

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МО ЭВМ**

**КУРСОВАЯ РАБОТА**  
**по дисциплине «Программирование»**  
**Тема: Работа с изображениями**

Студент гр. 4342

\_\_\_\_\_

Ларионов В.А.

Преподаватель

\_\_\_\_\_

Виноградова Е.В.

Санкт-Петербург

2025

## ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студент: Ларионов В.А.

Группа: 4342

Тема работы: Работа с изображениями

Исходные данные:

Программа обязательно должна иметь CLI (опционально дополнительное использование GUI). Программа должна реализовывать весь следующий функционал по обработке png-файла

(1) Рисование прямоугольника. Флаг для выполнения данной операции:

`--rect``.

(2) Сделай рамку в виде узора. Флаг для выполнения данной операции:

`--ornament``.

(3) Поворот изображения (части) на 90/180/270 градусов. Флаг для выполнения данной операции: `--rotate``.

Все подзадачи, ввод/вывод должны быть реализованы в виде отдельной функции.

Разделы пояснительной записки: «Содержание», «Введение», «Структуры», «Функции», «Заключение», «Список использованных источников», «Приложение А. Примеры работы программы», «Приложение Б. Исходный код программы».

Предполагаемый объем пояснительной записки:

Не менее 61 страниц.

Дата выдачи задания: 07.03.2025

Дата сдачи реферата: 10.05.2025

Дата защиты реферата: 16.05.2025

Студент

\_\_\_\_\_

Ларионов В.А.

Преподаватель

\_\_\_\_\_

Виноградова Е.В.

## **АННОТАЦИЯ**

Курсовая работа представляет собой разработку программы для обработки изображений в формате PNG. Проект реализуется на языке C, с соблюдением требований к отказоустойчивости. Особенности реализации описаны в общих сведениях и описании работы функций. Примеры использования CLI в Приложении А. Исходный код в Приложении Б. Программа корректно обрабатывает ошибки с выводом понятных сообщений пользователю.

## **ABSTRACT**

This coursework involves developing a program for processing PNG images. The project is implemented in C, meeting fault-tolerance requirements. Implementation details are described in the general information and function descriptions. Examples of CLI usage are in Appendix A. The source code is in Appendix B. The program correctly handles errors, displaying clear messages to the user.

## СОДЕРЖАНИЕ

|       |   |    |
|-------|---|----|
|       | Введение                                    | 5  |
| 1.    | Структуры                                   | 6  |
| 1.1.  | Структура png_t                             | 6  |
| 1.2.  | Структура options_t                         | 6  |
| 1.3.  | Структура options_mask_t                    | 7  |
| 1.4.  | Структура point_t                           | 8  |
| 1.5.  | Структура rgb_color_t                       | 8  |
| 1.6.  | Структура line_t                            | 8  |
| 1.7.  | Перечисление fill                           | 9  |
| 1.8.  | Перечисление pattern_of_ornament            | 9  |
| 1.9.  | Перечисление angle                          | 9  |
| 1.10. | Перечисление error_marker_t                 | 9  |
| 2.    | Функции                                     | 11 |
| 2.1.  | Функция main                                | 11 |
| 2.2.  | Подготовительные функции                    | 12 |
| 2.3.  | Функции для работы с файлами                | 12 |
| 2.4.  | Функции выполнения заданий                  | 12 |
| 2.5.  | Функции рисования и вспомогательные для них | 13 |
|       | Заключение                                  | 15 |
|       | Список использованных источников            | 16 |
|       | Приложение А. Примеры работы программы      | 17 |
|       | Приложение Б. Исходный код программы        | 19 |

## **ВВЕДЕНИЕ**

Цель работы: написать программу на языке C, которая получает аргументы переданные через командную строку, считывает и обрабатывает изображение формата PNG требуемым образом. Для этого требовалось написать:

- Функцию получения аргументов из командной строки;
- Функции валидации аргументов;
- Функции для чтения и записи изображений формата PNG;
- Функции обработки изображения.

## 1. СТРУКТУРЫ

Чтобы облегчить написание и чтение кода, было принято решение реализовать и использовать основные структуры: `png_t`, `options_t`; вспомогательные: `options_mask_t`, `point_t`, `rgb_color_t`, `line_t` и перечисления: `fill`, `pattern_of_ornament`, `angle`, `error_marker_t`.

### 1.1 Структура `png_t`

Структура `png_t` представляет информацию об изображении в формате PNG и массив пикселей. Она содержит следующие поля:

- `int width`: Ширина изображения в пикселях;
- `int height`: Высота изображения в пикселях;
- `png_byte color_type`: Тип цвета изображения;
- `png_byte bit_depth`: Глубина цвета изображения;
- `png_byte channels`: Количество цветовых каналов у изображения;
- `png_structp png_ptr`: Указатель на структуру `libpng` для чтения/записи данных PNG;
- `png_infor info_ptr`: Указатель на структуру `libpng` для хранения информации о PNG;
- `int number_of_passes`: Количество проходов, необходимых для полного отображения изображения;
- `png_bytep row_pointers`: Указатель на массив указателей, каждый из которых указывает на строку данных изображения.

Эта структура необходима для представления информации о PNG-изображении в программе, а также для работы с библиотекой `libpng` для чтения и записи данных в формате PNG.

### 1.2 Структура `options_t`

Эта структура предназначена для хранения переданных пользователем аргументов.

Содержит следующие поля:

- `enum error_marker_t err`: перечисление хранящее значение ошибки;

- int func\_name: переменной присваивается значения соответствующие значению функции;
- char input\_filename[]: хранит относительный путь к входному файлу;
- char output\_filename[]: хранит относительный путь к выходному файлу;
- point\_t left\_up: значение для левой верхней точки;
- point\_t right\_down: значение для правой нижней точки;
- rgb\_color\_t color: значение цвета;
- enum fill fill: принимает два значения, заливать или не заливать (вот в чем вопрос);
- rgb\_color\_t fill\_color: значение цвета для заливки;
- unsigned thickness: значение для толщины;
- unsigned count: значение для количества;
- enum pattern\_of\_ornament pattern: перечисления для паттернов орнамента;
- enum angle angle.

### **1.3 Структура options\_mask\_t**

Вспомогательная структура, которая предназначена для хранения служебной информации о заполненности основной структуры options\_t.

- int mask\_func\_name;
- int mask\_input\_filename;
- int mask\_output\_filename;
- int mask\_left\_up;
- int mask\_right\_down;
- int mask\_color;
- int mask\_fill;
- int mask\_fill\_color;
- int mask\_thickness;
- int mask\_count;
- int mask\_pattern;
- int mask\_angle.

Все поля используются одинаково, хранят информация о заполненности структуры. Решение принято до становления идеи хранения информации в битовом виде в одной переменной, с другой стороны, с таким количеством переменных, это было бы затруднительно.

#### **1.4 Структура `point_t`**

Вспомогательная структура, которая предназначена для хранения двух точек в знаковом формате.

- `int x`;
- `int y`.

#### **1.5 Структура `rgb_color_t`**

Вспомогательная структура, которая предназначена для хранения цвета в формате RGB, содержит переменные в беззнаковом формате, потому что значение цвета не может быть отрицательным.

- `unsigned char r`;
- `unsigned char g`;
- `unsigned char b`.

#### **1.6 Структура `line_t`**

Вспомогательная структура, которая предназначена для хранения информации о линии.

- `point_t point_1`: координаты первой точки;
- `point_t point_2`: координаты второй точки;
- `unsigned thickness`: толщина линии;
- `rgb_color_t color`: цвет линии.



### **1.7 Перечисление fill**

Перечисление для хранения информации о том, нужна ли заливка.

Принимает одно из значений:

- FILLON = 0;
- FILLOFF = 1.

### **1.7 Перечисление pattern\_of\_ornament**

Перечисление для хранения информации о том, какой паттерн выбрал пользователь.

Принимает одно из значений:

- NOPATTERN = 0;
- RECTANGLE = 1;
- CIRCLE = 2;
- SEMICIRCLES = 3.

### **1.8 Перечисление angle**

Перечисление для хранения информации о том, на какой угол нужно повернуть часть изображения.

Принимает одно из значений:

- NOARGS = 0;
- ANGLE90 = 1;
- ANGLE180 = 2;
- ANGLE270 = 3.

### **1.9 Перечисление error\_marker\_t**

Перечисление для хранения информации об ошибке во время выполнения программы или ее отсутствии.

Принимает одно из значений:

- ERR\_OK = 0;
- ERR\_MEM = 40;
- ERR\_NOTPNG = 41;

- ERR\_INCORRECTARG = 42;
- ERR\_EXTRARGS = 43;
- ERR\_FEWARGS = 44;
- ERR\_LOTSOFFUNC = 45;
- ERR\_FILENAME = 46;
- ERR\_READING = 47;
- ERR\_WRITING = 48.

## 2. ФУНКЦИИ

### 2.1 Функция `main`

Функция `main` является главной функцией программы, она определяет ход выполнения других основных функций. Принимает аргументы `argc` (количество аргументов командной строки) и `argv` (массив строк, содержащий аргументы командной строки), возвращает значение перечисления `error_marker_t`, которое, если в процессе выполнения не было ошибок, равно 0.

Внутри функции выполняются следующие основные шаги:

- Создается структура для записи аргументов `options_t *struct_for_options` и переменной перечисления `returned_value` присваивается значение `ERR_OK`;
- Если структура успешно создана, она передается в функцию `get_options()`, где будут получены аргументы и в случае их некорректности или при других ошибках, перечислению `error_marker_t` будет присвоено ненулевое значение. В противном случае, возвращаемому значению присваивается значение `ERR_MEM` не равное нулю;
- Если в перечислении `error_marker_t` нет кода ошибки, то есть аргументы считаны корректно и сами они валидны, то создается переменная `png_t image` и сразу передается функции `read_png_file()`;
- Если после чтения файла в перечислении `error_marker_t` по прежнему нулевое значение, выводится приветственное сообщение, далее запускаются функции `process_the_image()` и `write_png_file()`;
- переменной `returned_value` присваивается значение перечисления внутри структуры для опций, после этого освобождается память выделенная под структуру;
- Возвращается переменная `returned_value`. На этом работа функции завершена.

## 2.2 Подготовительные функции

Функция `get_options(options_t *struct_for_options, int argc, char* argv[])` обрабатывает аргументы командной строки, переданные программе, и соответствующим образом заполняет структуру `options_t`. Передается количество аргументов командной строки `argc`, массив строк, содержащий аргументы командной строки `argv`, и указатель на структуру `options_t`, где будут храниться разобранные аргументы. Внутри этой функции происходит и валидация переданных аргументов с помощью вспомогательных функций: `is_options_correct(options_mask_t* maskopt, options_t* opt)`; `is_color_correct(char* color)`; `is_number_correct(char* coordinate)`.

Функции `get_struct_for_options()` и `get_mask_to_struct_of_options()` нужны для получения основных структур для дальнейшей работы программы.

## 2.3 Функции для работы с файлами

Функции `read_png_file(options_t *opt, png_t *image)` и `write_png_file(options_t *opt, png_t *image)` нужны для чтения и записи файла соответственно, `read_png_file` записывает информацию о файле в структуру `image`, а `write_png_file` записывает обновленную информацию из этой структуры обратно в файл.

## 2.4 Функции выполнения заданий

Функция `process_the_image(options_t* processing_options, png_t* png_image)` определяет какую функцию нужно выполнить в зависимости от значения в структуре опций.

Функция `print_help(int func_name)` печатает справочную информацию в зависимости от выбранной функции, либо, если переданное значение не соответствует ни одной из функций (например, если пользователь не ввел аргументы), печатает полную справку.

Функция `print_info(png_t* image)` печатает информацию о переданном изображении.

Функция `draw_rectangle(png_t* image, options_t* processing_options)` выполняет задачу (1) из условия.

Функция `draw_ornament(png_t *img, options_t *opt)` выполняет задачу (2) из условия.

Функция `rotate_region(png_t *img, options_t *opt)` выполняет задачу (3) из условия.

## **2.5 Функции рисования и вспомогательные для них**

Вспомогательные:

- `int get_min(int p_1, int p_2)`: получение меньшей из двух координат;
- `int get_max(int p_1, int p_2)`: получение большей из двух координат;
- `double get_distance(point_t left_up, point_t right_down)`: получение дистанции между двумя точками;
- `void get_correct_lu_rd_points(point_t* left_up, point_t* right_down)`: перестановка точек, если пользователь не понимает где лево/право верх/низ;
- `void get_correct_points_for_rect_with_thic(point_t* right_up, point_t* left_up, point_t* right_down, point_t* left_down, unsigned thickness)`: получение корректных точек для рисования прямоугольника;
- `int get_radius_for_semicircle_ornament(int length, unsigned count)`: получения значения радиуса для орнамента из окружностей;
- `void copy_region(png_t* img, options_t* opt, png_t* region)`: копирует указанную в `opt` область из `img` в `region`;
- `png_t* create_png(int width, int height)`: создает пустую структуру `png_t`;
- `void delete_png(png_t* img)`: удаляет структуру `png_t`;
- `rgb_color_t get_color_from_pixel(png_t *img, point_t point)`: получение цвета пикселя.

Основные:

- `void draw_point_if_valid(png_t* img, rgb_color_t color, int x, int y)`: Рисует точку, если координаты допустимы;

- void draw\_horizontal\_line(png\_t\* img, line\_t\* line): Рисует горизонтальную линию;
- void draw\_vertical\_line(png\_t\* img, line\_t\* line): Рисует вертикальную линию;
- void draw\_line(png\_t\* img, line\_t\* line): Рисует произвольную линию (либо вертикальную, либо горизонтальную);
- void flood\_fill(png\_t\* img, options\_t\* opt, point\_t left\_up, point\_t right\_down): Заливка области цветом;
- void draw\_rectangle(png\_t\* img, options\_t\* opt): Рисует прямоугольник;
- void draw\_rect\_ornament(png\_t\* img, options\_t\* opt): Рисует орнамент из прямоугольников;
- void draw\_circle(png\_t\* img, rgb\_color\_t color, point\_t center, int radius, int thickness): Рисует окружность;
- void draw\_semicircle\_ornament(png\_t\* img, options\_t\* opt): Рисует орнамент из окружностей;
- void draw\_circle\_ornament(png\_t\* img, options\_t\* opt): Рисует орнамент в форме окружности;
- void draw\_ornament(png\_t\* img, options\_t\* opt): Основная функция флага – ornament, из нее запускаются draw\_circle\_ornament(), draw\_rect\_ornament(), draw\_semicircle\_ornament();
- void copy\_region(png\_t\* img, options\_t\* opt, png\_t\* region): Копирует область изображения;
- void rotate\_90(png\_t\* dest, png\_t\* src): Поворачивает изображение на 90°;
- void rotate\_region(png\_t\* img, options\_t\* opt): Поворачивает область изображения.

Разработанный программный код см. в приложении Б.

## ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была разработана программа на языке C для обработки изображений в формате PNG с возможностью передачи аргументов через командный интерфейс (CLI). Основной целью проекта являлась реализация функционала для рисования прямоугольников, наложения орнаментальных рамок и поворота областей изображения с соблюдением требований к корректной работе с форматом PNG и отказоустойчивости. Для достижения поставленной цели были применены следующие методы:

1. Использование библиотеки `libpng` для чтения и записи PNG-файлов;
2. Реализация парсинга аргументов командной строки через `getopt_long` с поддержкой сокращенных и полных ключей;
3. Организация данных через специализированные структуры (`png_t`, `options_t`), обеспечивающие удобное хранение метаданных и параметров обработки;
4. Разработка алгоритмов для рисования геометрических фигур с настраиваемыми параметрами (толщина, цвет, заливка и т.д.), поворота фрагментов изображения и генерации узоров.

Ключевые результаты:

1. Программа корректно обрабатывает все случаи, включая ошибки формата файлов, некорректные аргументы и выход за границы изображения;
2. Реализованы три основные функции;
3. Обеспечена модульность кода: функции чтения/записи, обработки изображений и CLI логически разделены, что упрощает расширение функционала.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Керниган Б., Ритчи Д. Язык программирования С.: Диалектика, 2020. 288 с.
2. Мартин Р. Чистый код: создание, анализ и рефакторинг.: Питер, 2025. 464 с.
3. Примеры использования библиотеки libpng // github. URL: <https://github.com/pnggroup/libpng/blob/libpng16/example.c> (дата обращения с 01.04.2025 по 10.05.2025).
4. Документация библиотеки libpng // libpng manual. URL: <http://www.libpng.org/pub/png/libpng-1.2.5-manual.html> (дата обращения с 01.04.2025 по 10.05.2025).
5. Базовые сведения к выполнению курсовой работы по дисциплине «Программирование». Второй семестр / сост.: М. М. Заславский, А. А. Лисс, А. В. Гаврилов и др.. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2024. 36 с.



## Приложение А.

### ПРИМЕРЫ РАБОТЫ ПРОГРАММЫ

Таблица 1 – Изображения для примеров





| Изображение image_1.png   | Изображение image_2.png  |
|---|--|
|  |  |

Таблица 2 – Иллюстрация работы функций rotate, ornament, rectangle, info (в т.ч. ошибочный вывод)

| Входные данные  | Выходные данные  |
|---|--|
| <pre>./cw --left_up 110.343 --right_down<br/>415.456 --input ./image_1.png --<br/>output ./output.png --rotate --angle 90</pre> |  |

|   |   |
|---|---|
| <pre>./cw --color 102.205.170 --input ./image_2.png --output ./output. png --ornament --pattern semicircles -t 6 --count 10</pre> |   |
| <pre>./cw --rect -l 20.20 -r 480.380 -c 255.255.0 -t 8 1746266976_2195_0i.png</pre>   |  |
| <pre>./cw --info 1746266976_2195_0i.png</pre>   | <p>picture resolution (hwxw): 400x500<br/>bit depth = 8, color type = 2</p>         |
| <pre>./cw --color 102.205.170 --input ./1746266976_2195_0i.png -- output ./output. png --ornament</pre>                           | <p>Exit code: 44 (ERR_FEWARGS)</p>  |

**Приложение Б.**  
**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл main.c:**

// Place your code here

**Файл func.h:**

// Place your code here