Минобрнауки России

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт

(технический университет)»

|  |  |  |  |
| --- | --- | --- | --- |
| **Направление подготовки** | | 09.03.01 Информатика и вычислительная техника | |
| **Направленность** | | Систем автоматизированного проектирования и управления | |
| **Факультет** | | Информационных технологий и управления | |
| **Кафедра** | | Систем автоматизированного проектирования и управления | |
| **Курс** | 1 | **Группа** | 404 |

**КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»**

|  |  |
| --- | --- |
| **Тема:** | Разработка приложения для интегрирования водяного знака в изображение |

|  |  |
| --- | --- |
| Выполнил обучающийся | Д. К. Азаров |
| Заведующий кафедрой, проф. | Т. Б. Чистякова |
| Руководитель, доц. | И. Г. Корниенко |
| Консультант, ст. преп. | А. К. Федин |

**СОДЕРЖАНИЕ**

[ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 3](#_Toc73620196)

[ВВЕДЕНИЕ 4](#_Toc73620197)

[1 Аналитический обзор 5](#_Toc73620198)

[1.1 Обзор и анализ процесса интегрирования водяного знака в изображение. Сравнительная характеристика существующих систем интегрирования водяного знака в изображение. Обоснование актуальности интегрирования водяного знака в изображение. 5](#_Toc73620199)

[1.2 Общая характеристика и особенности интегрирования водяного знака в изображение 8](#_Toc73620200)

[1.3 Обзор и обоснование выбора инструментальных средств разработки приложения для интегрирования водяного знака в изображение 9](#_Toc73620201)

[2 Цель и задачи курсового проекта 11](#_Toc73620202)

[3 Технологическая часть. 12](#_Toc73620203)

[3.1 Формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации. 12](#_Toc73620204)

[3.2 Постановка задачи обработки информации 12](#_Toc73620205)

[3.3 Разработка функциональной структуры приложения для интегрирования водяного знака в изображение 13](#_Toc73620206)

[3.4 Разработка компонентов математического обеспечения приложения для интегрирования водяного знака в изображение 13](#_Toc73620207)

[3.5 Разработка структуры интерфейса пользователя приложения для интегрирования водяного знака в изображение 16](#_Toc73620208)

[3.6 Описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях) 16](#_Toc73620209)

[3.7 Описание структуры программы (модули, основные функции, классы и т. д.) 17](#_Toc73620210)

[3.8 Тестирование программного комплекса (на примере интегрирования водяного знака в изображение) 18](#_Toc73620211)

[ХАРАКТЕРИСТИКА ПРОГРАММНОГО И АППАРАТНОГО ОБЕСПЕЧЕНИЯ 24](#_Toc73620212)

[ВЫВОДЫ ПО РАБОТЕ 26](#_Toc73620213)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 27](#_Toc73620214)

# ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В настоящем отчете о курсовом проекте применяются следующие сокращения и обозначения:

API – Application Programming Interface, описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой

BMP – Bitmap Picture буквально является картой битов, а так как не использует никакого сжатия, то файлы такого формата получаются весьма большого размера

GIF – Graphics Interchange Format, [растровый](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0) формат графических изображений. Способен хранить сжатые данные без потери качества в формате не более [256 цветов](https://ru.wikipedia.org/wiki/8-%D0%B1%D0%B8%D1%82%D0%BD%D1%8B%D0%B9_%D1%86%D0%B2%D0%B5%D1%82)

IDE – Integrated Development Environment, интегрированная среда разработки программного обеспечения. Это обычная программа, которая обладает рядом качеств, позволяющих сделать работу программиста более удобной и продуктивной

JPEG – Joint Photographic Experts Group, один из популярных [растровых](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0) [графических форматов](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D1%8B), применяемый для хранения [фотографий](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F) и подобных им изображений

PNG – Portable Network Graphics, [растровый](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0) формат хранения графической информации, использующий [сжатие без потерь](https://ru.wikipedia.org/wiki/%D0%A1%D0%B6%D0%B0%D1%82%D0%B8%D0%B5_%D0%B1%D0%B5%D0%B7_%D0%BF%D0%BE%D1%82%D0%B5%D1%80%D1%8C) по алгоритму [Deflate](https://ru.wikipedia.org/wiki/Deflate)

ГБ – гигабайт, [единица измерения](https://ru.wikipedia.org/wiki/%D0%95%D0%B4%D0%B8%D0%BD%D0%B8%D1%86%D0%B0_%D0%B8%D0%B7%D0%BC%D0%B5%D1%80%D0%B5%D0%BD%D0%B8%D1%8F) [количества информации](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D1%82%D0%B2%D0%BE_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8)

ГГц – гигагерц, частота ядра процессора. Это значение определяет количество тактов в секунду у каждого ядра в процессоре

ЭВМ – Электронно-вычислительная машина, комплекс технических, аппаратных и программных средств, предназначенных для автоматической обработки информации, вычислений, автоматического управления

# ВВЕДЕНИЕ

В наше время множество людей выкладывают различные изображения в интернет. Они используются в разных сферах, начиная от ведения блога и заканчивая профессиональным заработком на творчестве. Однако, выкладывая личную работу в интернет, необходимо обезопасить себя и свои авторские права, поскольку без этого в большинстве случаев изображение попросту украдут для другого сайта или продадут от своего имени. Специально для выхода из такой ситуации есть водяные знаки.

Водяной знак – это отличный способ защитить контент и заявить об авторе изображения. Водяной знак может быть бледным и занимать большую часть изображения, а может разместиться в углу изображения и быть почти незаметным. Здесь всё зависит от цели, с которой вы наносите на изображение водяной знак.

Расположение водяного знака зависит от цели, которую вы преследуете, нанося знак. Чтобы просто указать авторство или брендировать контент, достаточно в одном из углов изображения разместить полупрозрачный знак, который не будет отвлекать на себя внимание. А для защиты изображения лучше разместить знак так, чтобы его было сложно убрать, но вместе с тем он не мешал восприятию изображения [[1]](#_Список_использованной_литературы).

Водяной знак можно накладывать с помощью множества различных [графический редактор](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80)ов, но обычно они очень сложны и в них надо разбираться. Поэтому существуют приложения, направленные на решение исключительно этой задачи. Самыми популярными программами для наложения водяного знака на изображение являются FastStone Photo Resizer, Easy Image Modifier, BImage Studio [[2]](#_Список_использованной_литературы).

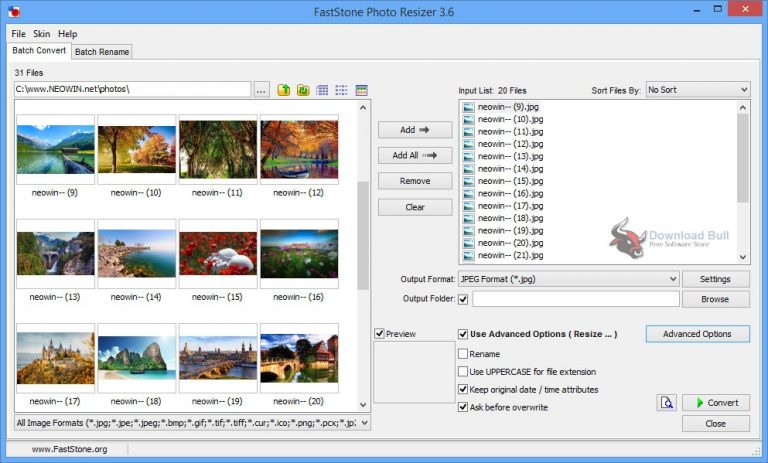
В связи с современными проблемами было решено создать программу для интегрирования водяного знака в изображение. Данными в данной программе являются 2 изначальных изображения (основное изображение и водяной знак). Они могу храниться как непосредственно на компьютере, так и на внешнем носителе. Изображения имеют формат \*.png, \*.bmp, \*.gif, \*.jpg. Данные форматы были выбраны из-за их популярности [3 – 7].

# 1 Аналитический обзор

## 1.1 Обзор и анализ процесса интегрирования водяного знака в изображение. Сравнительная характеристика существующих систем интегрирования водяного знака в изображение. Обоснование актуальности интегрирования водяного знака в изображение.

Программы для интегрирования водяного знака в изображение:

1. FastStone Photo Resizer v3.6

  
Рисунок 1 – Интерфейс приложения FastStone Photo Resizer

Возможности программы: включение/отключение наложения водяного знака, указание файла, который будет использоваться в этих целях (то есть сам файл у вас должен быть подготовлен заранее), подгон его позиционирования относительно основы, регулировка прозрачности и пара эффектов. Доступна пакетная обработка, что очень актуально в условиях необходимости быстрого нанесения водяного знака на несколько файлов, например, скриншотов для инструкции.

В дополнение разрешается управлять размерами фото, конвертировать их в другие популярные расширения, настраивая параметры процесса преобразования. Помимо всего прочего, в меню расширенных опций кроме раздела с водяным знаком можно найти и другие функциональные вкладки, позволяющие поворачивать, обрезать, отражать файлы, управлять глубиной цветов, добавлять текст, рамки [[2]](#_Список_использованной_литературы).

1. Easy Image Modifier v4.8

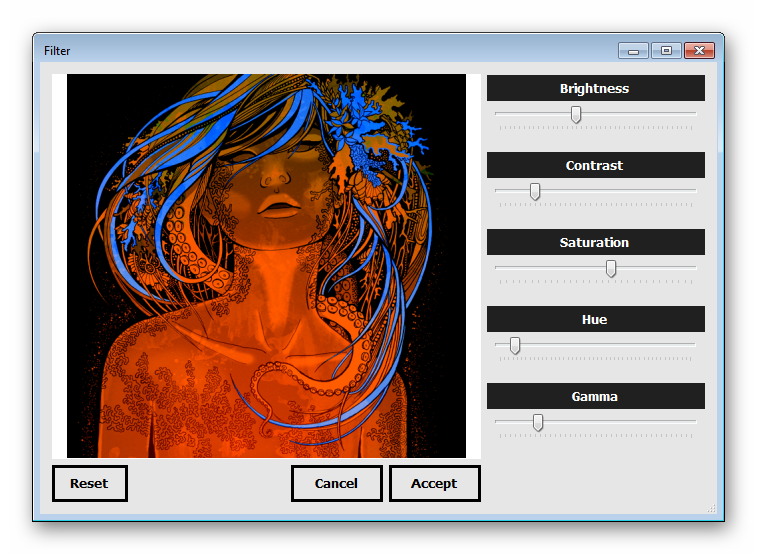
Рисунок 2 – Интерфейс приложения Easy Image Modifier



Чтобы осуществить наложение водяного знака на одно или сразу несколько изображений в этом приложении, надо активировать это средство в разделе добавления водяного знака, напишите текст, который хотите видеть на изображении, укажите параметры шрифта и расположение знака. После этого останется нажать на кнопку «Обработать». В этом приложении нельзя выбрать и наложить какое-то изображение из собственных ресурсов. В связи с этим можно сделать вывод, что софт подходит только для максимально простого способа нанесения водяных знаков, которые часто легко просто убрать, обрезав изображение или отредактировав его в условном Photoshop.

В этом приложении доступны: поворот, отражение, изменение размера, конвертирование в некоторые распространенные форматы. Словом, программа пригодится для максимально простого и незамысловатого редактирования и новичкам, которые не умеют или не хотят создавать персональные водяные знаки, довольствуясь текстовыми средствами защиты, созданными через стандартные шрифты [[2]](#_Список_использованной_литературы).

1. BImage Studio v1.2.1

  
Рисунок 3 – Интерфейс приложения BImage Studio

Первый вариант, текстовый, позволяет лишь только ввести слова, изменить цвет и стиль шрифта, указать позиционирование этой надписи. Тут же есть подсказка по возможным вариантам расположения водяного знака. Второй вариант предлагает указать адрес файла с готовым водяным знаком на компьютере, изменить его размер и настроить место, куда он будет помещен.

Из дополнительных возможностей тут есть изменение размера, фильтры (яркость, контраст и пр.), обрезка, поворот и пара других мелких опций. Также здесь имеется пакетная обработка, что поможет быстро нанести водяной знак сразу на все нужные изображения, загрузив их с компьютера и отметив галочками те фото, с которыми будет происходить дальнейшая работа [[2]](#_Список_использованной_литературы).

**Выводы по сравнительной характеристике**

В итоге можно сделать выводы, что самыми важными возможностями таких приложений являются: изменение размера, прозрачности и расположения водяного знака.

## 1.2 Общая характеристика и особенности интегрирования водяного знака в изображение

В приложении реализованы возможности:

**1) Выбор картинок**

у пользователя будет возможность выбрать 2 картинки, которые он хочет преобразовать. Выбранные пользователем картинки будут отображаться.

  
Рисунок 4 – Интерфейс загрузки изображений

**2) Масштабирование**

у пользователя будет возможность изменять размер водяного знака с помощью ползунка. Изменения будут отображаться в отдельном окне.

  
Рисунок 5 – Интерфейс масштабирования

**3) Прозрачность**

у пользователя будет возможность изменять прозрачность водяного знака с помощью ползунка. Изменения будут отображаться в отдельном окне.

  
Рисунок 6 – Интерфейс изменения прозрачности

**4) Место расположения**

у пользователя будет возможность изменять место расположения водяного знака на основном изображении с помощью ползунков. Изменения будут отображаться в отдельном окне.

  
Рисунок 7 – Интерфейс изменения расположения знака

**5) Сохранение изображения**

у пользователя будет возможность выбрать место, в котором он хочет сохранить изображение. Итоговое изображение будет отображаться.

   
Рисунок 8 – Интерфейс сохранения изображения

## 1.3 Обзор и обоснование выбора инструментальных средств разработки приложения для интегрирования водяного знака в изображение

**1) Сравнение С++ и Java:**

C++ и Java являются языками программирования общего назначения, что означает, что вы можете создавать практически любые типы программных приложений практически для любых платформ, используя правильные инструменты, IDE, библиотеки и фреймворки этих языков.

C++ используется для создания операционных систем, настольных приложений, веб-браузеров, механизмов рендеринга веб-браузеров, библиотек машинного обучения, приложений с тяжелой графической обработкой, баз данных, встроенных систем, мобильных приложений и т.д.

Java используется для создания серверов приложений, веб-приложений, мобильных приложений, настольных приложений, модульных тестов, корпоративных приложений, игр, облачных приложений, веб-API и т.д.

Из списка видно, что С++ больше подходит для поставленной задачи. Также можно учесть, что такая программа как Adobe Photoshop была написана на С++.

Программы на C++ обычно работают быстрее, так как их не нужно интерпретировать, а программы на Java не так быстры, поскольку их нужно сначала интерпретировать, и это требует времени.

Хотя у С++ есть и недостатки перед Java. C++ не является безопасным для памяти языком. Вы можете управлять памятью, что является большим преимуществом. Но могут возникать ошибки памяти и серьезные проблемы, такие как сбои, во время выполнения программы. Java не позволяет манипулировать памятью, она контролируется системой. Так что проблем не возникнет. Java - это язык, безопасный для памяти [[8]](#_Список_использованной_литературы).

**2) Сравнение С++ и С#:**

Достоинства С++ были рассмотрены выше, в С# огромным достоинством является возможность легкого создания пользовательского интерфейса, а также функция автоматической сборки мусора. Но несмотря на это большим минусом C# является не кроссплатформенность и то, что компиляция происходит по мере необходимости, то есть во время работы приложения могут быть подтормаживания [[9]](#_Список_использованной_литературы).

Таблица 1 – Сравнительная характеристика языков

|  |  |  |  |
| --- | --- | --- | --- |
|  | **C++** | **C#** | **Java** |
| **Целые числа произвольной длины** | - | - | + |
| **Перегрузка функций** | + | + | + |
| **Значения параметров по умолчанию** | + | - | - |
| **Поддержка исключений** | + | + | + |
| **Множественное наследование** | + | - | - |
| [**Ручное управление памятью**](https://ru.wikipedia.org/wiki/%D0%A1%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_%D1%8F%D0%B7%D1%8B%D0%BA%D0%BE%D0%B2_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F#%D0%A0%D1%83%D1%87%D0%BD%D0%BE%D0%B5_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C%D1%8E) | + | + | - |

Исходя из перечисленных выше пунктов, наиболее подходящий язык программирования для реализации проекта – C++.

# 2 Цель и задачи курсового проекта

Целью курсового проекта является разработка приложения для интегрирования водяного знака в изображение.

Задачи курсового проекта состоят в:

* написании формализованного описание процесса интегрирования водяного знака в изображение как объекта обработки и информации,
* постановке задачи обработки информации,
* разработке функциональной структуры приложения для интегрирования водяного знака в изображение,
* разработке компонентов математического обеспечения приложения для интегрирования водяного знака в изображение,
* описании структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях),
* описании структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях),
* описании структуры программы (модули, основные функции, классы и т. д.),
* тестировании программного комплекса (на примере интегрирования водяного знака в изображение).

# 3 Технологическая часть.

## 3.1 Формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации.

На рисунке 9 представлено формализованное описание процесса интегрирования водяного знака в изображение

  
Рисунок 9 – Формализованное описание процесса интегрирования водяного знака в изображение

где

X – вектор входных данных:

p – основное изображение;

w – водное изображение.

U – вектор управляющих воздействий:

size – размер;

pos – позиция;

tr – прозрачность.

Y – вектор выходных данных:

r – результирующее изображение.

## 3.2 Постановка задачи обработки информации

Задача состоит в наложении водяного знака на изображение с возможностью изменять размер, расположение и прозрачность знака.

## 3.3 Разработка функциональной структуры приложения для интегрирования водяного знака в изображение

На рисунке 10 представлена функциональная структура приложения для интегрирования водяного знака в изображение



Рисунок 10 – Функциональная структура приложения для интегрирования водяного знака в изображение

## 3.4 Разработка компонентов математического обеспечения приложения для интегрирования водяного знака в изображение

Для наложения водяного знака на изображение используется метод *DrawImage* из класса *Graphics* [[10 –11]](#_Список_использованной_литературы).

Для изменения размера водяного знака разработан метод *changeSizeWatermark* из класса *ImageWork* в этом методе используются формулы:

|  |
| --- |
| , |
| , |
| , |
| , |
| , |
| , |
| , |
| , |

Для изменения прозрачности знака разработан метод *changeTransparencyWatermark* из класса *ImageWork,* который меняет прозрачность каждого пикселя в градации от 0 до 255.

Для изменения позиции знака разработан метод *changePositionWatermark* из класса *ImageWork.*

где

*kw –* отношение ширины основного изображения к ширине водяного знака;

*w1* – ширина основного изображения;

*w2* – ширина водяного знака;

*hn* – новая высота водяного знака;

*kh* – отношение высоты основного изображения к высоте водяного знака;

*h1* – высота основного изображения;

*h2* – высота водяного знака;

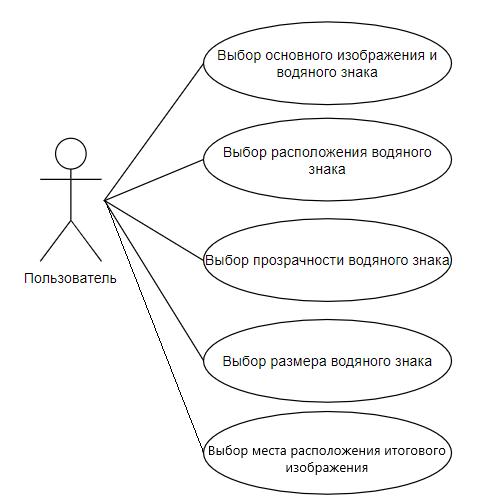
*wn* – новая ширина водяного знака;

*s* – новый размер водяного знака, % от основного изображения.

  
Рисунок 11 – Блок-схема алгоритма интегрирования водяного знака в изображение

## 3.5 Разработка структуры интерфейса пользователя приложения для интегрирования водяного знака в изображение

Структура пользовательского интерфейса состоит из 3 окон, по которым можно перемещаться между собой, каждое окно предоставляет пользователю свой тип взаимодействия.

  
Рисунок 12 – Структура пользовательского интерфейса

## 3.6 Описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях)

Таблица 2 – Переменные, составляющий модуль ImageWorkForm.cpp

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя переменной** | **Тип** | **Значение** | **Описание** |
| maxPercent | const int | 100 | Обозначает максимальное значение процента |
| minPercent | const int | 0 | Обозначает минимальное значение процента |
| maxAlpha | const double | 255 | Обозначает максимальное значение прозрачности |

## 3.7 Описание структуры программы (модули, основные функции, классы и т. д.)

Таблица 3 – Класс, составляющий модуль ImageWorkClass.h

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя класса** | **Имя переменной** | **Тип** | **Описание** |
| ImageWork | ­­­­­­­­– | – | Класс для интегрирования водяного знака в изображение |
| \_resultingImage | Bitmap^ | Хранит преобразованное изображение |
| \_mainImage | Bitmap^ | Хранит основное изображение |
| \_mainImagePath | String^ | Хранит путь к основному изображению |
| \_watermark | Bitmap^ | Хранит водяной знак |
| \_watermarkImagePath | String^ | Хранит путь к водяному знаку |
| \_transparencyPixel | array<  bool, 2>^ | Хранит матрицу прозрачности пикселя водяного знака |
| \_alpha | int | Прозрачность изображения |
| \_x | int | Хранит позицию водяного знака по горизонтали |
| \_y | int | Хранит позицию водяного знака по вертикали |
| \_newWidthWatermark | int | Хранит измененный размер водяного знака по горизонтали |
| \_newHeightWatermark | int | Хранит измененный размер водяного знака по вертикали |
| **Метод** | **Тип** | **Описание** |
| changePositionWatermark | Void | Изменяет позицию водяного знака |
| changeTransparencyWatermark | Void | Изменяет прозрачность водяного знака |
| changeSizeWatermark | void | Изменяет размер водяного знака |
| getMainImagePath | String^ | Возвращает путь основного изображения |
| getHeightMainImage | int | Возвращает размер основного изображения по вертикали |
| getWidthMainImage | int | Возвращает размер основного изображения по горизонтали |
| getWatermark | Bitmap^ | Возвращает водяной знак |
| getWatermarkImagePath | String^ | Возвращает путь водяного знака |
| getHeightWatermark | int | Возвращает размер водяного знака по вертикали |

Продолжение таблицы 3

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя класса** | **Метод** | **Тип** | **Описание** |
|  | getWidthWatermark | int | Возвращает размер водяного знака по горизонтали |
| getResultingImage | Bitmap^ | Возвращает итоговое изображение |

## 3.8 Тестирование программного комплекса (на примере интегрирования водяного знака в изображение)

В ходе тестирования было проверено:

1. Приложение успешно загружает изображения

  
Рисунок 13 – Загрузка изображений

1. При некорректном пути выводится сообщение о ошибке

  
Рисунок 14 – Сообщение о ошибочных данных

1. Водяной знак успешно накладывается на изображение

  
Рисунок 15 – Наложение водяного знака на изображение

  
Рисунок 16 – Наложение водяного знака на изображение

  
Рисунок 17 – Наложение водяного знака на изображение

1. Итоговое изображение успешно сохраняется

  
Рисунок 18 – Сохранение итогового изображения

  
Рисунок 19 – Сохранённое изображение

1. При некорректном пути или названии файла выводится сообщение о ошибке

  
Рисунок 20 – Сообщение о ошибке при некорректном пути

  
Рисунок 21 – Сообщение о существовании файла с данным названием

  
Рисунок 22 – Сообщение при попытке перезаписать изображение, использующееся в программе

# ХАРАКТЕРИСТИКА ПРОГРАММНОГО И АППАРАТНОГО ОБЕСПЕЧЕНИЯ

На рисунке 23 представлена структура программного обеспечения. Приложение разработано под управлением операционной системы Windows 10 на платформе Windows Forms в .NET 5. Средой разработки является Microsoft Visual Studio 2017 [[12 - 13]](#_Список_использованной_литературы).

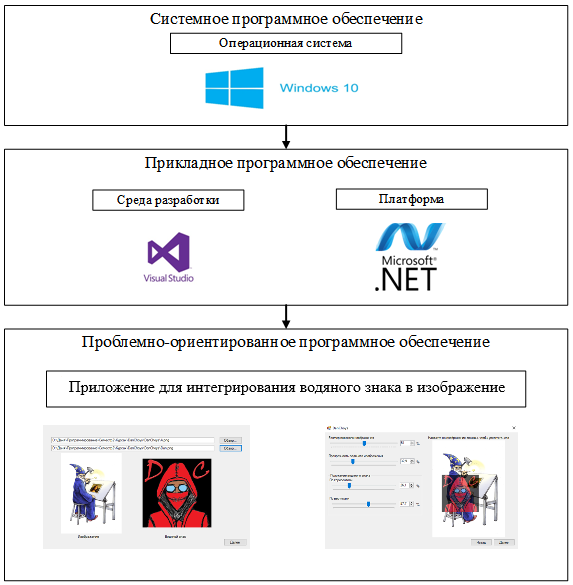


Рисунок 23 – Структура программного обеспечения

В таблице 4 приведена характеристика проблемно-ориентированного программного обеспечения.

Таблица 4 – Характеристика программного обеспечения

| Показатель | Значение |
| --- | --- |
| Среда разработки | Microsoft Visual Studio 2017 |
| Технология программирования | Объектно-ориентированное программирование |
| Язык программирования | C++ |
| Количество входных переменных | 2 |
| Количество внутренних переменных | 11 |
| Количество выходных переменных | 1 |
| Количество классов, структур | 1 |
| Количество функций | 12 |

Требования к ЭВМ, необходимой для нормального функционирования дистанционной системы представлены в таблице 5

Таблица 5 – Минимальные системные требования

| Показатель | Значение |
| --- | --- |
| Тип ЭВМ | Персональный компьютер |
| Тактовая частота процессора, ГГц | 1 |
| Объем оперативной памяти, ГБ | 1 |
| Объем внешней памяти, ГБ | 16 |
| Состав и характеристика периферийных устройств ЭВМ | Клавиатура, мышь, монитор с разрешением 800 × 600 пикселей |
| Состав и характеристика сетевого оборудования | Сетевой адаптер |
| Операционная система | Windows 10 |

# ВЫВОДЫ ПО РАБОТЕ

В итоге написания курсового проекта:

* создан класс для интегрирования водяного знака в изображение,
* разработан пользовательский интерфейс для интегрирования водяного знака в изображение,
* написано формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации,
* поставлена задачи обработки информации,
* разработана функциональная структура приложения для интегрирования водяного знака в изображение,
* разработаны компоненты математического обеспечения приложения для интегрирования водяного знака в изображение,
* описаны структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях),
* описаны структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях),
* описаны структуры программы (модули, основные функции, классы и т. д.),
* протестирован программный комплекс (на примере интегрирования водяного знака в изображение).

В перспективе в данное приложение можно будет добавить возможности:

* добавления текста,
* поворота водяного знака,
* накладывания эффектов,
* рисования на основном изображении.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Блог Sociate [Электронный ресурс]. – Режим доступа: <https://blog.sociate.ru/brendiruyte-i-zashchishchayte-kontent>/ (Дата обращения 26.05.2021)
2. [lumpics.ru](https://lumpics.ru/) [Электронный ресурс]. – Режим доступа: https://lumpics.ru/software-for-applying-a-watermark-on-a-photo/ (Дата обращения 26.05.2021)
3. libpng [Электронный ресурс]. – Режим доступа: http://www.libpng.org/pub/png/libpng-1.2.5-manual.html (Дата обращения 26.05.2021)
4. Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/BMP (Дата обращения 26.05.2021)
5. ZX Spectrum [Электронный ресурс]. – Режим доступа: https://zxpress.ru/article.php?id=7463 (Дата обращения 26.05.2021)
6. ZX Spectrum [Электронный ресурс]. – Режим доступа: https://zxpress.ru/article.php?id=7463 (Дата обращения 26.05.2021)
7. Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/JPEG (Дата обращения 16.05.2021)
8. ItGap [Электронный ресурс]. – Режим доступа: https://itgap.ru/post/cpp-vs-java (Дата обращения 26.05.2021)
9. C# – Преимущества и недостатки [Электронный ресурс]. – Режим доступа: <https://shwanoff.ru/plus-minus-c-sharp/> (Дата обращения 16.05.2021)
10. Microsoft [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.drawing.graphics.drawimage (Дата обращения 16.05.2021)
11. Microsoft [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.drawing.graphics (Дата обращения 26.05.2021)
12. Microsoft [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/ (Дата обращения 16.05.2021)
13. TechArks [Электронный ресурс]. – Режим доступа: https://techarks.ru/windows/rukovodstvo-dlya-nachinayushhih-o-tom-kak-nachat-rabotu-s-visual-studio/ (Дата обращения 26.05.2021)
14. Мейерс, С. Эффективный и современный С++ / C. Мейерс. – Москва : Вильямс, 2018. – 304 с. – ISBN 978-5-8459-2000-3.
15. Лафоре, Р. [Объектно-ориентированное программирование в С++](https://revall.info/obektno-orientirovannoe-programmirovanie-v-s.html) / Р. Лафоре. – Санкт-Питербург : Питер, 2004. – 923 c. – ISBN 978-5-496-00353-7.
16. [Wang](https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Meiqing+Wang&text=Meiqing+Wang&sort=relevancerank&search-alias=books), M. A Concise Introduction to Image Processing using C++ / [M. Wang](https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Meiqing+Wang&text=Meiqing+Wang&sort=relevancerank&search-alias=books), [C. Lai](https://www.amazon.com/s/ref=dp_byline_sr_book_2?ie=UTF8&field-author=Choi-Hong+Lai&text=Choi-Hong+Lai&sort=relevancerank&search-alias=books). – England : Chapman and Hall, 2008. – 268 c. – ISBN 978-1-584-88897-0.
17. Петцольд, Ч. Программирование с использованием Microsoft Windows Forms / Ч. Петцольд. – Санкт-Питербург : Питер, 206. – 432 с. – ISBN 5-7502-0284-4.
18. Рязанова, Н. Ю. Программирование на языке C++ в среде Visual Studio CLR Windows Forms / Н. Ю. Рязанова, К. Л. Тассов, М. В. Филиппов. – Москва : МГТУ имени Н. Э. Баумана, 2015. – 50 с. – ISBN 978-5-7038-4563-9.