**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

Курсовая РАБОТА

**по дисциплине «Программирование»**

Тема: Обработка BMP-изображений

**Вариант 6**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 1382 |  | Рымарь М.И. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2022

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студентка Рымарь М.И. | | |
| Группа 1382 | | |
| Тема работы: Обработка BMP-файлов | | |
| Исходные данные:  Программа принимает на вход аргументы и изображение в формате BMP. Необходимо преобразовать картинку в соответствии с условиями и сохранить изменившуюся копию. Поддержка ведётся через терминальный интерфейс (CLI – Command Line Interface). | | |
|  | | |
| Предполагаемый объем пояснительной записки:  Не менее 15 страниц. | | |
| Дата выдачи задания: 22.03.2022 | | |
| Дата сдачи реферата: 28.08.2022 | | |
| Дата защиты реферата: 01.09.2022 | | |
| Студентка |  | Рымарь М.И. |
| Преподаватель |  | Жангиров Т.Р. |

**Аннотация**

В ходе выполнения курсовой работы была создана программа на языке программирования С, которая обрабатывает BMP-файл. Программа имеет CLI (Command Line Interface) с помощью библиотеки getopt.h. CLI даёт возможность вывода справки о программе, информации о файле, а также реализуемых функциях, их аргументах и примерах ввода. Программа поддерживает BMP-файлы третьей версии, глубиной кодирования 24 бита, без сжатия.

**Summary**

During the course work, a program was developed in the C programming language that processes a BMP file. The program has a CLI (Command Line Interface) using the getopt.h library. The CLI makes it possible to output help about the program, information about the file, as well as implemented functions, their arguments and input examples. The program supports BMP files of the third version, encoding depth of 24 bits, without compression.

**содержание**

|  |  |  |
| --- | --- | --- |
| 1. | Введение | 5 |
| 1.1. | Задание курсовой работы | 6 |
| 1.2. | Цели и задачи работы | 7 |
| 2. | Ход выполнения работы | 8 |
| 2.1.  2.2.  2.3.  2.4. | Заголовочные файлы  Структуры  Интерфейс командной строки  Реализуемые функции | 8  8  9  9 |
| 3. | Тестирование | 11 |
| 4. | Заключение | 0 |
| 5. | Список использованных источников | 0 |
| 6. | Приложение А. Исходный код | 0 |
|  |  |  |
|  |  |  |
|  |  |  |

**введение**

Кратко описать цель работы, основные задачи и методы их решения. Требуется написать программу, которая производит выбранную пользователем обработку изображения.

Программа разработана на базе ОС Windows 11 в IDE Clion. Тестирование велось в терминале Ubuntu, встроенном в IDE.

Программа предоставляет CLI – терминальный интерфейс. Все случаи некорректного ввода команд отлавливаются; выводятся соответствующие ошибки, предупреждения и рекомендации.

**1.1. Задание курсовой работы**

*Вариант 6*

Общие сведения

* 24 бита на цвет
* без сжатия
* файл всегда соответствует формату BMP (но стоит помнить, что версий у формата несколько)
* обратите внимание на выравнивание; мусорные данные, если их необходимо дописать в файл для выравнивания, должны быть нулями.
* обратите внимание на порядок записи пикселей
* все поля стандартных BMP заголовков в выходном файле должны иметь те же значения что и во входном (разумеется, кроме тех, которые должны быть изменены).

Программа должна реализовывать следующий функционал по обработке bmp-файла:

(1) Рисование отрезка. Отрезок определяется:

координатами начала

координатами конца

цветом

толщиной

(2) Инвертировать цвета в заданной окружности. Окружность определяется либо координатами левого верхнего и правого нижнего угла квадрата, в который она вписана, либо координатами ее центра и радиусом.

(3) Обрезка изображения. Требуется обрезать изображение по заданной области. Область определяется:

Координатами левого верхнего угла

Координатами правого нижнего угла

**1.2. Цели и задачи работы**

Цель: ознакомиться с особенностями работы с BMP файлами и написать программу, которая выполняет определённую обработку изображений, заданную пользователем с помощью Command Line Interface.

Задачи:

* Реализовать корректное считывание и хранение bmp-файла;
* Осуществить правильную обработку запросов пользователя, используя CLI;
* Выполнить обработку возможных ошибок;
* Создать следующие функции: рисование отрезка, инвертирование цветов в заданной окружности и обрезка изображения.

**2. ХОД РАБОТЫ**

**2.1. Выполнение**

Для корректной работы программы подключаем следующие библиотеки:

1. *<getopt.h>* - обеспечивает command line interface, удобный для пользователя. Функции *getopt* и *getopts\_long* автоматизируют часть рутинной работы, связанной с анализом типичных параметров командной строки;
2. *<stdio.h>* - заголовочный файл, который обеспечивает считывание данных, введённых пользователем, и работу с самим bmp-файлом (используемые функции*: fopen(), fclose(), fread(), fwrite(), puts(), scanf(), printf(), sscanf()*);
3. *<stdlib.h>* - заголовочный файл, который содержит в себе функции выделения памяти (используемые функции: *malloc(), free()*);
4. *<string.h>* - заголовочный файл для работы со строками (используемая функция – *strlen()*);
5. *<math.h>* - заголовочный файл, который используется для выполнения простых математических операций (используемая функция – *fabs()*).

**2.2. Структуры**

Все структуры оборачиваем в *#pragma pack(push, 1)* и *#pragma pack(pop).*

Первое устанавливает размер выравнивания в 1 байт, второе возвращает предыдущую настройку. Без этого размер структур будет меняться в зависимости от компилятора.

Далее создаём структуры *BitmapFileHeader* и *BitmapInfoHeader*, которые содержат в себе поля, соответствующие выбранной версии формата *bmp*. Структура *RGB* содержит три поля, которые определяют один пиксель с помощью трёх цветов. В последней структуре image определены поля с информацией о файле – *bmfh, bmih* и массив пикселей *rgb*.

**2.3. Интерфейс командной строки**

Строка, которой пользователь вводит команды имеет вид:  
*./src -[ключ команды] [имя обрабатываемого файла] -o [файл для сохранения]*

*–[ключ1] [аргумент1] …*

Обработку запросов делаем в функции *main*: создаём строку коротких ключей и сохраняем в *opts*, создаём объекты структуры *option*, в которых определяем длинные и короткие ключи, из библиотеки *getopt.h*. Если введённых аргументов меньше двух, то выводим ошибку и вызываем функцию помощи *help\_output()*. Далее с помощью оператора *switch(opt)* прописываем случаи обработки для каждого введённого ключа, в том числе обрабатываем возможные ошибки. Дефолтное значение устанавливаем в виде вывода сообщения об ошибке: *“Wrong key”*. Если вызванный ключ является ключом для функции обработки, то в этом case добавляем в переменную *funcName* номер функции, чтобы вызывать её в отдельном операторе *switch(funcName)*. В этом операторе дефолтное значение устанавливаем в виде вывода об ошибке *“No function called”.*

**2.4. Реализуемые функции**

Нужно упомянуть, что в программе содержится две важные функции – считывание файла *readImage()* и *saveImage()*. Также была создана дополнительная функция, которая проверяет файл на корректность и выводит ошибки-подсказки в обратном случае – *ifCorrect()*. Теперь перейдём к основным реализуемым операциям по обработке изображений:

1. Обрезка изображения

Данная операция осуществляется с помощью функции *void cut()*. В неё передаются 6 аргументов: обрабатываемый файл (структуры *image)*, название файла, в котором нужно сохранить обработанный файл, 4 координаты - *x1,y1,x2,y2*. Обработка возможных ошибок производится с помощью двух условий: первое – вторая пара координат не должна быть больше первой, второе – ни одна из координат не должна выходить за границы файла. Если данные введены корректно, то создаём новый файл, в который сохраняем только те пиксели, которые останутся внутри прямоугольника из заданных координат.

2. Инвертирование цвета в заданной окружности

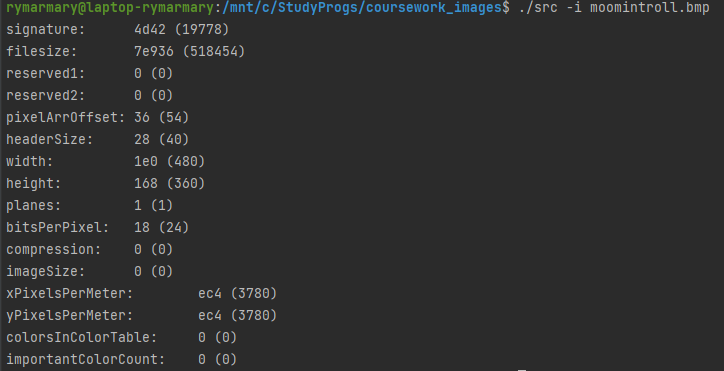
Данная операция осуществляется с помощью функции *void paintOverTheCircle().* В неё передаётся 5 аргументов: обрабатываемый файл (структуры *image)*, название файла, в котором нужно сохранить обработанный файл, 3 координаты - *x,y,R.* Обработка ошибок осуществляется подобным образом, как и в функции обрезки файла – с помощью двух условий. Первое проверяет, находятся ли переданные координаты центра за границами файла. Второе условие проверяет, не выходят ли за границы файла *(x+R)* и *(x-R)*. Если программа не вывела ошибку, то, проходя по вложенному циклу, инвертирует цвета каждого пикселя, предварительно проверив, находятся ли эти пиксели внутри заданной окружности.

3. Рисование отрезка

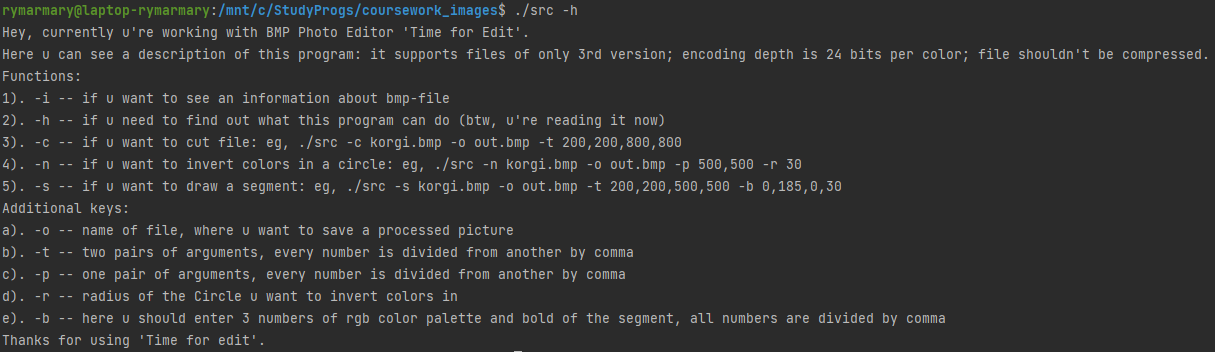
Данная операция осуществляется с помощью одной основной функции *void drawSegment()* и двух побочных – *int accuracy()* и *int isLine()*. Начнём с двух побочных: вторая проверяет, могут ли введённые данные образовать линию, в этой же функции вызывается первая, которая проверяет, не превышают ли введённые аргументы заданную погрешность в виде половины ширины линии. Далее в основной функции обрабатываем ошибки: не выходят ли заданные координаты начала и конца отрезка за границу файла, правильно ли введены цвета (должны быть от 0 до 255), также проверяется толщина линии – она не должна быть меньше одного пикселя и не должна быть больше длины или ширины картинки.

**3. ТЕСТИРОВАНИЕ**

1. Вывод информации о файле:

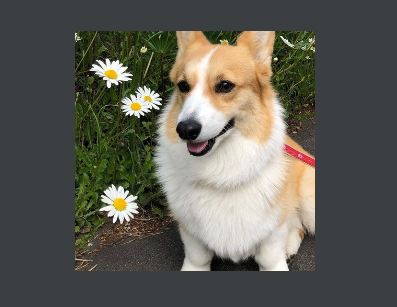


2. Вывод помощи:



3. Обрезка изображения:





4. Инвертирование цвета в заданной окружности:





**заключение**

Кратко подвести итоги, проанализировать соответствие поставленной цели и полученного результата.

**список использованных источников**

1. Статья в Wikipedia. URL: <https://en.wikipedia.org/wiki/Getopt> (дата обращения: 01.08.2022)

11. Описание электронного ресурса // Наименование сайта. URL: http://east-front.narod.ru/memo/latchford.htm (дата обращения: 00.00.2010).

12. ГОСТ 0.0–00. Описание стандартов. М.: Изд-во стандартов, 2010.

13. Пат. RU 00000000. Описание патентных документов / И. И. Иванов, П. П. Петров, С. С. Сидоров. Опубл. 00.00.2010. Бюл. № 00.

14. Иванов И.И. Описание авторефератов диссертаций: автореф. дисс. канд. техн. наук / СПбГЭТУ «ЛЭТИ», СПБ, 2010.

15. Описание федерального закона: Федер. закон [принят Гос. Думой 00.00.2010] // Собрание законодательств РФ. 2010. № 00. Ст. 00. С. 000–000.

16. Описание федерального постановления: постановление Правительства Рос. Федерации от 00.00.2010 № 00000 // Опубликовавшее издание. 2010. № 0. С. 000–000.

17. Описание указа: указ Президента РФ от 00.00.2010 № 00 // Опубликовавшее издание. 2010. № 0. С. 000–000.

**приложение А**

**Исходный код**