МИНИСТЕРСТВА НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования «Самарский национальный исследовательский   
университет имени академика С.П. Королева»

(Самарский университет)

|  |  |
| --- | --- |
| Институт | информатики, математики и электроники |
| Факультет | информатики |
| Кафедра | геоинформатики и информационной безопасности |

КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ

«Технологии и методы программирования»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

«Разработка редактора растровой графики»

|  |  |
| --- | --- |
| Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(подпись)* | К.А. Козловская |
|  |  |
| Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(подпись)* | И.О. Фамилия |
|  |  |

САМАРА 2024

МИНИСТЕРСТВА НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования «Самарский национальный исследовательский   
университет имени академика С.П. Королева»

(Самарский университет)

|  |  |
| --- | --- |
| Институт | информатики, математики и электроники |
| Факультет | информатики |
| Кафедра | геоинформатики и информационной безопасности |

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

Студенту ***Козловской Ксении Александровны*** группы 631?

Тема проекта: ***«***Разработка редактора растровой графики***»***

|  |  |  |
| --- | --- | --- |
| Планируемые результаты освоения образовательной программы (компетенции) | Планируемые результаты практики | Содержание задания |
| ОПК-3 ‑ способность применять языки, системы и инструментальные средства программирования в профессиональной деятельности | Знать: основные виды программных средств, технологию разработки алгоритмов и программ и методы их отладки, основы объектно-ориентированного подхода к программированию  Уметь: работать с современными системами программирования, самостоятельно осваивать новые программные средства  Владеть: языками процедурного и объектно-ориентированного программирования, навыками разработки и отладки программ. | 1. Изучение существующих методов / алгоритмов…  2. Разработка метода / алгоритма…  3. Изучение существующих библиотек/программных средств…  4. Программная реализация метода/алгоритма…  5. Разработка программы/программного обеспечения …  6. Отладка и тестирование разработанной программы/программного обеспечения.  7. Проведение экспериментов и анализ результатов. |

Дата выдачи задания 5 сентября 2018 г.

Срок представления на кафедру отчета о практике 21 декабря 2020 г.

Руководитель курсового проекта

доцент каф. ГИиИБ, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.О. Фамилия

*(подпись)*

Задание принял к исполнению

студент группы № \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ К.А. Козловская

*(подпись)*

РЕФЕРАТ

**Пояснительная записка к курсовому проекту:**  ?? c., ?? рисунков, ?? таблиц, ?? источников, ?? приложений.

ЦИФРОВОЙ ВОДЯНОЙ ЗНАК, СЖАТИЕ ИЗОБРАЖЕНИЙ, ИНТЕРПОЛЯЦИЯ, ГРАФИЧЕСКИЙ РЕДАКТОР, ВИЗУАЛИЗАЦИЯ ГРАФОВ, ИНДЕКСИРОВАНИЕ ФАЙЛОВ, СОРТИРОВКА МАССИВОВ *(5-15 слов)*

Цель работы – разработка графического приложения для рисования с использованием Python и библиотеки Tkinter.

В ходе работы выполнен обзор существующих методов рисования и алгоритмов, таких как алгоритмы отрисовки графических примитивов и заливки. На основе проведенного анализа предложен алгоритм, позволяющий создавать и редактировать графические изображения, включая инструменты для рисования линий, кругов и квадратов.

Разработана программа, которая предоставляет интуитивно понятный интерфейс для пользователей. Программа позволяет изменять цвет и толщину линий, а также сохранять и загружать изображения в различных форматах. Исследование проводилось на целевой платформе Windows с использованием языка Python и среды разработки Visual Studio Code. В качестве системного программного обеспечения использовалась библиотека Pillow для обработки изображений.

В результате работы была создана функциональная программа, отвечающая заявленным требованиям, что подтверждено тестированием и анализом полученных результатов.

Содержание

[ВВЕДЕНИЕ 8](#_Toc532486987)

[1 МЕТОДЫ/АЛГОРИТМЫ/БИБЛИОТЕКИ/ ПРОГРАММНЫЕ СРЕДСТВА… 9](#_Toc532486988)

[1.1 Анализ методов … 9](#_Toc532486989)

[1.2 Описание алгоритмов … 9](#_Toc532486990)

[1.3 Существующие библиотеки/программные средства … 9](#_Toc532486991)

[1.4 Разработка алгоритма… 9](#_Toc532486992)

[1.5 Выводы и результаты 9](#_Toc532486993)

[2 ОПИСАНИЕ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 10](#_Toc532486994)

[2.1 Выбор средств разработки и системных программных средств 10](#_Toc532486995)

[2.2 Требования к разрабатываемой программе 10](#_Toc532486996)

[2.3 Структура программы 10](#_Toc532486997)

[2.3.1 Основные модули 10](#_Toc532486998)

[2.3.2 Описание основных классов 10](#_Toc532486999)

[2.3.3 Описание основных структур данных 11](#_Toc532487000)

[2.3.4 Описание основных функций и процедур 11](#_Toc532487001)

[2.4 Описание интерфейса пользователя 11](#_Toc532487002)

[2.5 Контрольный пример и результаты тестирования 12](#_Toc532487003)

[2.6 Выводы и результаты 12](#_Toc532487004)

[3 ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ 13](#_Toc532487005)

[3.1 Данные для экспериментов 13](#_Toc532487006)

[3.2 Методика проведения эксперимента 13](#_Toc532487007)

[3.3 Результаты экспериментов 13](#_Toc532487008)

[3.4 Выводы и результаты 13](#_Toc532487009)

[ЗАКЛЮЧЕНИЕ 14](#_Toc532487010)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_Toc532487011)

[ПРИЛОЖЕНИЕ А 16](#_Toc532487012)

ВВЕДЕНИЕ

Разработка графических приложений является важной и актуальной задачей в современном мире информационных технологий. Графические редакторы и средства визуализации данных широко используются в различных сферах, таких как дизайн, инженерия, образование и научные исследования. Создание удобных и функциональных инструментов для рисования и редактирования изображений может значительно повысить производительность и эффективность работы пользователей.

В настоящее время существует множество графических приложений, как коммерческих, так и с открытым исходным кодом. Однако многие из них обладают сложным интерфейсом, ограниченным функционалом или высокой стоимостью, что затрудняет их использование широким кругом пользователей. Кроме того, разработка собственных графических приложений с необходимым функционалом может быть сложной задачей для многих разработчиков.

Целью данной работы является разработка графического приложения для рисования с использованием Python и библиотеки Tkinter. Для достижения поставленной цели были сформулированы и решены следующие задачи:

1. Изучение существующих методов рисования и алгоритмов, таких как алгоритмы отрисовки графических примитивов и заливки;
2. Разработка алгоритма, позволяющего создавать и редактировать графические изображения, включая инструменты для рисования линий, кругов и квадратов;
3. Разработка программы с интуитивно понятным интерфейсом, позволяющей изменять цвет и толщину линий, а также сохранять и загружать изображения в различных форматах;
4. Проведение тестирования и анализа разработанного приложения.

# МЕТОДЫ/АЛГОРИТМЫ/БИБЛИОТЕКИ/ ПРОГРАММНЫЕ СРЕДСТВА.

## Анализ методов

В данном проекте используются различные алгоритмы и методы для создания графического редактора. Основное внимание уделяется обработке событий и рисованию на холсте с использованием библиотеки Tkinter.

## Описание алгоритмов

#### Основные компоненты

**Импорт библиотек**:

* tkinter: для создания графического интерфейса.
* PIL: для работы с изображениями.

**Класс** Paint:

* Основной класс, который инициализирует интерфейс и управляет логикой рисования.

**Конструктор**\_\_init\_\_

* Создаёт главное окно приложения.
* Инициализирует переменные для цвета, толщины линии, истории действий и состояния (например, режим рисования или стирания).
* Создаёт меню и панель инструментов.

**Методы**

**Создание меню**:

* Опции для сохранения, загрузки и очистки холста.
* Выбор фигур (квадрат, круг, линия и т.д.).

**Создание кнопок инструментов**:

* Кнопки для выбора ручки, ластика, заливки и управления историей действий (отмена/повтор).

**Функции работы с изображениями**:

* save\_image(): сохраняет текущее изображение.
* load\_image(): загружает изображение из файла.
* clear\_canvas(): очищает холст.

**Управление историей действий**:

* undo(): отменяет последнее действие.
* redo(): повторяет отменённое действие.

**Рисование и заливка**:

* paint(): рисует линии на холсте.
* bucket\_fill(): заливает область цветом, используя алгоритм заливки.
* \_flood\_fill(): реализует алгоритм заливки.

**Выбор цвета**:

* choose\_color(): открывает диалог выбора цвета.
* add\_recent\_color(): добавляет выбранный цвет в список последних цветов.

**Работа с формами**:

* Методы для выбора и рисования различных фигур (квадрат, круг, линия и т.д.).

## Существующие библиотеки/программные средства

#### **1.3.1 Tkinter**

**Описание**: Стандартная библиотека для создания графических интерфейсов в Python. Она предоставляет инструменты для создания окон, кнопок, меню и других элементов интерфейса.

**Преимущества:**

* Простота использования и интуитивно понятный интерфейс.
* Кроссплатформенность: работает на Windows, macOS и Linux.
* Легкость интеграции с другими библиотеками Python.

#### **1.3.2 PIL/Pillow**

**Описание:** Библиотека для обработки изображений в Python. Pillow является обновлённой версией оригинальной библиотеки PIL и предоставляет возможности для работы с изображениями.

**Преимущества:**

* Поддержка множества форматов изображений (JPEG, PNG и др.).
* Возможности редактирования изображений, такие как изменение размера, обрезка и применение фильтров.
* Удобный интерфейс для работы с графикой.

## Разработка алгоритма…

#### Алгоритм заливки

**Описание**: Алгоритм заливки использует метод "поиска в глубину" (DFS) для заполнения области на изображении заданным цветом, начиная с указанной точки. Он реализован с помощью стека, что позволяет избежать проблем с переполнением стека при рекурсивном подходе.

**Входные данные**:

* x, y: координаты начальной точки заливки.
* target\_color: цвет, который необходимо заменить.
* replacement\_color: цвет, которым нужно заполнить область.

**Выходные данные**:

* Изменённое изображение с залитой областью.

Псевдокод

flood\_fill(x, y, target\_color, replacement\_color):

создать стек и добавить (x, y)

пока стек не пуст:

(x, y) = извлечь из стека

если (x, y) вне границ изображения:

продолжить

если цвет пикселя (x, y) не равен target\_color:

продолжить

если цвет пикселя (x, y) равен replacement\_color:

продолжить

изменить цвет пикселя (x, y) на replacement\_color

добавить соседние пиксели в стек:

(x + 1, y)

(x - 1, y)

(x, y + 1)

(x, y - 1)

Значимость алгоритма

**Эффективность**: Использование стека вместо рекурсии позволяет избежать переполнения стека при больших областях заливки, что делает алгоритм более устойчивым к ошибкам.

**Гибкость**: Алгоритм может быть легко адаптирован для работы с различными цветами и формами, что делает его универсальным инструментом для редактирования изображений.

### Пример реализации

Вот пример кода, реализующего алгоритм заливки:

def \_flood\_fill(self, x, y, target\_color, replacement\_color):

stack = [(x, y)]

while stack:

x, y = stack.pop()

if x < 0 or x >= self.image.width or y < 0 or y >= self.image.height:

continue

if self.image.getpixel((x, y)) != target\_color:

continue

if self.image.getpixel((x, y)) == replacement\_color:

continue

self.image.putpixel((x, y), replacement\_color)

stack.append((x + 1, y)) # Правый пиксель

stack.append((x - 1, y)) # Левый пиксель

stack.append((x, y + 1)) # Нижний пиксель

stack.append((x, y - 1)) # Верхний пиксель

Этот алгоритм является важной частью функциональности графического редактора, позволяя пользователям легко изменять цвета на изображении.

## Выводы и результаты

Результаты работы подтверждают правильность выбранного подхода и технологий. Графический редактор уже сейчас демонстрирует стабильную работу и высокую производительность, что создаёт хорошую основу для его дальнейшего развития и улучшения.

# ОПИСАНИЕ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## Выбор средств разработки и системных программных средств

## Язык программирования

**Python** был выбран в качестве основного языка программирования для разработки графического редактора по следующим причинам:

* **Простота и читаемость**: Python известен своей простой и понятной синтаксисом, что облегчает процесс разработки и сопровождения кода. Это особенно важно для студентов и начинающих программистов.
* **Широкая библиотека стандартных модулей**: Python имеет обширную стандартную библиотеку, которая включает в себя модули для работы с графикой (например, Tkinter) и обработки изображений (Pillow). Это позволяет быстро реализовать функционал без необходимости создавать все с нуля.
* **Кроссплатформенность**: Python работает на различных операционных системах (Windows, macOS, Linux), что позволяет разрабатывать приложение, доступное для широкой аудитории пользователей.

### Среда разработки

**Visual Studio Code (VS Code)** была выбрана в качестве основной среды разработки по следующим причинам:

* **Легкость и быстродействие:** VS Code — это легковесный редактор, который быстро загружается и не требует много ресурсов, что делает его удобным для разработки.
* **Расширяемость:** VS Code поддерживает множество расширений, что позволяет настраивать редактор под конкретные нужды проекта. Например, можно установить расширения для работы с Python, управления версиями и других необходимых инструментов.
* **Интеграция с системами контроля версий:** VS Code имеет встроенную поддержку Git, что упрощает управление версиями и совместную работу над проектом.
* **Поддержка отладки:** VS Code предоставляет мощные инструменты для отладки, что позволяет легко находить и устранять ошибки в коде.

### Системные программные средства

Для работы программы необходимы следующие системные программные средства:

**Операционная система**:

* Windows 10 или новее
* macOS 10.14 или новее
* Linux (распределения, поддерживающие Python)

**Система управления базами данных**:

Для данного проекта не требуется полноценная система управления базами данных, так как приложение не хранит данные в базе. Однако, в будущем, если потребуется хранение пользовательских настроек или истории действий, можно использовать SQLite, который является легковесной встроенной СУБД.

**Дополнительные библиотеки**:

* **Tkinter:** Библиотека для создания графического интерфейса.
* **Pillow:** Библиотека для обработки изображений.

## Требования к разрабатываемой программе

### Функциональные требования

Функциональные требования описывают основные функции и возможности, которые должна обеспечивать разрабатываемая программа:

### Создание и редактирование изображений:

* Пользователь должен иметь возможность создавать новое изображение с заданными размерами.
* Возможность редактирования существующих изображений (открытие, сохранение, изменение).

### Инструменты рисования:

* Наличие инструментов для рисования линий, кругов, прямоугольников и других геометрических фигур.
* Возможность выбора цвета и толщины линий.

### Заливка цветом:

* Пользователь должен иметь возможность заливать области цветом с помощью инструмента заливки.

### Работа с текстом:

Возможность добавления текста на изображение с настройками шрифта, размера и цвета.

### Отмена и повтор действий:

* Реализация функции отмены и повтора последних действий пользователя.

### Сохранение и загрузка файлов:

* Поддержка различных форматов файлов для сохранения изображений (например, PNG, JPEG).
* Возможность открытия изображений в указанных форматах.

### Нефункциональные требования

Нефункциональные требования определяют качества системы, такие как производительность, безопасность и удобство использования:

**Производительность:**

* Программа должна обеспечивать быструю загрузку изображений и отзывчивый интерфейс при выполнении операций рисования.

**Удобство использования:**

* Интерфейс должен быть интуитивно понятным и удобным для пользователей с разным уровнем подготовки.
* Наличие справочной информации и подсказок для пользователей.

**Кроссплатформенность:**

* Программа должна работать на основных операционных системах: Windows, macOS и Linux.

**Безопасность:**

* Необходимо обеспечить защиту данных пользователя, особенно при сохранении изображений и настроек.

**Документация:**

* Должна быть предоставлена документация по установке, настройке и использованию программы.

## Структура программы

Здесь описывается структура программы. Указывается, из каких программных компонент (исполняемых файлов) она состоит.

### Основные модули

Для каждой компоненты указывается, из каких файлов с исходными кодами осуществляется сборка компоненты.

### Описание основных классов

Здесь дается описание основных классов, разработанных в рамках проекта. Для каждого класса приводится описание наиболее значимых полей и методов.

### Описание основных структур данных

Здесь дается описание основных структур данных (за исключением описанных в предыдущем разделе), разработанных в рамках проекта, если таковые имеются.

### Описание основных функций и процедур

В том случае, если часть логики программы располагается в подпрограммах (за исключением методов классов), здесь дается описание основных функций и процедур.

Фрагменты кода подпрограмм, используемые в разделе, рекомендуется оформлять следующим образом:

int Sum(int a, int b);

Рисунок 2 – Заголовок функции суммирования

Функция *Sum* возвращает сумму двух целочисленных значений, задаваемых входными параметрами *a* и *b*.



Рисунок 3 – Пример фрагмента кода

## Описание интерфейса пользователя

В этом разделе описывается интерфейс пользователя с указанием порядка работы с программой и назначением всех элементов пользовательского интерфейса.

## Контрольный пример и результаты тестирования

Необходимо разработать контрольные примеры и показать, что на них разработанная программа работает нормально, что должно подтверждаться приводимыми на рисунках копиями экрана.

Для исследовательских работ, содержащих раздел 3 «ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ», допускается не выполнять п. 2.5.

## Выводы и результаты

В этом подразделе приводятся основные выводы и результаты по разделу.

ЗАКЛЮЧЕНИЕ

Краткие выводы и оценка полученных в курсовом проекте результатов (не более 1 страницы).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Лебеденко, Ю.И. Биометрические системы безопасности [Текст]: учеб. пособие для вузов / Ю.И. Лебеденко. – М.: Директ-Медиа, 2012. – 160 с.  
- пример оформления учебного пособия

2 Форматы звуковых файлов: [Электронный ресурс] // Ингибитор. Второй сборник. URL: http://ingibit.rigalink.lv/info/c2/formats.html. (Дата обращения:  18.02.2018).  
- пример оформления электронного ресурса

3 Сорокин, В.Н. Распознавание личности по голосу: аналитический обзор [Текст] / В.Н. Сорокин, В.В. Вьюгин, А.А. Тананыкин // Информационные процессы. – 2012. – Т. 12, №1. – С. 1-30.  
- пример оформления книга 3 автора

4 Первушин, Е.А. Обзор основных методов распознавания дикторов [Текст] / Е.А. Первушин // Математические структуры и моделирование. –2011. – № 24. – С. 41-54.  
- пример оформления статья в журнале

5 Рабинер, Л.Р. Цифровая обработка речевых сигналов [Текст]: пер. с англ. / Л.Р. Рабинер, Р.В. Шафер. – М.: Радио и связь, 1981. – 496 с.

ПРИЛОЖЕНИЕ А

Код программы

В приложения выносится вспомогательный материал:

* исходные коды программ;
* результаты измерений;
* описания используемых наборов данных и т.д.

Допускается разделять раздел на подразделы, например, «А.1», «А.2». Такие подразделы могут соответствовать, например, разным программным модулям.