения со всеми вариантами глубины цвета и маленький пакет инструментов. Из явных плюсов моей программы – повреждение и исправление изображений. Хотелось бы доба­вить, что при необходимости большинство функций из Universal Viewer можно воссоздать в моей программе.

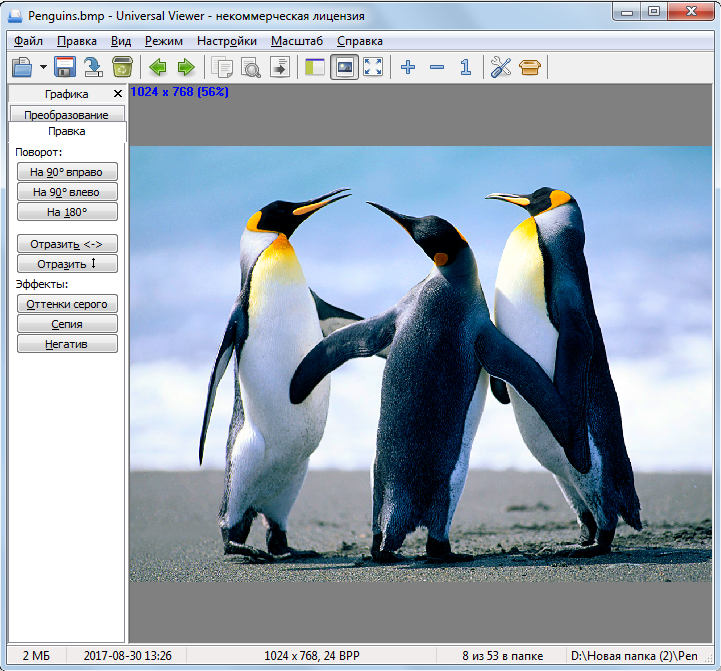


Рисунок 1 – Функционал работы с BMP-изображениями в программе-аналоге Universal Viewer

1. Постановка задачи

# Сущность задачи

Цель состоит в реализации программного средства, которое предоставит пользователям возможность обрабатывать bmp-изображения с 24-хбитной глубиной цвета.

Для достижения поставленной цели требуется реализовать следующие задачи:

* авторизация и аутентификация пользователей (администратор, пользователь), локализа­ция на английском, русском и белорусском языках;
* возможность выбрать цвет фона окна приложения;
* реализовать фильтр «сепия»;
* реализовать фильтр «негатив»;
* реализовать рандомное расположение на изображении «испорченных» пикселей, контра­стно отличающихся от цвета окружающих пикселей (черные, белые);
* программное исправление «поврежденных» (сильно отличающихся от окружающих) пик­селей на усредненный цвет окружающих пикселей;
* обеспечить исправление пикселей как в центре изображения, так и по его краям и уг­лам.

# Проектирование модели

При проектировании был организован поэтапный план в виде основных аспектов реали­зации: подготовки к проектированию и сбор необходимой информации, установка используе­мого программного средства, создание и подключение библиотек для упрощения реализации, проектирование базовой модели для первичного тестирования, обновление и дополнение суще­ствующих компонентов, многоступенчатое тестирование продукта и его оптимизация.

# Вычислительная система

# Требования к аппаратным и операционным ресурсам

Во время разработки приложение было оптимизировано. Использование динамической памяти совместно с современной системой управления ресурсами привело к снижению потреб­ления оперативной памяти и позволило приложению беспроблемно функционировать при ис­пользовании на устаревшей аппаратной базе. Для работы приложения компьютер пользователя должен удовлетворять следующим минимальным требованиям:

* место на диске: 470 Кбайт;
* оперативная память: 40 Мбайт;
* процессорcтактовой частотой не менее 1 ГГц;
* операционная система: MS Windows 7 или новее;
* .NET Framework 4.0 или новее.

# Инструменты разработки

Для разработки программного средства было принято решение использовать следующие инструменты:

* операционная система MS Windows 7 x64;
* язык программирования C++/CLI;
* среда программирования MS Visual Studio 2019.

MS Windows 7 – операционная система семейства Windows NT, следующая за Window Vista. В линейке Windows NT система имеет номер версии 6.1 (Windows 2000 – 5.0, Windows XP – 5.1, Windows Server 2003 – 5.2, Windows Vista и Windows Server 2008 – 6.0). Серверной версией является Windows Server 2008 R2, версией для интегрированных систем (построенных из компонентов Windows) – Windows Embedded Standard 2011 (Quebec), мобильной – Win­dows Embedded Compact 2011 (Chelan, Windows CE 7.0).

Операционная система поступила в продажу 22 октября 2009 года, меньше, чем через три года после выпуска предыдущей операционной системы, Windows Vista. Партнёрам и кли­ентам, обладающим лицензией Volume Licensing доступ к RTM был предоставлен 24 июля 2009 года. В интернете оригинальные установочные образы финальной версии системы были дос­тупны с 21 июля 2009 года.

Язык программирования C++, на сегодняшний день, является одним из самых высоко­скоростных языков программирования. Благодаря CLI, C++ поддерживает разработку оконных приложений. Он обеспечивает отличную скорость выполнения написанных программ, что не­обходимо для нашего программного решения.

Среда программирования Visual Studio 2019 Community также является продуктом ком­пании Microsoft. Она обеспечивает комфортную разработку, тестирование и компиляцию при­кладных программ.

# Проектирование задачи

# Требования к приложению

Программное средство должно удовлетворять следующему комплексу требований:

* быстрота выполнения действий;
* простота в использовании (понятный интерфейс);
* целостность данных;
* многократное использование данных;
* уменьшение излишней избыточности данных;
* средства защиты данных (авторизация).

# Организация данных

Таблица 1 – Элементы управления и визуализации

|  |  |  |  |
| --- | --- | --- | --- |
| Размещение | Элемент экрана | Класс | Действие |
| Окно авторизации | Цвет | Button | Выбор цвета для всех форм |
| Перезагрузка | Button | Перезагрузка приложения |
| Создать новую очередь | Button | Переход на форму «Создание очереди» |
| Выход | Button | Выход из программы |
| Вход | Button | Переход на главную форму |
| Регистрация | Button | Переход на форму регистрации |
| Русский | Button | Устанавливает язык приложения Русский |
| Английский | Button | Устанавливает язык приложения Английский |
| Белорусский | Button | Устанавливает язык приложения Белорусский |
| textBox1 | TextBox | Для ввода Логина пользователем |
| textBox2 | TextBox | Для ввода Логина пользователем |
| checkBox1 | ChekBox | Для видимости / не видимости пароля |
| Окно регистрации | Назад | Button | Возвращает на окно Авторизации |
| Зарегистрироваться | Button | Регистрирует пользователя |
| textBox1 | TextBox | Для ввода Логина пользователем |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
|  | checkBox1 | ChekBox | Для видимости / не видимости пароля |
| Окно администрации | Получить пользователей | Button | Выдает список зарегистрированных пользователей |
| Сохранить измененную информацию | Button | Сохраняет измененную администратором информацию о данных пользователей |
| Назад | Button | Возвращает на окно авторизации |
| Окно обработки изображения | Перезагрузка | Button | Полностью перезагружает приложение(первое окно после перезагрузки будет Авторизация) |
| Открыть фото | Button, OpenFileDialog | Позволяет выбрать пользователю любое доступное изображения из проводника. |
| Закрыть фото | Button | Удаляет фото из pictureBox |
| textBox1 | TextBox | Для ввода значения  «От» |
| textBox2 | TextBox | Для ввода значения  «До» |
| Испортить черными пикселями | Button | «Портит» изображение черными пикселями |
| Испортить белыми пикселями | Button | «Портит» изображение белыми пикселями |
| Исправить черные пиксели | Button | Исправляет «Грязное» изображение |
| Исправить белые пиксели | Button | Исправляет «Грязное» изображение |
| Негатив | Button | Применяет фильтр «Негатив» на изображение |
| Сепия | Button | Применяет фильтр «Сепия» на изображение |
| Открыть результат | Button | Позволяет выбрать пользователю обработанное изображение по казанному ранее пути |
| < | Button | Перемещение влево в выбранной директории |
| > | Button | Перемещение влево в выбранной директории |
| Закрыть фото | Button | Перемещение вправо в выбранной директории |

# Функции: логическая и физическая организация и элементы управления

Таблица 2 – Функции и закрепленные за ними элементы управления

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Компонента меню | Название элемента  интерфейса | Реализация |
| Отображение пользователей | GetUsers\_Click  () | Щелчок мыши по кнопке «Получить пользователей» | Приложение A, страница 28 |
| Сохранений измененной информации | SaveNewInfo\_Click() | Щелчок мыши по кнопке «Сохранить информацию» | Приложение A, страница 28 |
| Возврат на окно авторизации | Back\_Click() | Щелчок мыши по кнопке «Назад»/Загрузка формы «Авторизация» | Приложение A, страница 28 |
| Переход на окно БМП, если логин и пароль верны | Enter\_Click() | Щелчок мыши по кнопке «Вход»/Загрузка формы БМП | Приложение A, страница 28 |
| Выход из приложения | Exit\_Click () | Щелчок мыши по кнопке «Выход» | Приложение A, страница 30 |
| Переход на окно регистрации | Registration\_Click () | Щелчок мыши по кнопке «Регистрация»/Загрузка формы «Регистрация» | Приложение A, страница 30 |
| Открытие/  Сокрытие пароля символом | checkBox1\_CheckedChanged() | Щелчок мыши по элементу chekBox1 | Приложение A, страница 30 |
| Установка русского языка | RusLang\_Click  () | Щелчок мыши по кнопке «Русский» | Приложение A, страница 31 |
| Установка английского языка | EngLang\_Click  () | Щелчок мыши по кнопке «Английский» | Приложение A, страница 31 |
| Установка белорусского языка | BelLang\_Click  () | Щелчок мыши по кнопке «Белорусский» | Приложение A, страница 31 |
| Перезагрузка приложения | Restart\_Click() | Щелчок мыши по кнопке «Перезагрузка» | Приложение A, страница 31 |
| Выбор цвета окон | Color\_Click() | Щелчок мыши по кнопке «Цвет» | Приложение A, страница 31 |
| Перезагрузка приложения | Reset\_All\_Click() | Щелчок мыши по кнопке «Перезагрузка» | Приложение A, страница 32 |

Продолжение таблицы 2

|  |  |  |  |
| --- | --- | --- | --- |
| Удаление изображения | Close\_Click() | Щелчок мыши по кнопке «Закрыть фото» | Приложение A, страница 33 |
| Выбор изображения из проводника | Open\_Click() | Щелчок мыши по кнопке «Открыть фото» | Приложение A, страница 32 |
| Выход из приложения | выходToolStripMenuItem\_Click() | Щелчок мыши по кнопке «Выход» | Приложение A, страница 32 |
| Загрязнение изображения черными пикселями | Shit\_Click() | Щелчок мыши по кнопке «Испортить черными пикселями» | Приложение A, страница 33 |
| Загрязнение изображения белыми пикселями | Shit\_Click1() | Щелчок мыши по кнопке «Испортить белыми пикселями» | Приложение A, страница 35 |
| Исправления загрязненного изображения черными пикселями | Fix\_Click() | Щелчок мыши по кнопке «Исправить черные пиксели» | Приложение A, страница 36 |
| Исправления загрязненного изображения белыми пикселями | Fix1\_Click() | Щелчок мыши по кнопке «Исправить белые пиксели» | Приложение A, страница 36 |
| Выбор обработанного изображения из проводника | Open\_Res\_Click() | Щелчок мыши по кнопке «Открыть результат» | Приложение A, страница 39 |
| Применение фильтра негатив на изображение | Negativ\_Click() | Щелчок мыши по кнопке «Негатив» | Приложение A, страница 38 |
| Применение фильра сепия на изображение | Sepia\_Click() | Щелчок мыши по кнопке «Сепия» | Приложение A, страница 37 |
| Отображения окна с инструкцией | инструкцияToolStripMenuItem\_Click() | Щелчок мыши по кнопке «Инструкция»/Загрузка формы «Инструкция» | Приложение A, страница 32 |
| Отображение изображений по порядку в направлении вправо | ToRight\_Click() | Щелчок мыши по кнопке «>» | Приложение A, страница 39 |

# Продолжение таблицы 2

|  |  |  |  |
| --- | --- | --- | --- |
| Отображение изображений по порядку в направлении влево | ToBack\_Click() | Щелчок мыши по кнопке «<» | Приложение A, страница 39 |
| Удаление изображения | Close1\_Click() | Щелчок мыши по кнопке «Закрыть фото» | Приложение A, страница 32 |
| Отображение информации о программе | информацияToolStripMenuItem\_Click() | Щелчок мыши по кнопке «Информация» | Приложение A, страница 32 |
| Регистрирует нового пользователя | button1\_Click() | Щелчок мыши по кнопке «Зарегистрироваться» | Приложение A, страница 41 |

# Проектирование справочной системы приложения

Справочная система приложения должна быть реализована в виде дополнительного окна помощи пользователю, с указанием о том, как работать с главной формой.

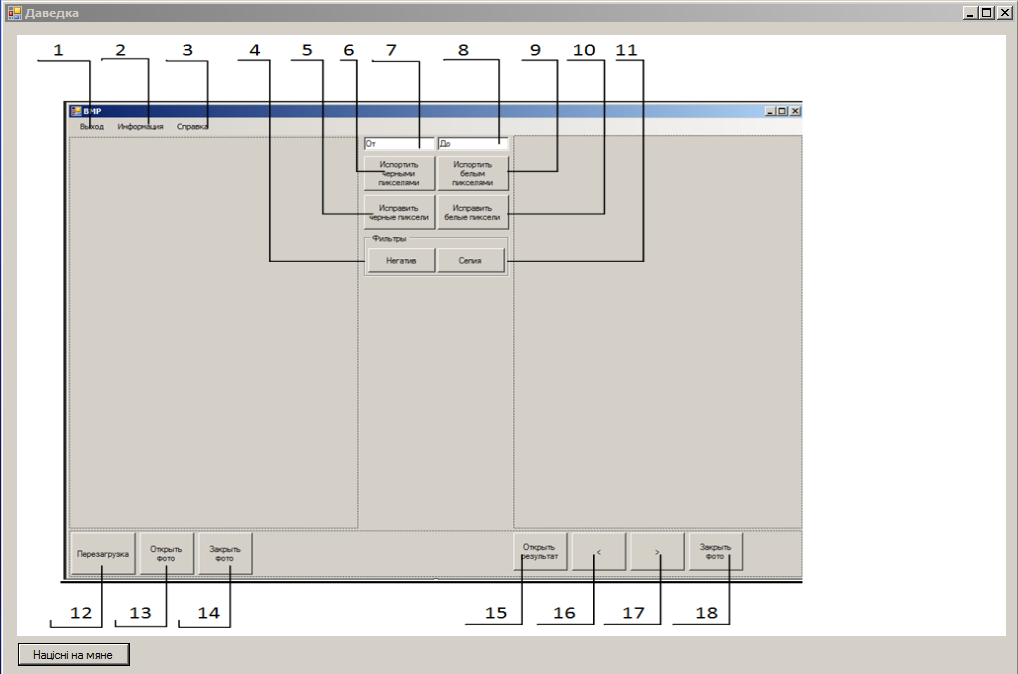


Рисунок 2 – Окно справки

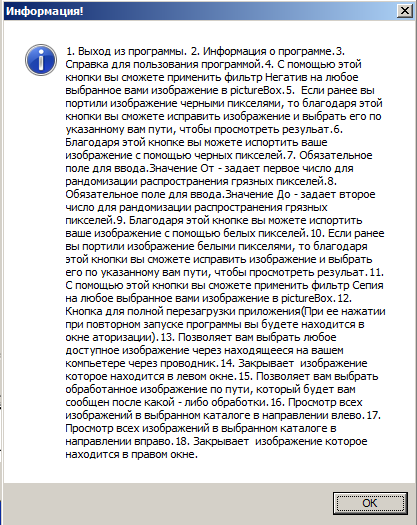


Рисунок 3 – Окно с пояснением элементов интерфейса

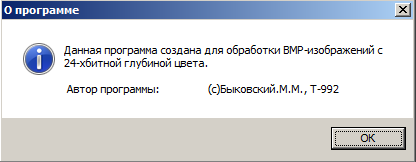


Рисунок 4 – Окно «О программе»

# Описание программного средства

# Общие сведения

На электронном носителе (диске) представлен установочник программного средства (Queue.exe), при запуске будет предоставлен выбор директории установки приложения.

Используемые библиотеки С++:

* <iostream> – объектно-ориентированная иерархия классов, где используется и множест­венное, и виртуальное наследование;
* <Windows.h> – содержит заявления для всех функций в Windows API, все общие мак­росы, которые используются программистами окон, и все типы данных, используемых различ­ными функциями и подсистем;
* <msclr/marshal.h> – для конвертирования из String^ в const char\*;
* <fstream> – предоставляет функционал для считывания данных из файла и для записи в файл;
* sys/types.h – определяет набор символов и структур typedef;
* sys/stat.h – содержит объявление функций и требуемых macros, struct и т. д., используе­мых для управления каталогами файловой системы.

# Функциональное назначение

Представлено в п. 1.1.

# Входные данные

Таблица 3 – Структура входных данных

|  |  |  |  |
| --- | --- | --- | --- |
| Входные данные | Тип данных | Размер, байт | Описание |
| Login | String^ | – | Логин пользователя |
| Password | String^ | – | Пароль пользователя |
| adminPassword | String^ | – | Пароль администратора |
| adminLogin | String^ | – | Логин администратора |
| Filename | Const char\* | 1 | Путь к файлу, который нужно обработать |
| Image | BMP | >=3 | Изображение для обработки |
| Value From | Static int | 4 | Значение ‹‹От›› для рандомного расположения ‹‹грязных пикселей›› |
| Value To | Static int | 4 | Значение ‹‹До›› для рандомного расположения ‹‹грязных пикселей›› |

4.4 Выходные данные

Таблица 4 – Структура выходных данных

|  |  |  |  |
| --- | --- | --- | --- |
| Выходные данные | Тип данных | Размер,байт | Описание |
| Image | BMP | >=3 | Обработанное изображение |
| Filename | String^ | – | Путь к обработанному файлу |

# Методика испытаний

# Технические требования

Минимальные системные требования:

* процессор с частотой не менее 2.1 ГГц;
* ОЗУ не менее 58 МБайт;
* память на диске в размере 1.5 МБайт;
* MS Windows 7 x32;
* оптический привод.

Рекомендуемые системные требования:

* процессор с частотой не менее 2.4 ГГц;
* ОЗУ не менее 75 Мбайт;
* память на диске в размере не менее 2 Мбайт;
* MS Windows 10 x64;
* оптический привод.

# Порядок проведения испытаний

# Функциональное тестирование

При создании приложения важную роль играют формы, так как они являются основным диалоговым средством работы пользователя. Разрабатываемое приложение будет состоять из 5 форм, которые будут появляться/пропадать по нажатию определенных кнопок, 5 файлов (.cpp), в которых будут описаны функции работы элементов, 4 файла заголовков (.h), в которых будут хранится функции, 3 файла ресурсов сборки (.resx), в которых будут описаны строки для лока­лизации приложения на заданных языках, 1 файл конфигурации (.config) для стабильной работы локализации.

При запуске программы будет открываться форма авторизации. На форме авторизации имеется: 3 элемента label (Авторизация, Логин, Пароль), 2 элемента textBox, 1 элемент group­Box, 1 элемент chekBox, 8 элементов button: ‹‹Цвет›› – для установки цвета через ColorDialog на все формы, ‹‹Перезагрузка›› – для перезагрузки приложения, ‹‹Выход›› – для выхода из прило­жения, ‹‹Вход›› – для входа в основную форму приложения, ‹‹Регистрация›› – для регистрации нового пользователя, ‹‹Русский›› – для установки русского языка, ‹‹Английский›› – для уста­новки английского языка, ‹‹Белорусский›› – для установки белорусского языка.

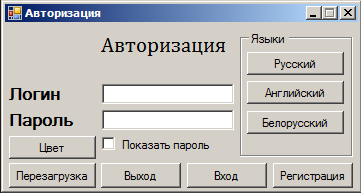


Рисунок 5 – Форма авторизации

Если выбрать кнопку ‹‹Регистрация›› то форма авторизации заменится формой регист­рации, где пользователь должен будет зарегистрировать нового пользователя, в противном слу­чае – вернуться на форму авторизации или выйти из программы. На форме регистрации у нас находится 3 элемента label (Авторизация, Логин, Пароль), 2 элемента textBox, 1 элемент chek­Box, 2 элемента button: ‹‹Назад›› – для возврата на форму авторизации, ‹‹Зарегистрироваться›› – для регистрации нового пользователя.

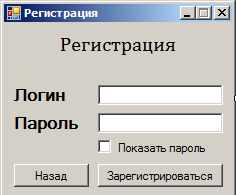


Рисунок 6 – Форма регистрации

Если верно ввести логин и пароль, затем выбрать кнопку ‹‹Вход›› на форме Авториза­ции, то данная форма заменится главной формой (BMP), где пользователю будет доступен весь функционал для работы с bmp-изображениям. На BMP-форме у нас находится 1 элемент me­nuStrip, 1 элемент openFiledialog, 1 элемент panel, 1 элемент groupBox, 2 элемента textBox, 2 элемента pictrueBox 13 элементов button: ‹‹Перезагрузка›› – для полной перезагрузки приложе­ния (при повторном запуске первой формой будет открыта форма регистрации), ‹‹Открыть фото›› – для выбора изображения из проводника компьютера, ‹‹Закрыть фото›› – для удаления фото из pictureBox1, ‹‹Испортить черными пикселями›› – для добавления черных ‹‹испорчен­ных›› пикселей в диапазоне чисел введенных пользователем, ‹‹Испортить белыми пикселями›› – для добавления белых ‹‹испорченных›› пикселей в диапазоне чисел введенных пользователем, ‹‹Исправить черные пиксели›› – для исправления черных ‹‹поврежденных›› пикселей на изо­бражении, ‹‹Исправить белые пиксели›› – для исправления белых ‹‹поврежденных›› пикселей на изображении, ‹‹негатив›› – для применения фильтра ‹‹негатив›› на изображение, ‹‹сепия›› – для применения фильтра ‹‹сепия›› на изображение, ‹‹Открыть результат›› – для выбора обрабо­танного фото по указанному пути, ‹‹<›› и ‹‹>›› – для просмотра всех изображений по порядку в выбранной папке, ‹‹Закрыть фото›› – для удаления фото из pictureBox2.

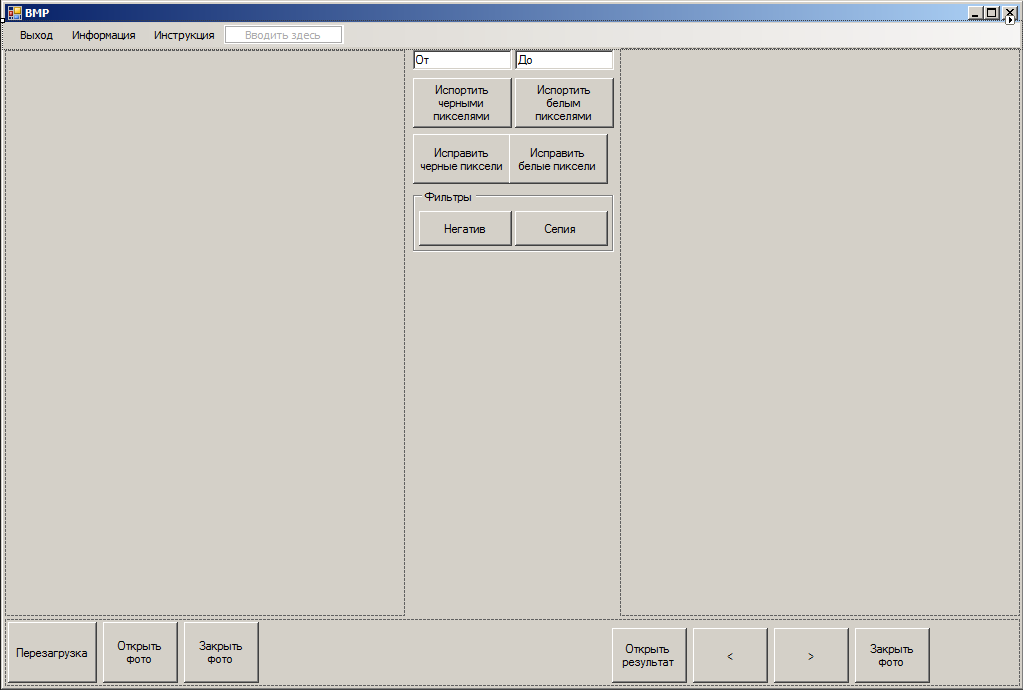


Рисунок 7 – Главная форма (BMP)

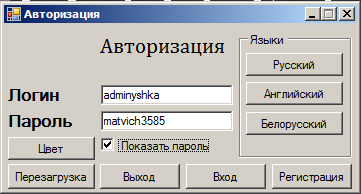


Рисунок 8 – Форма авторизации с введенными данными для перехода в форме Администратора.

Когда вы ввели верный пароль и логин для формы администратора мы попадём в меню администрации где мы сможем добавлять, удалять и просматривать информацию о зарегистри­рованных пользователей (Логин и Пароль). На форме Администратора у нас находится 1 эле­мент richtextBox, 3 элемента button: ‹‹Получить пользователей›› – для отображения всех заре­гистрированных пользователей, ‹‹Сохранить измененную информацию›› – для перезаписи в файл с информацией о пользователях измененных данных, ‹‹Назад›› – для возврата на форму Авторизации.

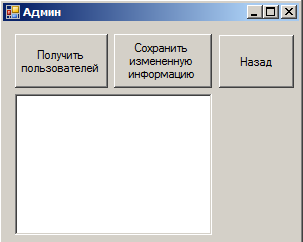


Рисунок 9 – Кабинет администрации

Тестирование всего программного средства происходит по средствам проверки всех элементов меню, представленных в программном средстве. В ходе тестирования была прове­рена работоспособность всех пунктов меню, которые содержатся в программе, а также запись в файл.

В случае попытки пользователя войти в программу без ввода логина и пароля будет вы­дана ошибка:

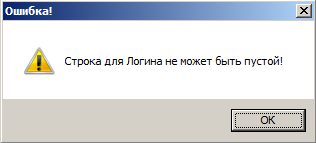


Рисунок 10 – Уведомление об ошибке связанной с пустотой поля для логина

В случае попытки пользователя войти в программу без ввода пароля будет выдана ошибка:

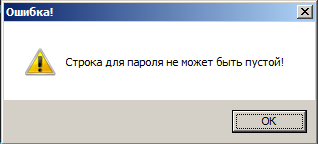


Рисунок 11 – Уведомление об ошибке связанной с пустотой поля для пароля

В случае неверного ввода пароля или логина будет выдано уведомление:

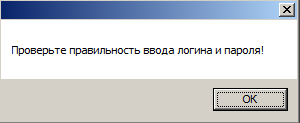


Рисунок 12 – Уведомление об перепроверке правильности введенных данных

В случае верного ввода пароля и логина будет выдано уведомление:

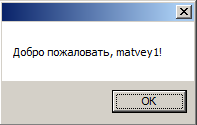


Рисунок 13 – Уведомление при входе в акканут

В случае попытки пользователя зарегистрировать аккаунт без ввода логина будет выдана ошибка:

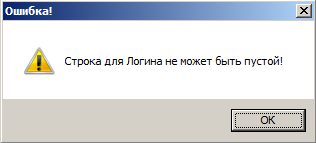


Рисунок 14 – Уведомление об ошибке

В случае попытки пользователя зарегистрировать аккаунт без ввода пароля будет выдана ошибка:

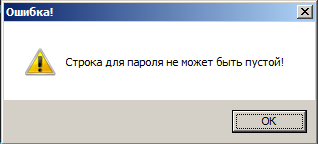


Рисунок 15 – Уведомление об ошибке

В случае ввода доступного пароля и логина будет выдано уведомление о регистрации:

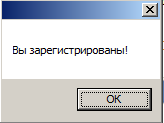


Рисунок 16 – Уведомление об успешной регистрации

В случае попытки ‹‹испортить›› изображение без ввода значений ‹‹от›› и ‹‹до›› будет выдана ошибка:

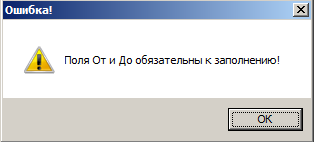


Рисунок 17 – Уведомление об ошибке связанной с пустотой полей

В случае попытки ‹‹испортить›› изображение без выбора изображения будет выдана ошибка:

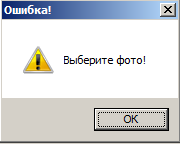


Рисунок 18 – Уведомление об ошибке связанной с отсутствием изображения

В случае попытки обработки изображения с неверным расширением будет выдана ошибка:

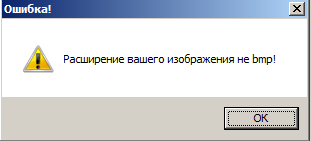


Рисунок 19 – Уведомление об ошибке связанной с неверно выбранным расширением

В случае попытки обработки изображения с неверным форматом будет выдана ошибка:

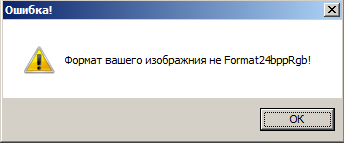


Рисунок 20 – Уведомление об ошибке связанной с неверно выбранным форматом

В случае попытки исправления изображения которое не было испорчено будет выдана ошибка:

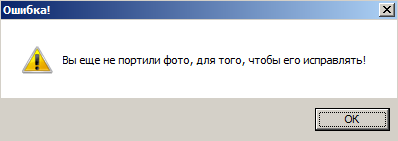


Рисунок 21 – Уведомление об ошибке связанной с отсутствием ‹‹испорченного›› изображения

В случае ‹‹порчи›› изображения черными пикселями будет выдан путь, по которому хранится обработанное изображение:

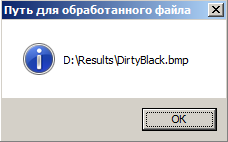


Рисунок 22 – Путь для обработанного файла черными пикселями

В случае ‹‹порчи›› изображения белыми пикселями будет выдан путь, по которому хра­нится обработанное изображение:

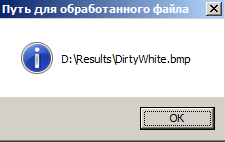


Рисунок 23 – Путь для обработанного файла белыми пикселями

В случае применения фильтра ‹‹сепия›› будет выдан путь, по которому хранится обрабо­танное изображение:

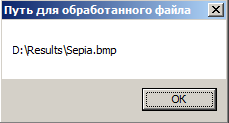


Рисунок 24 *–* Путь для обработанного файла фильтром ‹‹сепия››.

В случае применения фильтра ‹‹негатив›› будет выдан путь, по которому хранится обра­ботанное изображение:

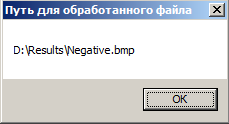


Рисунок 25 *–* Путь для обработанного файла фильтром ‹‹негатив››.

# Применение

# Назначение программы

Данное программное средство предназначено для работы с BMP-изображениями 24-хбитной глубиной цвета. Основными функциями являются:

* безопасность данных о пользователе;
* регистрация нового пользователя;
* редактирование данных пользователей в окне администрации;
* выбор любого языка из предложенных (Белорусский, Русский, Английский);
* выбор цвета на все формы приложения средствами colorDialog;
* возможность обрабатывать любое bmp-изображение с 24-хбитной глубиной цвета;
* возможность ‹‹испортить›› изображение рандомным расположением черных или бе­лых пикселей, благодаря вводу диапазона чисел пользователем;
* исправление ‹‹поврежденных›› пикселей;
* применение фильтров ‹‹сепия››, ‹‹негатив››.

# Условия применения

Для пользования данным программным средством необходимо обладать лицензионной копией MS Windows 10. Наличие последней версии .NET Framework 4.8.1.

Минимальные системные требования:

* процессор с частотой не менее 2.1 ГГц;
* ОЗУ не менее 58 МБайт;
* память на диске в размере 1.5 МБайт;
* MS Windows 7 x64;
* оптический привод.

Рекомендуемые системные требования:

* процессор с частотой не менее 2.4 ГГц;
* ОЗУ не менее 75 МБайт;
* память на диске в размере не менее 2 МБайт;
* MS Windows 10 x64;
* оптический привод.

# Заключение

Итогом работы можно считать разработку приложения, с помощью которого можно мо­дифицировать BMP-изображение c 24-хбитной глубиной цвета, путем применения различных инструментов, таких как: фильтр ‹‹сепия››, фильтр ‹‹негатив››, и самое главное – ‹‹загрязнение›› изображения черными и белыми пикселями, возможность исправления этого же ‹‹загряз­ненного›› изображения как по краям так и по углам изображения. Были проработаны и приме­нены 3 интересных алгоритма для исправления: для пикселей находящихся в углах, по бокам, и внутри боковых граней.

Также был выбран не самый легкий путь решения реализации локализации: был задей­ствован файл конфигурации (.config), в котором был написан xml код. Также я использовал Re­source Manager, благодаря которому был получен доступ к ресурсам, связанным с языком и ре­гиональными параметрами.

Во время разработки программного средства возникла трудность в использовании стро­ковых типов данных C++/CLI в файловых потоках C++, поскольку они несовместимы.

Дабы решить сложившуюся проблему мною было принято решение: использовать мар­шалинг строк как для функций C++, так и для функций C++/CLI.

В дальнейшем данное программное средство можно модернизировать путем смены маршалинга строк на встроенные функции C++/CLI. А также путем добавления нового функ­ционала, вплоть до несуществующих инструментов для работы с BMP-изображениями, которые можно будет придумать.

В процессе разработки приложения мною было освоено множество алгоритмов и их про­граммных реализаций, которые позволяют в дальнейшем эффективно использовать ресурсы при разработке.

# Список информационных источников

1. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск : КБП , 2013 . – 29 c.
2. Багласова, Т.Г. Методические указания по оформлению курсовых работ, дипломных проектов и отчётов для учащихся специальности 2–40 01 01 «ПОИТ» / Т.Г. Багласова. – Минск: КБП , 2016 . – 30 с.
3. Дейтел, Х. Как программировать на C++ / Х. Дейтел, П. Дейтел. – М.:ДМК Пресс, 2018 . – 1021 с.
4. Каталог API (Microsoft) и справочных материалов [Электронный ресурс].  – Microsoft, 2020.  –  Режим доступа : http://msdn.microsoft.com/ru–ru/library. – Дата доступа : 25.05.2021.
5. Общие требования к тестовым документам:ГОСТ 2.105–95. – Введ. 01.01.1996. – Минск:Межгос. совет по стандартизации, метрологии и сертификации , 1995 . – 84 с.
6. Описание программы. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.402–2000. – Введ. 01.09.2001 . – Минск:Межгос. совет по стандартизации, метрологии и сертификации , 2000 . – 14 с.
7. Текст программы. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.401–2000. – Введ. 01.09.2001 . – Минск:Межгос. совет по стандартизации, метрологии и сертификации , 2000 . – 16 с.

# Приложение А

(обязательное)

Код программы

//Admin.cpp

#include <Windows.h>

#include "Admin.h"

#include "Authorizati.h"

using namespace КурсоваяBMPFinal;

System::VoidКурсоваяBMPFinal::Admin::GetUsers\_Click(System::Object^sender,System::EventArgs^ e)

{

//чтение из файла в richTextBox1

auto f = System::IO::File::OpenText("records.txt");

//если файл пустой, то сообщить это админу

if (f–>Peek() == EOF)

{

MessageBox::Show(rm–>GetString(L"FreeFile"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

//читать файл до конца

auto s = f–>ReadToEnd();

//прочитанную информацию помещаем в richTextBox1

richTextBox1–>Text = s;

f–>Close();

}

System::VoidКурсоваяBMPFinal::Admin::SaveNewInfo\_Click(System::Object^sender,System::EventArgs^ e)

{

//проверяем richTextBox1 на пустоту

if (richTextBox1–>Text == "")

{

//если пустой, то выводим сообщение об этом

MessageBox::Show(rm–>GetString(L"FreeRich"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else

{

//если не пусто

System::IO::File::WriteAllText("records.txt", richTextBox1–>Text);

}

}

System::VoidКурсоваяBMPFinal::Admin::Back\_Click(System::Object^sender,System::EventArgs^ e)

{

this–>Hide(); // Скрываем Форму2

Form^ form = this–>Owner; // Получаем указатель на владельца

form–>Show(); // Показываем владельца

}

//Authorizati.cpp

#include <Windows.h>

#include "Authorizati.h"

#include "Errors.h"

#include <sys/types.h>

#include <sys/stat.h>

using namespace КурсоваяBMPFinal;

struct stat info;

[STAThreadAttribute]

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew Authorizati());

return 0;

}

System::Void КурсоваяBMPFinal::Authorizati::Enter\_Click(System::Object^ sender, System::EventArgs^ e)

{

try

{

User prof1;

String^ log = textBox1–>Text;

String^ pas = textBox2–>Text;

string log1, pas1;

//Преобразуем из String^ в string

MarshalString(log, log1);

MarshalString(pas, pas1);

//Методы get и set у объекта prof1 класса User

prof1.SetLogin(log1);

prof1.SetPassword(pas1);

if (log == "") throw freelog(); //проверка логина на пустоту и сразу же отлов ошибки

else if (pas == "") throw freepass(); //проверка пароля на пустоту и сразу же отлов ошибки

else if (textBox2–>TextLength < 5 || textBox2–>TextLength > 13) throw badlenght(); //проверка ввода допустимой длинны и сразу же отлов ошибки

//если введены спец.данные для админа, то открыть форму для администрации

else if (log == "adminyshka" && pas == "matvich3585")

{

this–>Hide();

f2–>Show();

}

else if (prof1.Chek() == true)

//если вошел зарегистрированный пользователь

{

if (stat("D:\\Results", &info) != 0)

{

//создание каталога Results на диске D если его там нет

std::wstring name(L"D:\\Results");

const wchar\_t\* szName = name.c\_str();

\_wmkdir(szName);

}

//приветствие пользователя

MessageBox::Show(rm–>GetString(L"Welcome") + " " + log + "!");

this–>Hide();

f1–>Show();

}

else

{

//сообщение об ошибке

MessageBox::Show(rm–>GetString(L"Smth"));

}

}

catch (freelog e)

{

//сообщение о пустоте строки для логина

MessageBox::Show(rm–>GetString(L"LoginFree"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

catch (freepass e)

{

//сообщение о пустоте строки для проля

MessageBox::Show(rm–>GetString(L"PassFree"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

catch (badlenght e)

{

//сообщение о недопустимой длинне символов

MessageBox::Show(rm–>GetString(L"PassLenght"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

catch (...)

{

//Универсальный отлов ошибок на всякий случай

MessageBox::Show(rm–>GetString(L"GG"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

System::VoidКурсоваяBMPFinal::Authorizati::Exit\_Click(System::Object^sender,System::EventArgs^ e)

{

//выход из приложения

Application::Exit();

}

System::VoidКурсоваяBMPFinal::Authorizati::Registration\_Click(System::Object^sender,System::EventArgs^ e)

{

//скрыть данную форму и перейти к форме регистрации

this–>Hide();

f–>Show();

}

System::VoidКурсоваяBMPFinal::Authorizati::Authorizati\_Load(System::Object^sender,System::EventArgs^ e)

{

//скрывает вводимые данные в textBox2

textBox2–>PasswordChar = '\*';

}

System::VoidКурсоваяBMPFinal::Authorizati::checkBox1\_CheckedChanged(System::Object^ sender, System::EventArgs^ e)

{

//если галочка стоит, то введенные данные видны

if (checkBox1–>Checked)

textBox2–>PasswordChar = false;

//иначе скрыть символом \*

else

textBox2–>PasswordChar = '\*';

}

System::VoidКурсоваяBMPFinal::Authorizati::RusLang\_Click(System::Object^sender,System::EventArgs^ e)

{

//обновление настроек для установки русского языка

changeLng–>UpdateConfig("language", "");

MessageBox::Show(rm–>GetString(L"Lang"),rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

System::VoidКурсоваяBMPFinal::Authorizati::EngLang\_Click(System::Object^sender,System::EventArgs^ e)

{

//обновление настроек для установки английского языка

changeLng–>UpdateConfig("language", "en–US");

MessageBox::Show(rm–>GetString(L"Lang"),rm–>GetString(L"Info"),MessageBoxButtons::OK, MessageBoxIcon::Information);

}

System::VoidКурсоваяBMPFinal::Authorizati::BelLang\_Click(System::Object^sender,System::EventArgs^ e)

{

//обновление настроек для установки белорусского языка

changeLng–>UpdateConfig("language", "be–BY");

MessageBox::Show(rm–>GetString(L"Lang"),rm–>GetString(L"Info"),MessageBoxButtons::OK, MessageBoxIcon::Information);

}

System::VoidКурсоваяBMPFinal::Authorizati::Restart\_Click(System::Object^sender,System::EventArgs^ e)

{

//перезагрузка приложения

Application::Restart();

}

System::VoidКурсоваяBMPFinal::Authorizati::Color\_Click(System::Object^sender,System::EventArgs^ e)

{

//если ответ диалога ОК

if (MyDialog–>ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

//конвертирование выбранного цвета в String

String^ a = Convert::ToString(MyDialog–>Color);

//если цвет черный, то на всех формах установить шрифт белый

if (a == "Color [Black]")

{

buttonWasClicked1 = true;

this–>BackColor = MyDialog–>Color;

this–>ForeColor = Color::White;

f–>UpdateFormColor(MyDialog);

f–>ForeColor = Color::White;

f1–>UpdateFormColor(MyDialog);

f1–>ForeColor = Color::White;

f2–>UpdateFormColor(MyDialog);

f2–>ForeColor = Color::White;

}

//если цвет белый, то на всех формах установить шрифт черный

else if (a == "Color [White]")

{

buttonWasClicked2 = true;

this–>BackColor = MyDialog–>Color;

this–>ForeColor = Color::Black;

f–>UpdateFormColor(MyDialog);

f–>ForeColor = Color::Black;

f1–>UpdateFormColor(MyDialog);

f1–>ForeColor = Color::Black;

f2–>UpdateFormColor(MyDialog);

f2–>ForeColor = Color::Black;

}

//вызвать глобальный метод UpdateFormColor для установки выбранного цвета на все формы

else

{

this–>BackColor = MyDialog–>Color;

f–>UpdateFormColor(MyDialog);

f1–>UpdateFormColor(MyDialog);

f2–>UpdateFormColor(MyDialog);

}

}

// оставить обычный цвет формы

else

{

return;

}

}

//BMP.cpp

#include "Func.h"

#include "BMP.h"

#include "Spravka.h"

System::VoidКурсоваяBMPFinal::BMP::Close1\_Click(System::Object^sender,System::EventArgs^ e)

{

if (pictureBox2–>Image != nullptr)

{

pictureBox2–>Image = nullptr;

pictureBox2–>Invalidate();

}

}

System::VoidКурсоваяBMPFinal::BMP::информацияToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

MessageBox::Show(rm–>GetString(L"Info1"),rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

System::VoidКурсоваяBMPFinal::BMP::выходToolStripMenuItem\_Click(System::Object^sender, System::EventArgs^ e)

{

//Выход из программы

Application::Exit();

}

System::VoidКурсоваяBMPFinal::BMP::инструкцияToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

Spravka^ s = gcnew Spravka(); //создать новую оконную форму типа класса Spravka

s–>Show();//отобразить окно s на экране ПК, при этом главное окно остается активным

}

System::VoidКурсоваяBMPFinal::BMP::Reset\_All\_Click(System::Object^sender,System::EventArgs^ e)

{

//перезагрузка приложения

Application::Restart();

}

System::VoidКурсоваяBMPFinal::BMP::Open\_Click(System::Object^sender,System::EventArgs^ e)

{

//если в диалоге был выбран вариант события ОК

if (ofd–>ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

pictureBox1–>ImageLocation = ofd–>FileName;

//в way1 помещаем имя выбранного файла

way1 = ofd–>FileName;

//в path помещаем путь к выбранному файлу

path = System::IO::Path::GetDirectoryName(ofd–>FileName);

files = System::IO::Directory::GetFiles(path);

for (int i = 0; i < files–>Length; i++)

{

if (files[i] == ofd–>FileName)

{

currentIndex = i;

break;

}

}

}

}

System::VoidКурсоваяBMPFinal::BMP::Close\_Click(System::Object^sender,System::EventArgs^ e)

{

if (pictureBox1–>Image != nullptr)

{

pictureBox1–>Image = nullptr;

pictureBox1–>Invalidate();

}

}

System::VoidКурсоваяBMPFinal::BMP::Shit\_Click(System::Object^ sender, System::EventArgs^ e)

{

try

{

//считывание ввеленных данных из textBox1 и textBox2

String^ S1 = textBox1–>Text;

String^ S2 = textBox2–>Text;

//ковертрование в тип данных инт

int x = Convert::ToInt32(S1);

int y = Convert::ToInt32(S2);

//проверка допустимых значений

if (x < 1 && y < 2)

{

MessageBox::Show(rm–>GetString(L"Invalid"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else if (x > y)

{

MessageBox::Show(rm–>GetString(L"Invalid"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else

{

//проверка на не пустоту pictureBox1

if (pictureBox1–>Image != nullptr)

{

//преобразоваение из String^ в const chat\* путем маршалирования

ManagedEmp^ employee = gcnew ManagedEmp();

employee–>name = gcnew String(way1);

marshal\_context context;

NativeEmp\* result = context.marshal\_as<NativeEmp\*>(employee);

//проверка на верное расширение

if (GetFileExtension(result–>name) == "bmp")

{

//проверка на верный формат

if (String::Format("Pixel format: {0}", pictureBox1–>Image–>PixelFormat) == "Pixel format: Format24bppRgb")

{

//создание потока

FILE\* f = OpenF(result);

BITMAPFILEHEADER header1;

BITMAPINFOHEADER header2;

//выделение памяти

BIT1(f, &header1, &header2);

//создание переменных для получения значений из функции

size\_t padding;

size\_t pixel\_count;

Pixel\* raster;

Pixel\* temp;

BIT2(f, header1, header2, &padding, &pixel\_count, &raster, &temp);

//выделение памяти для двумерного массива

Pixel\*\* arrr = Mas(header2);

Pixel\*\* a;

//функция порчи изображения

a = Shit\_Cl(arrr, header2, raster, x, y);

const char cc[] = "D:\\Results\\DirtyBlack.bmp";

const char vv[] = "w+b";

//функция сохранения файла

if (Savef\_(cc, vv, f, raster, header1, header2, temp, padding))

{

MessageBox::Show("D:\\Results\\DirtyBlack.bmp",rm–>GetString(L"Path"), MessageBoxButtons::OK, MessageBoxIcon::Information);

\_fcloseall();

}

}

else

{

MessageBox::Show(rm–>GetString(L"Format"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

else

{

MessageBox::Show(rm–>GetString(L"Expansion"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

else

{

MessageBox::Show(rm–>GetString(L"PhotoChoose"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

}

catch (System::FormatException^ e)

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", this–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"FromTo"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

System::VoidКурсоваяBMPFinal::BMP::ShitCl1\_Click(System::Object^sender,System::EventArgs^ e)

{

try

{

String^ S1 = textBox1–>Text;

String^ S2 = textBox2–>Text;

int x = Convert::ToInt32(S1);

int y = Convert::ToInt32(S2);

if (x <= 0 && y <= 0)

{

MessageBox::Show(rm–>GetString(L"Invalid"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else if (x > y)

{

MessageBox::Show(rm–>GetString(L"Invalid"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else

{

if (pictureBox1–>Image != nullptr)

{

ManagedEmp^ employee = gcnew ManagedEmp();

employee–>name = gcnew String(way1);

marshal\_context context;

NativeEmp\* result = context.marshal\_as<NativeEmp\*>(employee);

if (GetFileExtension(result–>name) == "bmp")

{

if (String::Format("Pixel format: {0}", pictureBox1–>Image–>PixelFormat) == "Pixel format: Format24bppRgb")

{

FILE\* f = OpenF(result);

BITMAPFILEHEADER header1;

BITMAPINFOHEADER header2;

BIT1(f, &header1, &header2);

size\_t padding;

size\_t pixel\_count;

Pixel\* raster;

Pixel\* temp;

BIT2(f, header1, header2, &padding, &pixel\_count, &raster, &temp);

Pixel\*\* arrr = Mas(header2);

Pixel\*\* a;

if (arrr == NULL)

{

MessageBox::Show(rm–>GetString(L"NoMemory"),rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

a = Shit\_Cl1(arrr, header2, raster, x, y);

const char cc[] = "D:\\Results\\DirtyWhite.bmp";

const char vv[] = "w+b";

Savef\_(cc, vv, f, raster, header1, header2, temp, padding);

\_fcloseall();

MessageBox::Show("D:\\Results\\DirtyWhite.bmp",rm–>GetString(L"Path"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

else

{

MessageBox::Show(rm–>GetString(L"Format"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

else

{

MessageBox::Show(rm–>GetString(L"Expansion"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

else

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", this–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"PhotoChoose"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

}

catch (System::FormatException^ e)

{ Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", this–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"FromTo"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

System::Void КурсоваяBMPFinal::BMP::Fix\_Click(System::Object^ sender, System::EventArgs^ e)

{

FILE\* f1 = fopen("D:\\Results\\DirtyBlack.bmp", "rb");

if (f1 == nullptr)

{

MessageBox::Show(rm–>GetString(L"NoDirty"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else

{

BITMAPFILEHEADER header1;

BITMAPINFOHEADER header2;

BIT1(f1, &header1, &header2);

size\_t padding;

size\_t pixel\_count;

Pixel\* raster;

Pixel\* temp;

BIT2(f1, header1, header2, &padding, &pixel\_count, &raster, &temp);

Pixel\*\* arrrr = Mas(header2);

if (arrrr == NULL)

{

MessageBox::Show(rm–>GetString(L"NoMemory"),rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

Pixel\*\* b;

b = Fix\_(arrrr, header2, raster);

const char cc[] = "D:\\Results\\FixedBlack.bmp";

const char vv[] = "w+b";

Savef\_(cc, vv, f1, raster, header1, header2, temp, padding);

\_fcloseall();

MessageBox::Show("D:\\Results\\FixedBlack.bmp",rm–>GetString(L"Path"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

}

System::VoidКурсоваяBMPFinal::BMP::Fix1\_Click(System::Object^sender,System::EventArgs^ e)

{

FILE\* f2 = fopen("D:\\Results\\DirtyWhite.bmp", "rb");

if (f2 == nullptr)

{

MessageBox::Show(rm–>GetString(L"NoDirty"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else

{

BITMAPFILEHEADER header1;

BITMAPINFOHEADER header2;

BIT1(f2, &header1, &header2);

size\_t padding;

size\_t pixel\_count;

Pixel\* raster;

Pixel\* temp;

BIT2(f2, header1, header2, &padding, &pixel\_count, &raster, &temp);

Pixel\*\* arrrr = Mas(header2);

if (arrrr == NULL)

{

MessageBox::Show(rm–>GetString(L"NoMemory"),rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

Pixel\*\* b;

b = Fix\_(arrrr, header2, raster);

const char cc[] = "D:\\Results\\FixedWhite.bmp";

const char vv[] = "w+b";

Savef\_(cc, vv, f2, raster, header1, header2, temp, padding);

\_fcloseall();

MessageBox::Show("D:\\Results\\FixedWhite.bmp", rm–>GetString(L"Path"));

}

}

System::VoidКурсоваяBMPFinal::BMP::Sepia\_Click(System::Object^sender,System::EventArgs^ e)

{

ManagedEmp^ employee = gcnew ManagedEmp();

employee–>name = gcnew String(pictureBox1–>ImageLocation);

marshal\_context context;

NativeEmp\* result = context.marshal\_as<NativeEmp\*>(employee);

if (pictureBox1–>Image != nullptr)

{

if (GetFileExtension(result–>name) == "bmp")

{

if (String::Format("Pixel format: {0}", pictureBox1–>Image–>PixelFormat) == "Pixel format: Format24bppRgb")

{

FILE\* f3 = OpenF(result);

BITMAPFILEHEADER header1;

BITMAPINFOHEADER header2;

BIT1(f3, &header1, &header2);

size\_t padding;

size\_t pixel\_count;

Pixel\* raster;

Pixel\* temp;

BIT2(f3, header1, header2, &padding, &pixel\_count, &raster, &temp);

Pixel\*\* arrrrrr = Mas(header2);

if (arrrrrr == NULL)

{

MessageBox::Show(rm–>GetString(L"NoMemory"), rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

Pixel\*\* d;

d = Sepia\_(arrrrrr, header2, raster);

const char cc[] = "D:\\Results\\Sepia.bmp";

const char vv[] = "w+b";

Savef\_(cc, vv, f3, raster, header1, header2, temp, padding);

\_fcloseall();

MessageBox::Show("D:\\Results\\Sepia.bmp", rm–>GetString(L"Path"));

}

else

{

MessageBox::Show(rm–>GetString(L"Format"), rm–>GetString(L"Er"));

}

}

else

{

MessageBox::Show(rm–>GetString(L"Expansion"), rm–>GetString(L"Er"));

}

}

else

{

MessageBox::Show(rm–>GetString(L"PhotoChoose"), rm–>GetString(L"Er"));

}

}

System::VoidКурсоваяBMPFinal::BMP::Negativ\_Click(System::Object^sender,System::EventArgs^ e)

{

ManagedEmp^ employee = gcnew ManagedEmp();

employee–>name = gcnew String(pictureBox1–>ImageLocation);

marshal\_context context;

NativeEmp\* result = context.marshal\_as<NativeEmp\*>(employee);

if (pictureBox1–>Image != nullptr)

{

if (GetFileExtension(result–>name) == "bmp")

{

if (String::Format("Pixel format: {0}", pictureBox1–>Image–>PixelFormat) == "Pixel format: Format24bppRgb")

{

FILE\* f2 = OpenF(result);

BITMAPFILEHEADER header1;

BITMAPINFOHEADER header2;

BIT1(f2, &header1, &header2);

size\_t padding;

size\_t pixel\_count;

Pixel\* raster;

Pixel\* temp;

BIT2(f2, header1, header2, &padding, &pixel\_count, &raster, &temp);

Pixel\*\* arrrrr = Mas(header2);

Pixel\*\* c;

if (arrrrr == NULL)

{

MessageBox::Show(rm–>GetString(L"NoMemory"), rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

c = Negative\_(arrrrr, header2, raster);

const char cc[] = "D:\\Results\\Negative.bmp";

const char vv[] = "w+b";

Savef\_(cc, vv, f2, raster, header1, header2, temp, padding);

\_fcloseall();

MessageBox::Show("D:\\Results\\Negative.bmp", rm–>GetString(L"Path"));

}

else

{

MessageBox::Show(rm–>GetString(L"Format"), rm–>GetString(L"Er"));

}

}

else

{

MessageBox::Show(rm–>GetString(L"Expansion"), rm–>GetString(L"Er"));

}

}

else

{

MessageBox::Show(rm–>GetString(L"PhotoChoose"), rm–>GetString(L"Er"));

}

}

System::Void КурсоваяBMPFinal::BMP::Open\_Res\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (ofd–>ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

pictureBox2–>ImageLocation = ofd–>FileName;

way1 = ofd–>FileName;

path = System::IO::Path::GetDirectoryName(ofd–>FileName);

files = System::IO::Directory::GetFiles(path);

for (int i = 0; i < files–>Length; i++)

{

if (files[i] == ofd–>FileName)

{

currentIndex = i;

break;

}

}

}

}

System::VoidКурсоваяBMPFinal::BMP::ToBack\_Click(System::Object^sender,System::EventArgs^ e)

{

//перемещение в выбранной директории изображений влево

if (currentIndex != 0)

{

currentIndex––;

}

pictureBox2–>ImageLocation = files[currentIndex];

}

System::VoidКурсоваяBMPFinal::BMP::ToRight\_Click(System::Object^sender,System::EventArgs^ e)

{

//перемещение в выбранной директории изображений вправо

if (currentIndex != files–>Length – 1)

{

currentIndex++;

}

pictureBox2–>ImageLocation = files[currentIndex];

}

//Registr.cpp

#include "Registr.h"

#include "Errors.h"

System::VoidКурсоваяBMPFinal::Registr::button1\_Click(System::Object^sender,System::EventArgs^ e)

{

try

{

User profi;

String^ logg = textBox1–>Text;

String^ pass = textBox2–>Text;

string logg1, pass1;

MarshalString(logg, logg1);

MarshalString(pass, pass1);

profi.SetLogin(logg1);

profi.SetPassword(pass1);

if (logg == "") throw freelog();

else if (pass == "") throw freepass();

else if (textBox2–>TextLength < 5 || textBox2–>TextLength > 13) throw badlenght();

else if (profi.writer() == true)

{

MessageBox::Show(rm–>GetString(L"SamePerson"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

else

{

MessageBox::Show(rm–>GetString(L"Reg"));

}

}

catch (freelog e)

{

MessageBox::Show(rm–>GetString(L"LoginFree"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

catch (freepass e)

{

MessageBox::Show(rm–>GetString(L"PassFree"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

catch (badlenght e)

{

MessageBox::Show(rm–>GetString(L"PassLenght"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

catch (...)

{

MessageBox::Show(rm–>GetString(L"GG"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

System::VoidКурсоваяBMPFinal::Registr::Registr\_Load(System::Object^sender,System::EventArgs^ e)

{

textBox2–>PasswordChar = '\*';

}

System::VoidКурсоваяBMPFinal::Registr::checkBox1\_CheckedChanged(System::Object^sender, System::EventArgs^ e)

{

if (checkBox1–>Checked)

textBox2–>PasswordChar = false;

else

textBox2–>PasswordChar = '\*';

}

System::VoidКурсоваяBMPFinal::Registr::Back\_Click(System::Object^sender,System::EventArgs^ e)

{

this–>Hide(); // Скрываем Форму2

Form^ form = this–>Owner; // Получаем указатель на владельца

form–>Show(); // Показываем владельца

}

//Spravka.cpp

#include "Spravka.h"

System::VoidКурсоваяBMPFinal::Spravka::button1\_Click(System::Object^sender,System::EventArgs^ e)

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", this–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"Spravka"),rm–>GetString(L"Info"), MessageBoxButtons::OK, MessageBoxIcon::Information);

}

//ChangeLanguage.h

#pragma once

using namespace System::Globalization;

using namespace System::Threading;

using namespace System::Collections::Generic;

using namespace System::Text;

using namespace System::Threading::Tasks;

using namespace System::Xml;

namespace КурсоваяBMPFinal

{

public ref class ChangeLanguage

{

public:

//функция для обновления настроек приложения

void UpdateConfig(String^ key, String^ value)

{

XmlDocument^ xmlDoc = gcnew XmlDocument();

//загрузка xml документа из указанного URL–адреса

xmlDoc–>Load(AppDomain::CurrentDomain–>SetupInformation–>ConfigurationFile);

for each (XmlElement ^ xmlElement in xmlDoc–>DocumentElement)

{

if (xmlElement–>Name–>Equals("appSettings"))

{

//представление отдельного узла в XML–документе

for each (XmlNode ^ xNode in xmlElement–>ChildNodes)

{

//получение атрибутов данного узла(ключ – значение)

if (xNode–>Attributes[0]–>Value–>Equals(key))

{

xNode–>Attributes[1]–>Value = value;

}

}

}

}

//сохранение настроек

System::Configuration::ConfigurationManager::RefreshSection("appSettings");

xmlDoc–>Save(AppDomain::CurrentDomain–>SetupInformation–>ConfigurationFile);

}

};

}

//DlyaReg.h

#pragma once

#include <fstream>

#include <string>

using namespace std;

//класс для авторизации и регистрации пользователей

class User

{

private:

//поля логин и пароль типа string

string login;

string password;

public:

//конструктор без параметров

User() {}

User(string login\_, string password\_)

{

SetLogin(login\_);

SetPassword(password\_);

}

//сеттер для логина

string SetLogin(string login\_)

{

login = login\_;

return login;

}

//сеттер для пароля

string SetPassword(string password\_)

{

password = password\_;

return password;

}

//метод для записи нового пользователя

bool writer()

{

bool h;

fstream file("records.txt", ios::in);

fstream file2("records.txt", ios::app);

string a;

while (!file.eof())

{

//читаем в временную переменную а

file >> a;

if (login != a)

{

h = false;

}

else

{

h = true;

return true;

break;

}

}

//если пвоторений нет то записываем нового пользователя

if (h == false)

{

file2 << login << " " << password << endl;

return false;

}

}

//метод для проверки есть ли уже такой пользователь

bool Chek()

{

bool h;

string log\_pas = login + " " + password;

string temp;

//создаем поток для чтения

fstream file("records.txt", ios::in);

//читаем до конца

while (!file.eof())

{

//читаем по линиям

getline(file, temp);

if (log\_pas == temp)

{

h = true;

break;

}

else

{

h = false;

}

}

if (h == true)

{

return true;

}

file.close();

}

};

//Erros.h

#pragma once

//ошибка для пустоты лоигна

class freelog {};

//ошибка для пустоты пароля

class freepass {};

//ошибка для неверной длинны

class badlenght{};

//Func.h

#pragma once

#include "BMP.h"

#include <Windows.h>

#include <iostream>

#include <msclr/marshal.h>

//структура для маршалирования, хранит поле типа String^

value struct ManagedEmp

{

System::String^ name;

};

//структура для маршалирования, хранит поле типа const char\*

struct NativeEmp {

const char\* name;

};

//готовая структура, поскольку формат БМП разработали не мы. Эта структура для установки и хранения цвета одного пикселя

struct Pixel

{

unsigned char r;

unsigned char g;

unsigned char b;

};

//готовый код для маршалирования, взятый на оффициаельном сайте microsoft

namespace msclr {

namespace interop {

template<>

ref class context\_node<NativeEmp\*, ManagedEmp^> : public context\_node\_base

{

private:

NativeEmp\* toPtr;

marshal\_context context;

public:

context\_node(NativeEmp\*& toObject, ManagedEmp^ fromObject)

{

// Conversion logic starts here

toPtr = NULL;

const char\* nativeName;

const char\* nativeAddress;

// Convert the name from String^ to const char\*.

System::String^ tempValue = fromObject–>name;

nativeName = context.marshal\_as<const char\*>(tempValue);

toPtr = new NativeEmp();

toPtr–>name = nativeName;

toObject = toPtr;

}

~context\_node()

{

this–>!context\_node();

}

protected:

!context\_node()

{

// When the context is deleted, it will free the memory

// allocated for toPtr–>name and toPtr–>address, so toPtr

// is the only memory that needs to be freed.

if (toPtr != NULL) {

delete toPtr;

toPtr = NULL;

}

}

};

}

}

using namespace System;

using namespace msclr::interop;

using namespace КурсоваяBMPFinal;

//функция для получения расширения файла

std::string GetFileExtension(const std::string& FileName)

{

if (FileName.find\_last\_of(".") != std::string::npos)

return FileName.substr(FileName.find\_last\_of(".") + 1);

return "";

}

Pixel\*\* Mas(BITMAPINFOHEADER a)

{

//выделение памяти для двумерного динамического массива

Pixel\*\* arr = new Pixel \* [a.biHeight];

BMP^ gg = gcnew BMP();

//если нет памяти

if (arr == NULL)

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", gg–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"NoMemory"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

for (int i = 0; i < a.biHeight; i++)

{

arr[i] = new Pixel[a.biWidth];

if (arr[i] == NULL)

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", gg–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"NoMemory"), rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

return arr;

}

//функция для порчи изображения черными пикселями

Pixel\*\* Shit\_Cl(Pixel\*\* arr, BITMAPINFOHEADER a, Pixel\* ras, int x, int c)

{

//используем рандомный шаг

for (int i = 0; i < a.biHeight; i += x + rand() % c)

{

for (int j = 0; j < a.biWidth; j += x + rand() % c)

{

//для работы с цветами, создадим указатель типа p

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

arr[i][j].r = 0;

arr[i][j].g = 0;

arr[i][j].b = 0;

\*p = arr[i][j];

}

}

return arr;

}

//функция для порчи изображения белыми пикселями

Pixel\*\* Shit\_Cl1(Pixel\*\* arr, BITMAPINFOHEADER a, Pixel\* ras, int x, int c)

{

for (int i = 0; i < a.biHeight; i += x + rand() % c)

{

for (int j = 0; j < a.biWidth; j += x + rand() % c)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

arr[i][j].r = 255;

arr[i][j].g = 255;

arr[i][j].b = 255;

\*p = arr[i][j];

}

}

return arr;

}

//функция для исправления изображения

Pixel\*\* Fix\_(Pixel\*\* arr, BITMAPINFOHEADER a, Pixel\* ras)

{

//по углам

for (int i = 0; i < a.biHeight; i++)

{

for (int j = 0; j < a.biWidth; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

if (i == 0 && j == a.biWidth – 1)

{

arr[i][j].b = (arr[i][j – 1].b + arr[i + 1][j – 1].b + arr[i + 1][j].b) / 3.0;

arr[i][j].g = (arr[i][j – 1].g + arr[i + 1][j – 1].g + arr[i + 1][j].g) / 3.0;

arr[i][j].r = (arr[i][j – 1].r + arr[i + 1][j – 1].r + arr[i + 1][j].r) / 3.0;

\*p = arr[i][j];

}

if (i == 0 && j == 0)

{

arr[i][j].b = (arr[i][j + 1].b + arr[i + 1][j + 1].b + arr[i + 1][j].b) / 3.0;

arr[i][j].g = (arr[i][j + 1].g + arr[i + 1][j + 1].g + arr[i + 1][j].g) / 3.0;

arr[i][j].r = (arr[i][j + 1].r + arr[i + 1][j + 1].r + arr[i + 1][j].r) / 3.0;

\*p = arr[i][j];

}

if (i == a.biHeight – 1 && j == a.biWidth – 1)

{

arr[i][j].b = (arr[i – 1][j].b + arr[i – 1][j – 1].b + arr[i][j – 1].b) / 3.0;

arr[i][j].g = (arr[i – 1][j].g + arr[i – 1][j – 1].g + arr[i][j – 1].g) / 3.0;

arr[i][j].r = (arr[i – 1][j].r + arr[i – 1][j – 1].r + arr[i][j – 1].r) / 3.0;

\*p = arr[i][j];

}

if (i == a.biHeight – 1 && j == 0)

{

arr[i][j].b = (arr[i – 1][j].b + arr[i – 1][j + 1].b + arr[i][j + 1].b) / 3.0;

arr[i][j].g = (arr[i – 1][j].g + arr[i – 1][j + 1].g + arr[i][j + 1].g) / 3.0;

arr[i][j].r = (arr[i – 1][j].r + arr[i – 1][j + 1].r + arr[i][j + 1].r) / 3.0;

\*p = arr[i][j];

}

}

}

//по бокам

for (int i = 0; i < 1; i++)

{

for (int j = 1; j < a.biWidth – 1; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

arr[i][j].b = (arr[i][j – 1].b + arr[i + 1][j – 1].b + arr[i + 1][j].b + arr[i + 1][j + 1].b + arr[i][j + 1].b) / 5.0;

arr[i][j].g = (arr[i][j – 1].g + arr[i + 1][j – 1].g + arr[i + 1][j].g + arr[i + 1][j + 1].g + arr[i][j + 1].g) / 5.0;

arr[i][j].r = (arr[i][j – 1].r + arr[i + 1][j – 1].r + arr[i + 1][j].r + arr[i + 1][j + 1].r + arr[i][j + 1].r) / 5.0;

\*p = arr[i][j];

}

}

for (int i = 1; i < a.biHeight – 1; i++)

{

for (int j = a.biWidth – 1; j < a.biWidth; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

arr[i][j].b = (arr[i – 1][j].b + arr[i – 1][j – 1].b + arr[i][j – 1].b + arr[i + 1][j – 1].b + arr[i + 1][j].b) / 5.0;

arr[i][j].g = (arr[i – 1][j].g + arr[i – 1][j – 1].g + arr[i][j – 1].g + arr[i + 1][j – 1].g + arr[i + 1][j].g) / 5.0;

arr[i][j].r = (arr[i – 1][j].r + arr[i – 1][j – 1].r + arr[i][j – 1].r + arr[i + 1][j – 1].r + arr[i + 1][j].r) / 5.0;

\*p = arr[i][j];

}

}

for (int i = a.biHeight – 1; i < a.biHeight; i++)

{

for (int j = 1; j < a.biWidth – 1; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

arr[i][j].b = (arr[i][j – 1].b + arr[i – 1][j – 1].b + arr[i – 1][j].b + arr[i – 1][j + 1].b + arr[i][j + 1].b) / 5.0;

arr[i][j].g = (arr[i][j – 1].g + arr[i – 1][j – 1].g + arr[i – 1][j].g + arr[i – 1][j + 1].g + arr[i][j + 1].g) / 5.0;

arr[i][j].r = (arr[i][j – 1].r + arr[i – 1][j – 1].r + arr[i – 1][j].r + arr[i – 1][j + 1].r + arr[i][j + 1].r) / 5.0;

\*p = arr[i][j];

}

}

for (int i = 1; i < a.biHeight – 1; i++)

{

for (int j = 0; j < 1; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

arr[i][j].b = (arr[i – 1][j].b + arr[i – 1][j + 1].b + arr[i][j + 1].b + arr[i – 1][j – 1].b + arr[i + 1][j].b) / 5.0;

arr[i][j].g = (arr[i – 1][j].g + arr[i – 1][j + 1].g + arr[i][j + 1].g + arr[i – 1][j – 1].g + arr[i + 1][j].g) / 5.0;

arr[i][j].r = (arr[i – 1][j].r + arr[i – 1][j + 1].r + arr[i][j + 1].r + arr[i – 1][j – 1].r + arr[i + 1][j].r) / 5.0;

\*p = arr[i][j];

}

}

//внутри

for (int i = 1; i < a.biHeight – 1; i++)

{

for (int j = 1; j < a.biWidth – 1; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

if ((arr[i][j].b == 0 & arr[i][j].g == 0 & arr[i][j].r == 0) || (arr[i][j].b == 255 & arr[i][j].g == 255 & arr[i][j].r == 255))

{

arr[i][j].b = (arr[i – 1][j – 1].b + arr[i – 1][j].b + arr[i – 1][j + 1].b + arr[i][j + 1].b + arr[i + 1][j + 1].b + arr[i + 1][j].b + arr[i + 1][j – 1].b + arr[i][j – 1].b) / 8.0;

arr[i][j].g = (arr[i – 1][j – 1].g + arr[i – 1][j].g + arr[i – 1][j + 1].g + arr[i][j + 1].g + arr[i + 1][j + 1].g + arr[i + 1][j].g + arr[i + 1][j – 1].g + arr[i][j – 1].g) / 8.0;

arr[i][j].r = (arr[i – 1][j – 1].r + arr[i – 1][j].r + arr[i – 1][j + 1].r + arr[i][j + 1].r + arr[i + 1][j + 1].r + arr[i + 1][j].r + arr[i + 1][j – 1].r + arr[i][j – 1].r) / 8.0;

\*p = arr[i][j];

}

}

}

return arr;

}

//функция для применения фильтра негатив

Pixel\*\* Negative\_(Pixel\*\* arr, BITMAPINFOHEADER a, Pixel\* ras)

{

for (int i = 1; i < a.biHeight – 1; i++)

{

for (int j = 1; j < a.biWidth – 1; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

//присваиваем пикселю его обратный цвет

arr[i][j].r = 255 – arr[i][j].r;

arr[i][j].g = 255 – arr[i][j].g;

arr[i][j].b = 255 – arr[i][j].b;

\*p = arr[i][j];

}

}

return arr;

}

//функция для применения фильтра сепия

Pixel\*\* Sepia\_(Pixel\*\* arr, BITMAPINFOHEADER a, Pixel\* ras)

{

int tr, tg, tb;

for (int i = 1; i < a.biHeight – 1; i++)

{

for (int j = 1; j < a.biWidth – 1; j++)

{

Pixel\* p = ras + a.biWidth \* i + j;

arr[i][j] = \*p;

tr = (int)(0.259 \* arr[i][j].r + 0.259 \* arr[i][j].g + 0.259 \* arr[i][j].b);

tg = (int)(0.359 \* arr[i][j].r + 0.359 \* arr[i][j].g + 0.359 \* arr[i][j].b);

tb = (int)(0.459 \* arr[i][j].r + 0.459 \* arr[i][j].g + 0.459 \* arr[i][j].b);

if (tr > 255 || tg > 255 || tb > 255)

{

tr = 255;

tg = 255;

tb = 255;

}

arr[i][j].r = (tr);

arr[i][j].g = (tg);

arr[i][j].b = (tb);

\*p = arr[i][j];

}

}

return arr;

}

//функция для открытия файла

FILE\* OpenF(NativeEmp\* a)

{

FILE\* file = fopen(a–>name, "rb");

BMP^ gg = gcnew BMP();

if (file == nullptr)

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", gg–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"BadFile"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

return file;

}

//функция сохранения обработанного файла

FILE\* Savef\_(const char a[], const char b[], FILE\* f, Pixel\* ras, BITMAPFILEHEADER c, BITMAPINFOHEADER d, Pixel\* temp, size\_t pad)

{

FILE\* new\_file;

fopen\_s(&new\_file, a, b);

BMP^ gg = gcnew BMP();

if (new\_file == nullptr)

{

Rsources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString",gg–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"BadFile"),rm–>GetString(L"Er"), MessageBoxButtons::OK, MessageBoxIcon::Warning);

fclose(f);

free(ras);

}

else

{

c.bfOffBits = sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER);

size\_t fieles\_count = fwrite(&c, sizeof(c), 1, new\_file);

fieles\_count = fwrite(&d, sizeof(d), 1, new\_file);

temp = ras;

for (int row = 0; row < d.biHeight; row++)

{

size\_t count = fwrite(temp, sizeof(Pixel), d.biWidth, new\_file);

unsigned char zero = 0;

fwrite(&zero, 1, pad, new\_file);

temp += count;

}

fclose(new\_file);

return new\_file;

}

}

void BIT1(FILE\* f, BITMAPFILEHEADER\* a, BITMAPINFOHEADER\* b)

{

fread(&(\*a), sizeof(BITMAPFILEHEADER), 1, f);

BMP^ gg = gcnew BMP();

if ((\*a).bfType != 19778)

{

Resources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString",gg MessageBox::Show(rm–>GetString(L"BadFile"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

fread(&(\*b), sizeof(BITMAPINFOHEADER), 1, f);

if ((\*a).bfType != 19778)

{

Rsources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString", gg–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"BadFile"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

}

}

void BIT2(FILE\* f, BITMAPFILEHEADER a, BITMAPINFOHEADER b, size\_t\* pad, size\_t\* pixc, Pixel\* (\*ras), Pixel\* (\*tmp))

{

(\*pad) = b.biWidth % 4;

(\*pixc) = b.biWidth \* b.biHeight;

fseek(f, a.bfOffBits, SEEK\_SET); //функция может сдвигать, ставить курсор работы с файлом (чтения/записи) в конкретную позицию отсносительно начала файла

(\*ras) = (Pixel\*)malloc(sizeof(Pixel) \* (\*pixc));//хардкорное выделение памяти как на

(\*tmp) = (\*ras);

BMP^ gg = gcnew BMP();

for (int row = 0; row < b.biHeight; row++)

{

size\_t count = fread((\*tmp), sizeof(Pixel), b.biWidth, f);

fseek(f, (\*pad), SEEK\_CUR);

(\*tmp) += count;

if (count != b.biWidth)

{

Rsources::ResourceManager^rm=gcnewResources::ResourceManager(L"КурсоваяBMPFinal.MyString",gg–>GetType()–>Assembly);

MessageBox::Show(rm–>GetString(L"BadFile"),rm–>GetString(L"Er"),MessageBoxButtons::OK, MessageBoxIcon::Warning);

fclose(f);

free((\*ras));//освобождение ДООП

return;

}

}

}

<?xml version="1.0" encoding="utf–8" ?>

<!––app.config––>

<!––Добавление единственного поля с ключом key и значением value для локализации приложения––>

<configuration>

<appSettings>

<add key="language" value="en–US"/>

</appSettings>

</configuration>



КБиП

У

Т. Контр.

Разраб.

Быковский М.М.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

Программа для обработки BMP-изображений с 24-хбитной глуибиной цвета

Блок-схема работы алгоритма для исправления ‹‹поврежденных›› пикселей.

№ докум.

Подп.

Дата

Изм.

Лист

Масса

Лист

Масштаб

Лист 1

Листов 7

Инв.№подл.

Подп. и дата

Взам.инв.№

Инв.№дубл.

Подп. и дата

КП Т.992002.401 ГЧ



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
|  |  |  |  |  | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *2* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
|  |  |  |  |  | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *3* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
|  |  |  |  |  | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *4* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
|  |  |  |  |  | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *5* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
|  |  |  |  |  | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *6* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
|  |  |  |  |  | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *7* |



КБиП

У

Т. Контр.

Разраб.

Быковский М.М.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

Программа для обработки BMP-изображений с 24-хбитной глуибиной цвета

*Блок-схема работы программы (общая)*

№ докум.

Подп.

Дата

Изм.

Лист

Масса

Лист

Масштаб

Лист 1

Листов 6

Инв.№подл.

Подп. и дата

Взам.инв.№

Инв.№дубл.

Подп. и дата

КП Т.992002.401 ГЧ



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| *Изм.* | *Лист* | *№ докум.* | *Подп.* | *Дата* | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *2* |
|  |  |  |  |  |
|  | | | | | | |
| *Изм.* | *Лист* | *№ докум.* | *Подп.* | *Дата* | *КП Т.992002.401 ГЧ* | *Лист* |
|  |  |  |  |  | *3* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| *Изм.* | *Лист* | *№ докум.* | *Подп.* | *Дата* | КП Т.992002.401 ГЧ | *Лист* |
|  |  |  |  |  | *4* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| *Изм.* | *Лист* | *№ докум.* | *Подп.* | *Дата* | *КП Т.992002.401 ГЧ* | *Лист* |
|  |  |  |  |  | *5* |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| *Изм.* | *Лист* | *№ докум.* | *Подп.* | *Дата* | *КП Т.992002.401 ГЧ* | *Лист* |
|  |  |  |  |  | *6* |
|  |  |  |  |  |

**Удостоверяющий лист**

электронного документа –курсовой проект

Тема КП «Программа для обработки BMP-файлов фильтрами, ‹‹сепия››, ‹‹негатив››, добавлением в изображение ‹‹поврежденных›› пикселей, исправление в изображении ‹‹поврежденных›› пикселей.

Обозначение КП КП Т.992002.401

Разработчик Быковский М.М. Руководитель Шаляпин Ю.В.

(Ф.И.О.) (Ф.И.О.)

Нормоконтроль Шаляпин Ю.В. Техконтроль Шаляпин Ю.В.

(Ф.И.О.) (Ф.И.О.)

Председатель цикловой комиссии Михалевич В.Ю.

(Ф.И.О.)

Рецензент Шаляпин Ю.В

(Ф.И.О.) (подпись/дата)

Подписи лиц, ответственных за разработку электронного документа

|  |  |  |
| --- | --- | --- |
| Состав ЭД | Разработчик | Руководитель |
| Пояснительная записка (на бумажном носителе формата А4), файл mainPZ.docx |  |  |
| Справочное пособие: файл <Справочное пособие.docx> |  |  |
| ГЧ №1, файл <Быковский ГЧ №1.vsd>,  ГЧ №2, файл <Быковский ГЧ№2.vsd> |  |  |
| Программная документация: файлы <Admin.cpp>, <Admin.h>,  <Authprizati.cpp>, <Authprizati.h>, <BMP.cpp>, <BMP.h>,  <Registr.cpp>, <Registr.h>, <Spravka.cpp>, <Spravka.h>, <Func.h>,  <ChangeLanguage.h>, <DlyaReg.h>, <Errors.h>, <app.config>,  <MyString.be–BY.resx>, <MyString.en–US.resx>, <MyString.resx> |  |  |
| Установочный пакет программного средства: BMP Viewer.exe |  |  |
| Тип носителя: оптический диск |  |  |

**Этикетка**

для дипломных (курсовых) проектов

**Курсовой проект**

Тема «Программа для обработки BMP-файлов фильтрами, ‹‹сепия››, ‹‹негатив››, добавле­нием в изображение ‹‹поврежденных›› пикселей, исправление в изображении ‹‹поврежденных›› пикселей.

КП Т.992002.401

Разработан 25.05.2021

Утвержден 31.05.2021

Разработчик: Быковский М.М.

Руководитель: Шаляпин Ю.В.

Технические средства Ноутбук Acer Aspire V3-571G, мышь, клавиатура.

Программные средства: MS OC Windows 7, Microsoft Visual Studio 2019, Microsoft Word 2007, Microsoft Visio 2007

**Состав документа:**

Пояснительная записка – файл mainPZ.docx

Программные документы – файлы («Admin.cpp»), («Admin.h»), (« Authprizati.cpp»), («Authprizati.h»), (« BMP.cpp»), (« BMP.h»), (« Registr.cpp»), (« Registr.h»), (« Spravka.cpp»), (« Spravka.h»), («ChangeLanguage.h»), («DlyaReg.h»), («Errors.h»), («Func.h»), («MyString.be–BY.resx»), («MyString.en–EU.resx»), («MyString.resx»), («app.config»).

Графическая часть – файлы («Быковский ГЧ№1.vsd»), («Быковский ГЧ№2.vsd»)

Сведения о защите информации: пароль отсутствует