**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

Курсовая РАБОТА

**по дисциплине «ПРОГРАММИРОВАНИЕ»**

Тема: **BMP-файлы**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 6303 |  | Матюшина М.Е. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студентка Матюшина М.Е. | | |
| Группа 6303 | | |
| Тема работы: BMP-файл | | |
|  | | |
| Содержание пояснительной записки:   * Содержание * Введение * Описание структур BMP-файла * Проверка на корректность введенные данные * Построение изображения * Примеры работы программы * Заключение * Список использованных источников * Приложение А. Исходный код программы | | |
| Предполагаемый объем пояснительной записки:  Не менее 10страниц. | | |
| Дата выдачи задания: 28.04.2017 | | |
| Дата сдачи реферата: 1.06.2017 | | |
| Дата защиты реферата: 1.06.2017 | | |
| Студентка |  | Матюшина М.Ю. |
| Преподаватель |  | Берленко Т.А. |

**Аннотация**

В ходе работы была реализована программа на языке программирования С, которая рисует прямоугольный равнобедренный треугольник черным цветом в заданной области BMP-файла. Приведено полное описание исходного кода.

**содержание**

|  |
| --- |
| **Введение**…………………………………………………………………………………………………....5.   1. **Описание структур BMP-файла** …………………………………………………………………….. 6.    1. BITMAPFILEHEADER ………………………………………………………………………………………6.    2. BITMAPINFOHEADER ………………………………………………………………………………………6.    3. RGBTRIPLE………………………………………………………………………………………………………………..…….7. 2. **Проверка на корректность введенных данных** …………………………………………….............7. 3. **Построение изображения**………………………………………………………………………….…....7.    1. Функция, для считывания длины и ширины изображения……………………………………………7    2. Нахождение координат вершин квадратной области относительно первого бита изображения….8.    3. Находим координаты середин сторон квадрата, чтобы узнать, где начинается треугольник……..8.    4. Запись пикселей………………………………………………………………………………...…….….8. 4. **Примеры работы программы** ……………………………………………………………….………...9 5. **Заключение** ……………………………………………………………………………………………..10 6. **Список использованных источников** …………………………………………………………….....11 7. **Приложение А. Полный код программы.** …………………………………………………………..12 8. **Приложение Б. Загрузка файла на GitHub…**…………………………………………………………………...16 |
|  |
|  |

# 

# **введение**

В ходе работы необходимо написать программу, которая рисует прямоугольный равнобедренный треугольник черным цветом в заданной области BMP-файла. Прямой угол должен лежать в левом нижнем углу области, длина катетов определяется длиной стороны области, к которой они прилегают, и равна половине этой стороны.

Программа получает параметры из входного потока и должна проверить их корректность (в том числе, что область является квадратной). Параметры:

* input\_file
* x0
* y0
* x1
* y1

* input\_file - имя BMP файла
* x0 y0 левый верхний угол области (отсчет с точки 0, 0)
* x1 y1 правый нижний угол области

В случае, если программа получила некорректные параметры, то:

* не создается выходного в файла
* выводится сообщение об ошибке “Fail with <имя параметра>”.

### Общие сведения

* 24 бита на цвет
* без сжатия
* файл всегда соответствует формату (проверять не нужно)
* 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).

# 

# **Описание структур BMP-файла**

Для удобства переименуем:

typedef unsigned short int WORD;

typedef unsigned long int DWORD;

typedef long int LONG;

typedef char BYTE;

* 1. **BITMAPFILEHEADER** — 14-байтная структура, которая располагается в самом начале файла. С учетом выравнивания 16 байт.

typedef struct tagBITMAPFILEHEADER {

WORD bfType; // сигнатура формата

DWORD bfSize; // Размер файла в байтах.

WORD bfReserved1; // Зарезервированы и должны содержать ноль.

WORD bfReserved2;

DWORD bfOffBits; // Положение пиксельных данных относительно начала данной структуры (в байтах

} BITMAPFILEHEADER;

* 1. **BITMAPINFOHEADER.**

В файле идёт сразу за BITMAPFILEHEADER.

Состоит из трёх частей:

1. Структура с информационными полями.
2. Битовые маски для извлечения значений цветовых каналов (присутствуют не всегда).
3. Таблица цветов (присутствует не всегда).

typedef struct tagBITMAPINFOHEADER

{

DWORD biSize; *// Размер данной структуры в байтах, указывающий также на версию структуры*

LONG biWidth; *// Ширина (bcWidth) и высота (bcHeight) растра в пикселях.* LONG biHeight; *//*

WORD biPlanes; *// В BMP допустимо только значение 1. Это поле используется в*

*// значках и курсорах Windows.*

WORD biBitCount; *// Количество бит на пиксель*

DWORD biCompression; *// Указывает на способ хранения пикселей*

DWORD biSizeImage; *// Размер пиксельных данных в байтах*

LONG biXPelsPerMeter; *// Количество пикселей на метр по горизонтали*

LONG biYPelsPerMeter; // *и вертикали*

DWORD biClrUsed; *// Размер таблицы цветов в ячейках.*

DWORD biClrImportant*; // Количество ячеек от начала таблицы цветов до последней*

*// используемой (включая её саму).*

} BITMAPINFOHEADER;

* 1. **RGBTRIPLE**

Глубина цвета 24 бит, по 1 байту на цвет.

typedef struct tagRGBTRIPLE {

BYTE rgbtBlue; *// синяя составляющая*

BYTE rgbtGreen; *// зеленая составляющая*

BYTE rgbtRed; *// красная составляющая*

} RGBTRIPLE;

1. **Проверка корректности введенных данных.**

* Проверка, что область является квадратной.

if ( (x1-x0)!= (y1-y0) ) {

printf("Fail with square\n");

return 0;

}

* Проверка существования исходного файла

f1 = fopen(name, "rb");

if (f1==NULL) {

printf("Fail with file\_start \n" );

return 0;

}

* Проверка корректности введенных координат

if ( (x0<0)||(x1<0)||(y0<0)||(y1<0)|| (x1>heig) || (y1>wid) ) {

printf("Fail with coordinates\n");

return 0;

}

* Проверка существования файла для записи

f2 = fopen("new.bmp", "wb");

if (f2==NULL) {

printf("Fail with new\_file»" );

return 0;

}

1. **Построение изображения.**
   1. **Функция, для считывания длины и ширины изображения**

void Research\_size(char\* name, long int\* h, long int\* w) {

FILE \* f1;

BITMAPFILEHEADER BFH;

BITMAPINFOHEADER BIH;

f1 = fopen(name, "rb");

fread(&BFH,14,1,f1);

fread(&BIH,sizeof(BIH),1,f1);

\*h = BIH.biHeight;

\*w = BIH.biWidth;

fclose(f1);

}

Принимает имя файла, который необходимо считать и указатели на переменные, в которых хранится высота и ширина изображения.

Далее в функции main считываем BITMAPFILEHEADER BFH и BITMAPINFOHEADER BIH исходного файла и записываем в BITMAPFILEHEADER BFH и BITMAPINFOHEADER BIH конечного файла соответственно.

* 1. **Нахождение координат вершин квадратной области относительно первого бита изображения.**

Так как с консоли вводятся координаты относительно левого верхнего бита изображения, а в файле нумерация идет с левого нижнего бита, следует привести введенные данные к нужному для создания алгоритма виду.

int x0\_new=heig-x0;

int x1\_new=heig-x1;

* 1. **Находим координаты середин сторон квадрата, чтобы узнать, где начинается треугольник.**

int mid\_x = x0\_new+(x1\_new-x0\_new)/2;

int mid\_y=y0+(y1-y0)/2;

* 1. **Запись пикселей.**

Так как количество байт в строке должно быть кратно 4, создаем переменную padding, равную количеству байт, которые неоходимо заполнить нулями для выравнивания.

size\_t padding = 0;

if ((wid \* 3) % 4) {

padding = 4 - (wid \* 3) % 4;

}

Далее начинаем записывать пиксели. Если пиксель попадает в область треугольника, делаем его черным, присваивая составляющим RGB нули. Область треугольника находим с учетом того, что сумма координат каждой его точки меньше либо равна значению сумм координат первой точки.

for( i=0; i < heig ;i++) {

for (j = 0; j < wid; j++){

fread(&RGB, sizeof(RGB),1, f1);

if ( i<=mid\_x && j<mid\_y && i>=x1\_new && j>=y0 && ((mid\_x-i)>=(j-y0 )) ) {

RGB.rgbtBlue=0;

RGB.rgbtGreen=0;

RGB.rgbtRed=0;

fwrite(&RGB, sizeof(RGB), 1, f2);

}

else{

fwrite(&RGB, sizeof(RGB), 1, f2);

}

}

if(padding != 0) {

fread(&RGB, padding,1, f1);

fwrite(&RGB, padding, 1, f2);

}

}

Закрываем файлы, задача выполнена.

1. **Пример работы программы.**

Ввод данных:

Введите название исходного файла: 1.bmp

x0 = 100

y0 = 200

x1 = 700

y1 = 800

Результат: файл new.bmp - копия исходного файла 1.bmp с черным треугольником в левом нижнем углу указанной области.

2.

Ввод данных:

Введите название исходного файла: 1.bmp

x0 = 100

y0 = 200

x1 = 700

y1 = 700

Результат: вывод фразы Fail with square

Файл new.bmp не изменился.

# 

# **заключение**

В ходе работы была реализована программа на языке программирования С, которая рисует прямоугольный равнобедренный треугольник черным цветом в заданной области BMP-файла. Приведено полное описание исходного кода. Код и отчет по работе были загружены в репозиторий.

# **список использованных источников**

1. Язык программирования СИ / Керниган Б., Ритчи Д. СПб.: Издательство"Невский Диалект", 2001. 352 с.

# **приложение А**

# **Исходный код программы**

#include<stdio.h>

#include<stdlib.h>

typedef unsigned short int WORD;

typedef unsigned long int DWORD;

typedef long int LONG;

typedef char BYTE;

typedef struct tagBITMAPFILEHEADER {

WORD bfType;

DWORD bfSize;

WORD bfReserved1;

WORD bfReserved2;

DWORD bfOffBits;

} BITMAPFILEHEADER, \*PBITMAPFILEHEADER;

typedef struct tagBITMAPINFOHEADER

{

DWORD biSize;

LONG biWidth;

LONG biHeight;

WORD biPlanes;

WORD biBitCount;

DWORD biCompression;

DWORD biSizeImage;

LONG biXPelsPerMeter;

LONG biYPelsPerMeter;

DWORD biClrUsed;

DWORD biClrImportant;

} BITMAPINFOHEADER, \*PBITMAPINFOHEADER;

typedef struct tagRGBTRIPLE {

BYTE rgbtBlue;

BYTE rgbtGreen;

BYTE rgbtRed;

} RGBTRIPLE;

void Research\_size(char\* name, long int\* h, long int\* w) {

FILE \* f1;

BITMAPFILEHEADER BFH;

BITMAPINFOHEADER BIH;

f1 = fopen(name, "rb");

fread(&BFH,14,1,f1);

fread(&BIH,sizeof(BIH),1,f1);

\*h = BIH.biHeight;

\*w = BIH.biWidth;

fclose(f1);

}

int main() {

char\* name = (char\*)malloc(100\*sizeof(char));

printf("Введите название исходного файла ");

scanf("%s", name);

int x0, y0, x1, y1;

printf("x0 = ");

scanf("%d", &x0);

printf("y0 = ");

scanf("%d", &y0);

printf("x1 = ");

scanf("%d", &x1);

printf("y1 = ");

scanf("%d", &y1);

//проверка на квадратность области

if ( (x1-x0)!= (y1-y0) ) {

printf("Fail with square\n ");

return 0;

}

FILE \* f1, \* f2;

BITMAPFILEHEADER BFH;

BITMAPINFOHEADER BIH;

RGBTRIPLE RGB;

f1 = fopen(name, "rb");

// проверка существования исходного файла

if (f1==NULL) {

printf("Fail with file\_start \n" );

return 0;

}

long int heig, wid;

Research\_size(name,&heig, &wid);

// Проверка корректности введенных координат

if ( (x0<0)||(x1<0)||(y0<0)||(y1<0)|| (x1>heig) || (y1>wid) ) {

printf("Fail with coordinates\n");

return 0;

}

//новые координаты

int x0\_new=heig-x0;

int x1\_new=heig-x1;

int mid\_x = x0\_new+(x1\_new-x0\_new)/2;

int mid\_y=y0+(y1-y0)/2;

f2 = fopen("new.bmp", "wb");

// Проверка существования файла для записи

if (f2==NULL) {

printf("Fail ("Fail with new\_file\n" );

return 0;

}

fread(&BFH,16,1,f1);

fwrite(&BFH, sizeof(BFH), 1, f2);

fread(&BIH,sizeof(BIH),1,f1);

fwrite(&BIH, sizeof(BIH), 1, f2);

int i, j;

size\_t padding = 0;

if ((wid \* 3) % 4) {

padding = 4 - (wid \* 3) % 4;

}

for( i=0; i < heig ;i++) {

for (j = 0; j < wid; j++){

fread(&RGB, sizeof(RGB),1, f1);

if ( i<=mid\_x && j<mid\_y && i>=x1\_new && j>=y0 && ((mid\_x-i)>=(j-y0 )) ) {

RGB.rgbtBlue=0;

RGB.rgbtGreen=0;

RGB.rgbtRed=0;

fwrite(&RGB, sizeof(RGB), 1, f2);

}

else{

fwrite(&RGB, sizeof(RGB), 1, f2);

}

}

if(padding != 0) {

fread(&RGB, padding,1, f1);

fwrite(&RGB, padding, 1, f2);

}

}

fclose(f1);

fclose(f2);

free(name);

return 0;

}

**ПРИЛОЖЕНИЕ Б. Загрузка файла на GitHub.**

* В локальном репозитории была создана новая ветка matyushina\_2sem\_kurs

git checkout -b matyushina\_2sem\_ kurs

* Создана директория matyushina\_2sem\_ kurs

mkdir matyushina\_2sem\_ kurs

* В вышеуказанную папку был перемещен файл Kurs.c
* Исходный файл добавлен в локальный репозиторий

git add Kurs.c

* Командой <git commit –m> были зафиксированы изменения в проекте

git commit –m “matyushina\_2sem\_ kurs”

* Командой <git push> последний commit был отправлен на удаленный репозиторий

git push origin matyushina\_2sem\_ kurs