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

КУРСОВАЯ РАБОТА

по дисциплине «Программирование»

Тема: Обработка bmp-файла

Студент гр. 7304 _____ Нгуен К.Х

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2017

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студент Нгуен К.Х.

Группа 7304

Тема работы: Обработка bmp-файла на языке программирования Си

Исходные данные:

Полученный алгоритм решения должен быть насколько возможно эффективным и проходить тесты с любыми исходными параметрами

Содержание пояснительной записки:

1. Аннотация
2. Введение
3. Структура элемента списка
4. Функции для работы со списком
5. Демонстрационное использование
6. Заключение
7. Исходный код

Предполагаемый объем пояснительной записки:

Не менее 18 страниц.

Дата выдачи задания: 24.11.2017

Дата сдачи реферата:

Дата защиты реферата:

Студент _____ Нгуен К.Х

Преподаватель _____ Верленко Т.А.

Аннотация

В данной кусовой работе будет представлен один из вариантов реализации функции работы со BMP файлах. Для проверки работы были проведены не сколько тестов с разным элементами.

В курсовой работе представлены отдельные части програмы и их объяснение, а также скриншоты результатов тестов программы.

Содержание

Введение

BMP (Bitmap Picture), как следует из названия, является форматом, используемым для хранения растровых цифровых изображений. Формат BMP-файл позволяет хранить двумерные цифровые изображения как монохромные, так и цветные, а также, возможно, с сжатием данных, альфа-каналами и цветовыми профилями.

Цель работы

Целью данной работы является изучение структуры bmp-файла и реализация несколько функций по обработке bmp файла.

Задачи

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

- Рисование квадрата. Квадрат определяется:
  - Координатами левого верхнего угла
  - Размером стороны
  - Толщиной линий
  - Цветом линий
  - Может быть залит или нет
  - Цветом которым он залит, если пользователем выбран залитый
- Поменять местами 4 куска области. Выбранная пользователем прямоугольная область делится на 4 части и эти части меняются местами. Функционал определяется:
  - Координатами левого верхнего угла области
  - Координатами правого нижнего угла области
  - Способом обмена частей: “по кругу”, по диагонали
- Находит самый часто встречаемый цвет и заменяет его на другой заданный цвет. Функционал определяется
  - Цветом, в который надо перекрасить самый часто встречаемый цвет.

Основная структура работы

namespace llib

- Structures
  - **FileHeader**
  - **Header**
  - **Color**
- namespace BMPLib
  - Functions
    - **readBitmap**
    - **writeBitmap**
    - **drawRectangle**
    - **swapRegions**
    - **replaceMostCommon**

Структуры

FileHeader	
header_field	unsigned short
file_size	unsigned int
reserved1	unsigned short
reserved2	unsigned short
bitmap_offset	unsigned int

Header	
file_header	FileHeader
header_size	unsigned int
img_width	unsigned int
img_height	unsigned int
planes	unsigned short
bpp	unsigned short
compression	unsigned short
img_size	unsigned int
res_horizontal	int
res_vertical	int
color_palette	unsigned int
important_colors	unsigned int

Эти структуры построены на основе описания BITMAPFILEHEADER и Windows BITMAPINFOHEADER

BITMAPFILEHEADER

BITMAPFILEHEADER — 14-байтная структура, которая располагается в самом начале файла. Обратите внимание на то, что с самого начала структуры сбивается выравнивание ячеек. Если для вас оно важно, то в оперативной памяти данный заголовок располагайте по чётным адресам, которые не кратны четырём (тогда 32-битные ячейки попадут на выравненные позиции).

Поз. (hex)	Размер (байты)	Имя	Тип WinAPI	Описание
00	2	bfType	WORD	Отметка для отличия формата от других (сигнатура формата). Может содержать единственное значение $4D42_{16}$/$424D_{16}$ (little-endian/big-endian).
02	4	bfSize	DWORD	Размер файла в байтах.
06	2	bfReserved1	WORD	Зарезервированы и должны содержать ноль.
08	2	bfReserved2	WORD	
0A	4	bfOffBits	DWORD	Положение пиксельных данных относительно начала данной структуры (в байтах).

Windows BITMAPINFOHEADER

Позиция в файле (hex)	Позиция в структуре (hex)	Размер (байты)	Имя (версии 3/4/5)	Тип WinAPI	Описание
0E	00	4	biSize bV4Size bV5Size	DWORD	Размер данной структуры в байтах, указывающий также на версию структуры (см. таблицу версий выше).
12	04	4	biWidth bV4Width bV5Width	LONG	Ширина растра в пикселях. Указывается целым числом со знаком. Ноль и отрицательные не документированы.
16	08	4	biHeight bV4Height bV5Height	LONG	Целое число со знаком, содержащее два параметра: высота растра в пикселях (абсолютное значение числа) и порядок следования строк в двумерных массивах (знак числа). Нулевое значение не документировано.
1A	0C	2	biPlanes bV4Planes bV5Planes	WORD	В BMP допустимо только значение 1. Это поле используется в значках и курсорах Windows.

1C	0E	2	biBitCount bV4BitCount bV5BitCount	WORD	Количество бит на пиксель (список поддерживаемых смотрите в отдельном разделе ниже).
1E	10	4	biCompression bV4V4Compression[11] bV5Compression	DWORD	Указывает на способ хранения пикселей (см. в разделе ниже).
22	14	4	biSizeImage bV4SizeImage bV5SizeImage	DWORD	Размер пиксельных данных в байтах. Может быть обнулено если хранение осуществляется двумерным массивом.
26	18	4	biXPelsPerMeter bV4XPelsPerMeter bV5XPelsPerMeter	LONG	Количество пикселей на метр по горизонтали и вертикали (см. раздел «Разрешение изображения» данной статьи).
2A	1C	4	biYPelsPerMeter bV4YPelsPerMeter bV5YPelsPerMeter	LONG	
2E	20	4	biClrUsed bV4ClrUsed bV5ClrUsed	DWORD	Размер таблицы цветов в ячейках.
32	24	4	biClrImportant bV4ClrImportant bV5ClrImportant	DWORD	Количество ячеек от начала таблицы цветов до последней используемой (включая её саму).

Функции

```
void drawRectangle(Bitmap bitmap, const Vector *top_left,
const Vector *bot_right, Color *color, int thickness);
```

Функция рисует прямоугольник с заданными характеристиками

- bitmap - растровом изображении, на котором мы работаем
- top_left - координата верхнего левого угла прямоугольника
- bot_right - координата нижнего правого угла прямоугольника
- color - цвет, в котором будет окрашен прямоугольник
- thickness - толщина границы прямоугольника (массивная толщина, приводящая к заполненному прямоугольнику)

```
void BMPLib::drawRectangle(Bitmap bitmap, const Vector *top_left, const Vector *bot_right, Color
*color, int thickness){

        for(int i=bot_right->y;i<top_left->y;i++){
                for(int j=top_left->x;j<bot_right->x;j++){
```

```
                                        if((i-bot_right->y<thickness)||(top_left->y-
i<thickness+1)||(j-top_left->x<thickness)||(bot_right->x-j<thickness+1)){
                                                colorCopy(color,(*(bitmap+i)+j));
                                        }
                                }
                        }
        }
        void BMPLib::swapRegions(Bitmap bitmap, const Vector *top_left, const Vector *bot_right){
                Vector *pivot = (Vector*)malloc(sizeof(Vector));
                pivot->x=(bot_right->x+top_left->x)/2;
                pivot->y=(bot_right->y+top_left->y)/2;

                int w_region = pivot->x-top_left->x;
         int h_region = top_left->y-pivot->y;


                Color *temp = (Color*) malloc(sizeof(Color));
         for(int i=pivot->y;i<top_left->y;i++){
                        for(int j=top_left->x;j<pivot->x;j++){
                                colorCopy((*(bitmap+i)+j),temp);
                                colorCopy((*(bitmap+i)+j+w_region),(*(bitmap+i)+j));
                                colorCopy((*(bitmap+i-
h_region)+j+w_region),(*(bitmap+i)+j+w_region));
                                colorCopy((*(bitmap+i-h_region)+j),(*(bitmap+i-
h_region)+j+w_region));
                                colorCopy(temp,(*(bitmap+i-h_region)+j));
                        }
                }
                free(temp);
        }
```

```
void swapRegions(Bitmap bitmap, const Vector *top_left,
const Vector *bot_right);
```

Функция принимает область, обозначенную верхними и нижними углами, затем делит ее на 4 меньшие области, которые будут обмениваться по часовой стрелке.

- bitmap - Bitmap, на котором мы работаем
- top_left - координата верхнего левого угла прямоугольника
- bot_right - координата нижнего правого угла прямоугольника

```
void BMPLib::swapRegions(Bitmap bitmap, const Vector *top_left, const Vector *bot_right){
                Vector *pivot = (Vector*)malloc(sizeof(Vector));
                pivot->x=(bot_right->x+top_left->x)/2;
                pivot->y=(bot_right->y+top_left->y)/2;

                int w_region = pivot->x-top_left->x;
```

```
        int h_region = top_left->y-pivot->y;


              Color *temp = (Color*) malloc(sizeof(Color));
        for(int i=pivot->y;i<top_left->y;i++){
                    for(int j=top_left->x;j<pivot->x;j++){
                          colorCopy((*(bitmap+i)+j),temp);
                          colorCopy((*(bitmap+i)+j+w_region),(*(bitmap+i)+j));
                          colorCopy((*(bitmap+i-
h_region)+j+w_region),(*(bitmap+i)+j+w_region));
                          colorCopy((*(bitmap+i-h_region)+j),(*(bitmap+i-
h_region)+j+w_region));
                          colorCopy(temp,(*(bitmap+i-h_region)+j));
                    }
              }
              free(temp);
       }
```

```
void replaceMostCommon(Bitmap bitmap, const Vector
*img_size, Color *replace_by);
```

Функция заменяет наиболее распространенный цвет в растровом изображении с заданным цветом

- bitmap - растровом изображении, на котором мы работаем
- img_size - размер растрового изображения
- replace_by - цвет, который будет заполнен на месте наиболее распространенного цвета

```
void BMPLib::replaceMostCommon(Bitmap bitmap, const Vector *img_size, Color
*replace_by){

              Node *head=NULL;


              for(int i=0;i<img_size->y;i++){
                    for(int j=0;j<img_size->x;j++){
                          count(&head,(*(bitmap+i)+j));
                    }
              }
              Node *p=head;
              unsigned int max=0;
              Color *most_common = (Color*) malloc(sizeof(Color));
              while(p!=NULL){
                    if((p->count)>max){
                          max=p->count;
```

```
                        colorCopy(p->color,most_common);
                }
                p=p->next;
        }
        for(int i=0;i<img_size->y;i++){
                for(int j=0;j<img_size->x;j++){
                        if(colorEqual((*(bitmap+i)+j),most_common)){
                                colorCopy(replace_by,(*(bitmap+i)+j));
                        }

                }
        }
}
```

```
void readBitmap(const char* fileName, Header **header,
Bitmap *bitmap);
```

Прочитать файл, объявленный в fileName, в header и bitmap структуры

```
void BMPLib::readBitmap(const char* fileName, Header **header, Bitmap *bitmap){
        FILE *file;
        file = fopen(fileName,"rb+");
    if(file==NULL) {err_message=ERR_FILE_NOT_FOUND; return;}
    if((*header)==NULL)
        *header = (Header*) malloc(sizeof(Header));
        fread((*header),sizeof(Header),1,file);
    if((*header)->header_size!=40) {
        err_message=ERR_FILE_FORMAT;
        return;
    }
        int row_size = ((*header)->bpp*((*header)->img_width)+31)/32*4;

        fseek(file,((*header)->file_header).bitmap_offset,SEEK_SET);
    char* data = NULL;
    if((*bitmap)==NULL){
        data = (char*) malloc(sizeof(char)*row_size*((*header)->img_height));
        *bitmap = (Bitmap) malloc(sizeof(Color*)*((*header)->img_height));
    }
    for(unsigned int i=0;i<((*header)->img_height);i++){
                *((*bitmap)+i)=(Color*) ((void*)((char*)data+i*row_size));
                fread(*((*bitmap)+i),row_size,1,file);
        }
    for(unsigned int i=0;i<(*header)->img_height;i++){
        for(unsigned int j=0;j<(*header)->img_width;j++){
            unsigned char t = (*((*bitmap)+i)+j)->b;
            (*((*bitmap)+i)+j)->b = (*((*bitmap)+i)+j)->r;
            (*((*bitmap)+i)+j)->r=t;
        }
```

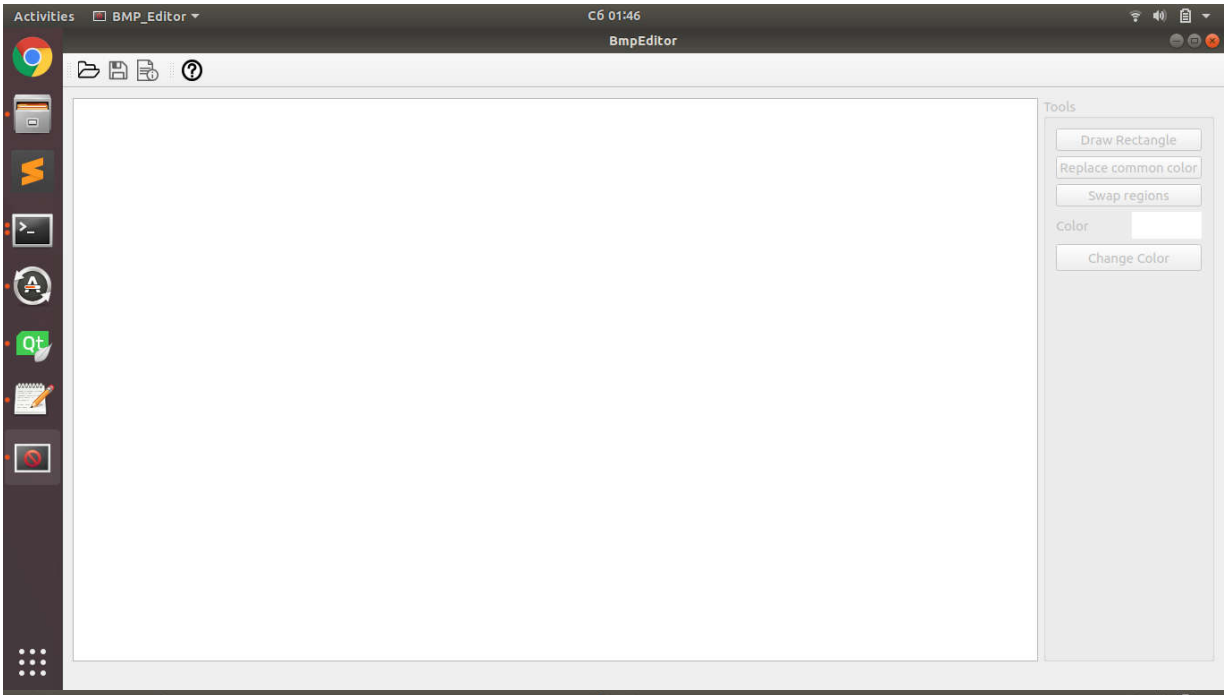
```
        }
        fclose(file);
    }
```

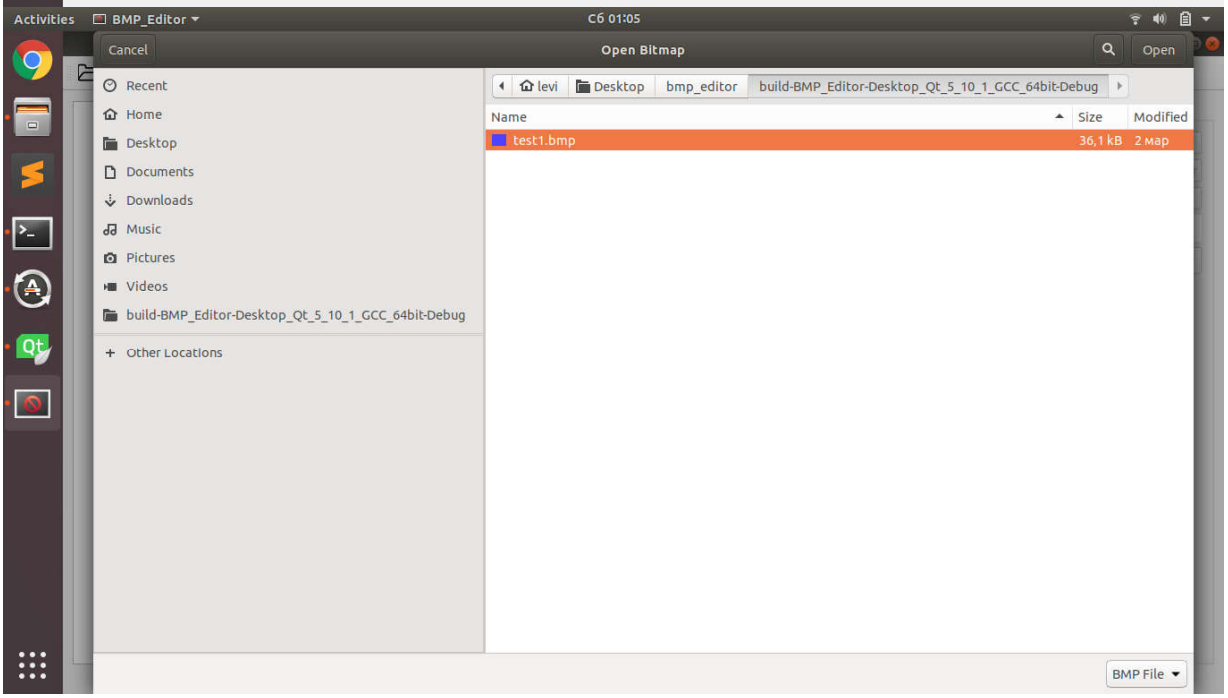
```
void writeBitmap(const char* fileName, Header *header,
Bitmap bitmap);
```

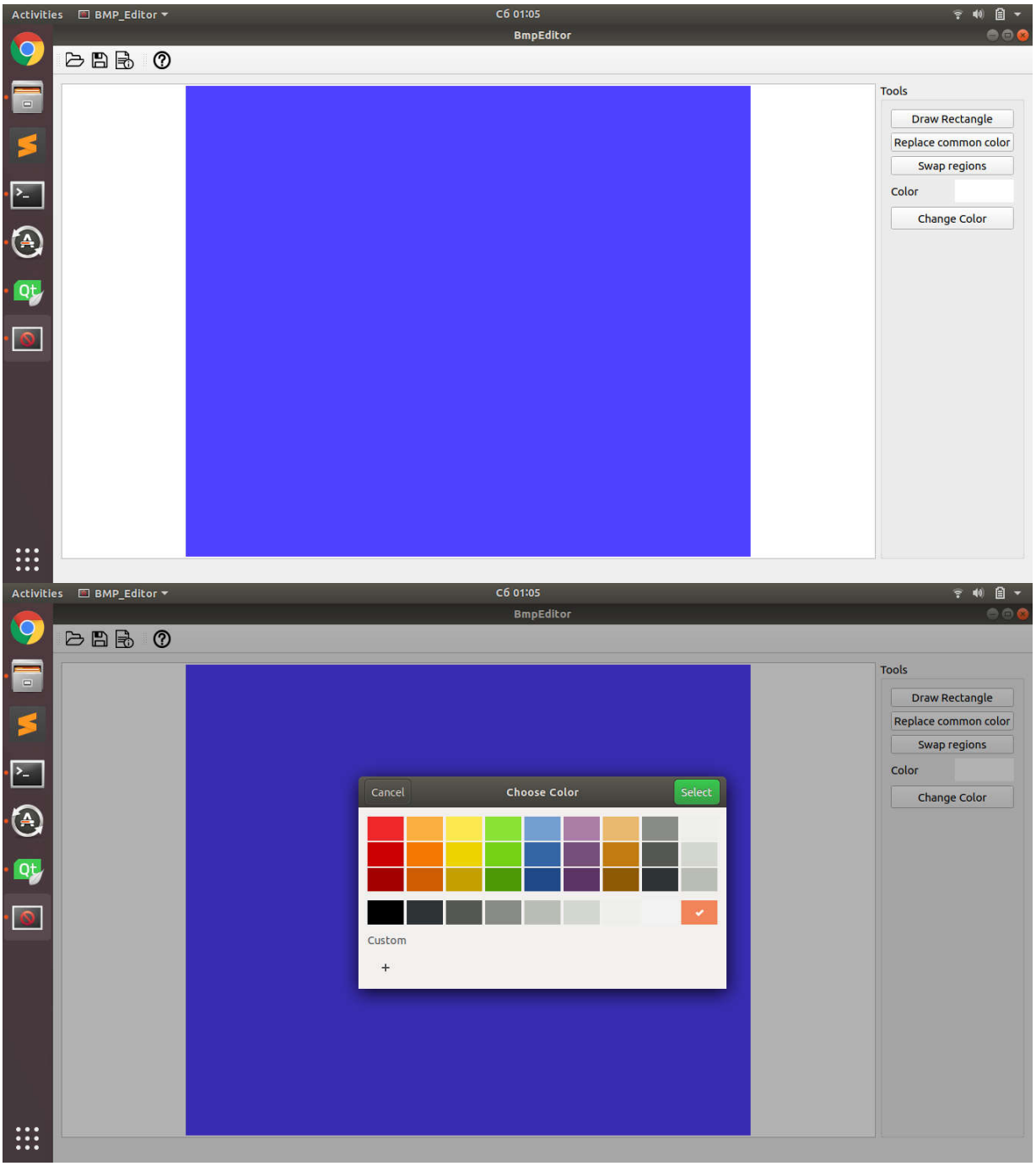
Сохранить данный из header и bitmap структурах в файл.

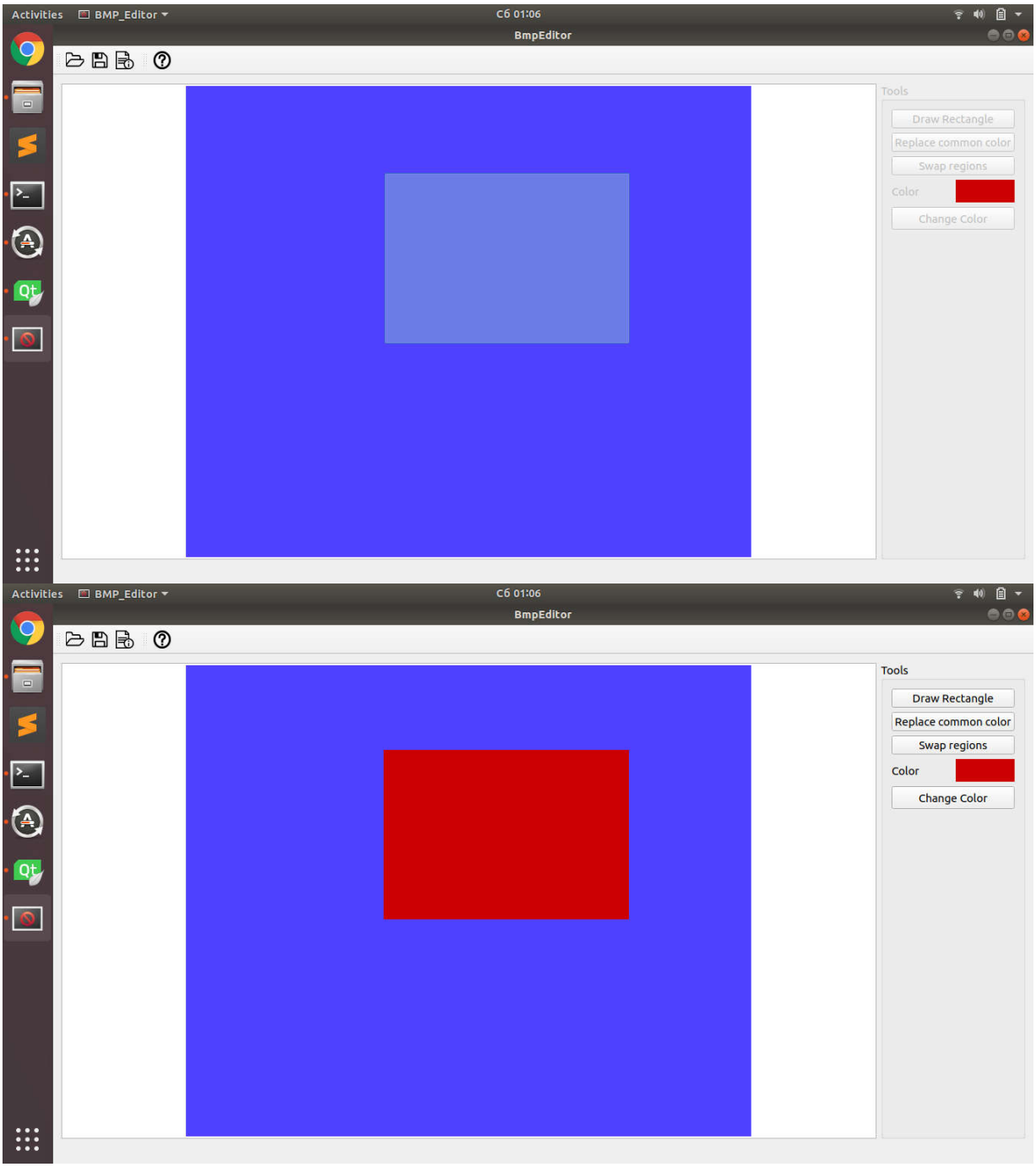
```
void BMPLib::writeBitmap(const char* fileName, Header *header, Bitmap bitmap){
            FILE *file;
        file = fopen(fileName,"rb+");
        if(file==NULL) {err_message=ERR_FILE_NOT_FOUND; printf("%s",fileName); return;}


        //WRITE
        int row_size = (header->bpp*(header->img_width)+31)/32*4;
        fseek(file,(header->file_header).bitmap_offset,SEEK_SET);
        unsigned char t;
        for(unsigned int i=0;i<(header->img_height);i++){
            for(unsigned int j=0;j<header->img_width;j++){
                t = (*(bitmap+i)+j)->b;
                (*(bitmap+i)+j)->b = (*(bitmap+i)+j)->r;
                (*(bitmap+i)+j)->r=t;
            }
            fwrite(*(bitmap+i),row_size,1,file);
            for(unsigned int j=0;j<header->img_width;j++){
                t = (*(bitmap+i)+j)->b;
                (*(bitmap+i)+j)->b = (*(bitmap+i)+j)->r;
                (*(bitmap+i)+j)->r=t;
            }
        }
              fflush(file);
              fclose(file);
       }
```

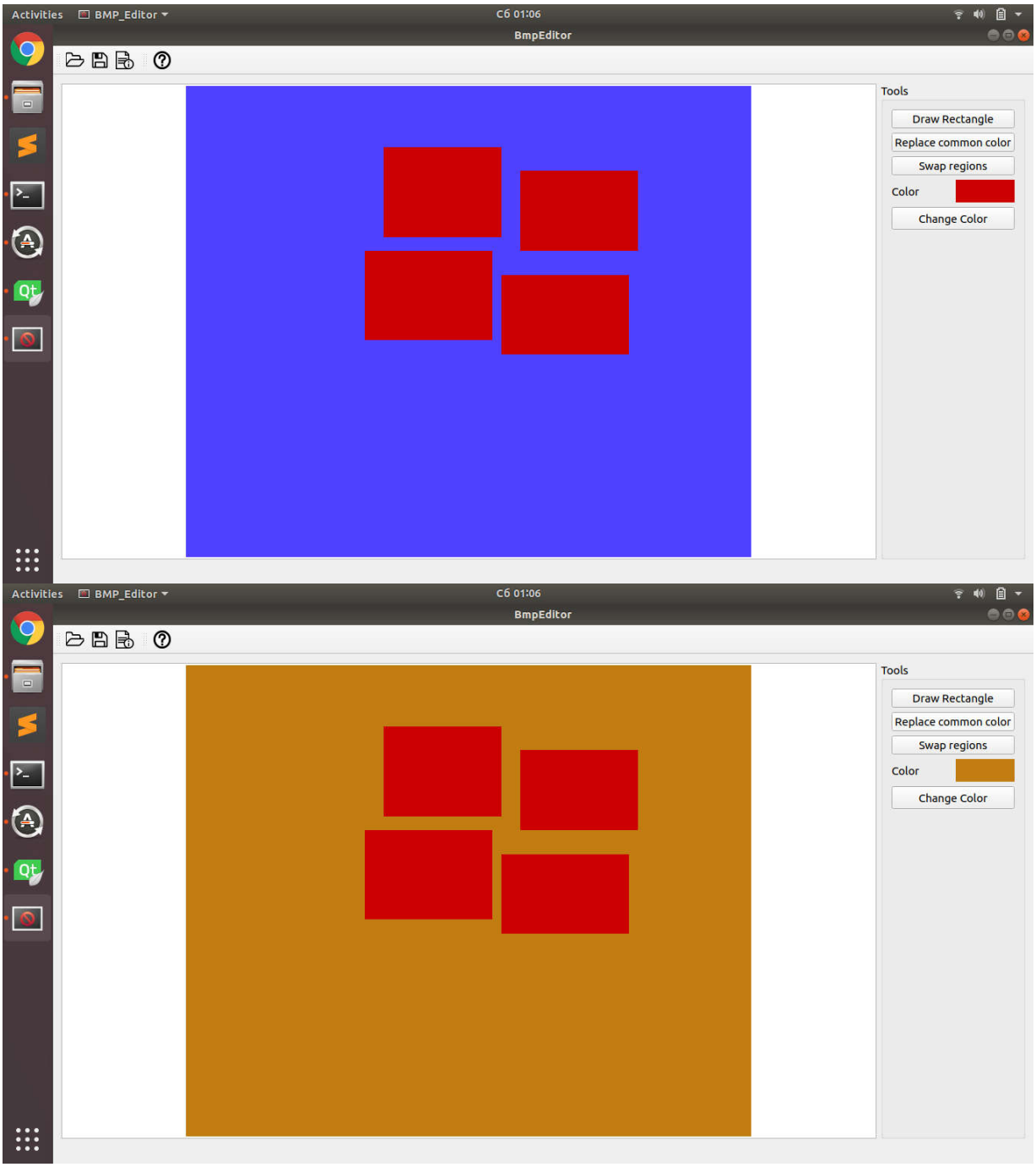
Демонстрационное использование

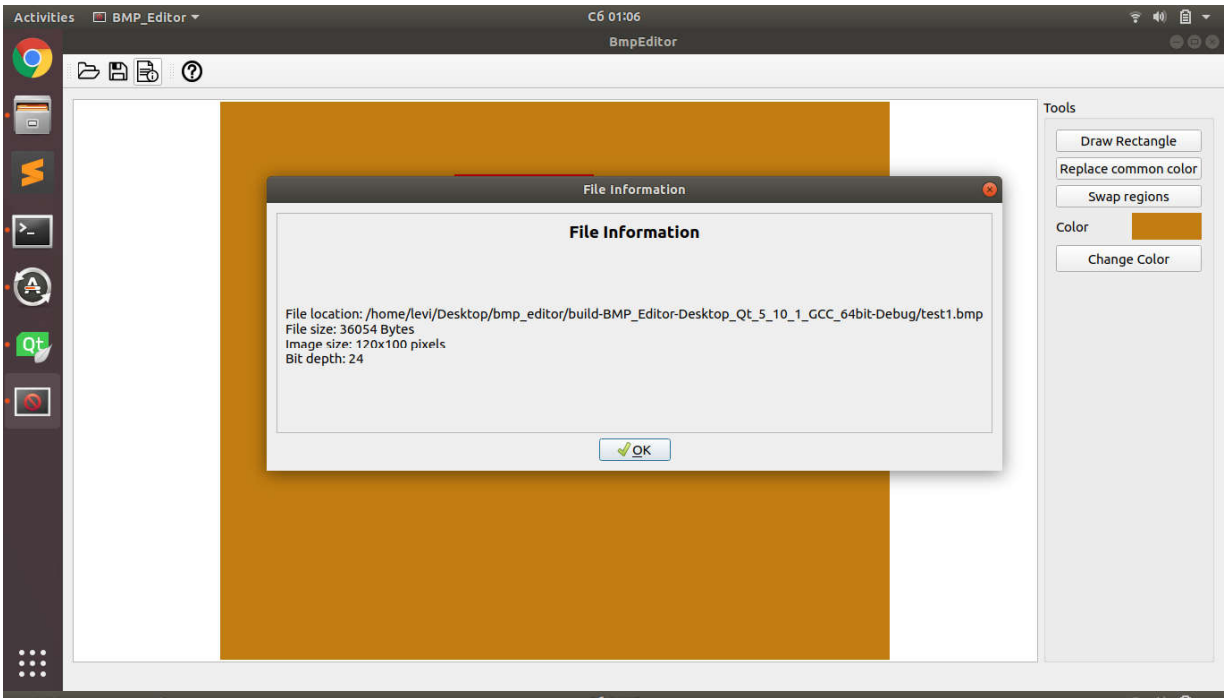


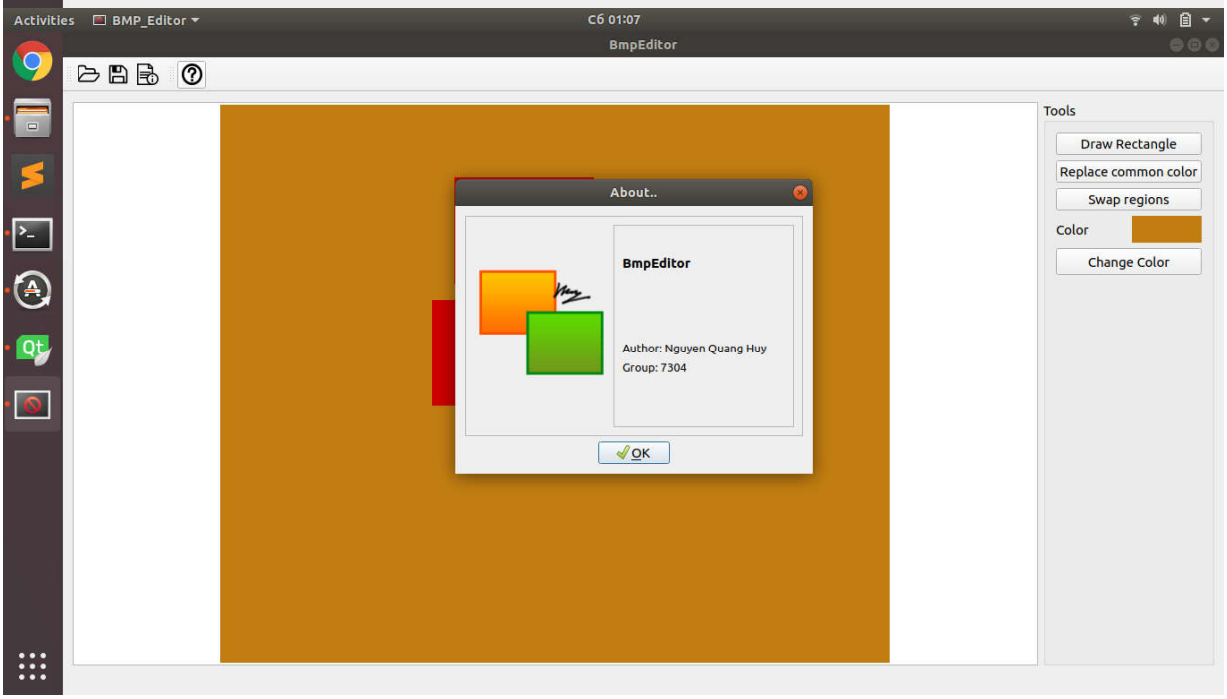


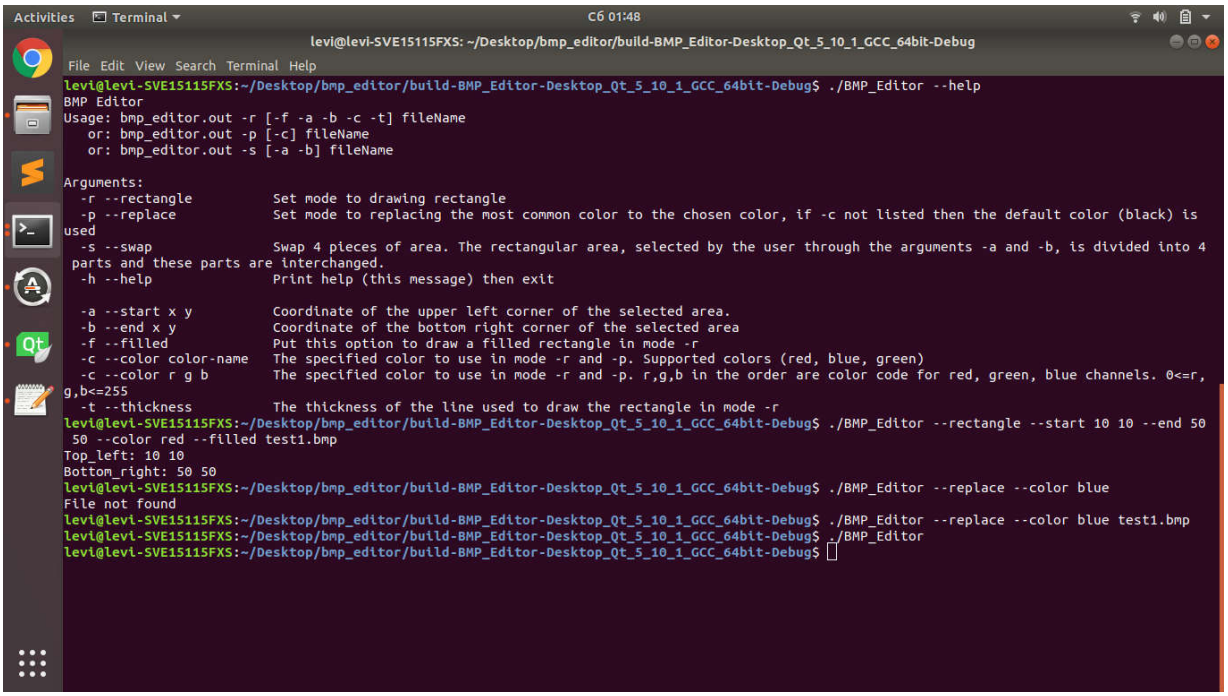

Activities
BMP_Editor
Сб 01:05
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color
Activities
BMP_Editor
Сб 01:05
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color
Cancel
Choose Color
Select
Custom


Activities
BMP_Editor
Сб 01:06
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color
Activities
BMP_Editor
Сб 01:06
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color


Activities
BMP_Editor
Сб 01:06
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color
Activities
BMP_Editor
Сб 01:06
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color


Activities
BMP_Editor
Сб 01:06
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color
File Information
File Information
File location: /home/levi/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug/test1.bmp
File size: 36054 Bytes
Image size: 120x100 pixels
Bit depth: 24
OK


Activities
BMP_Editor
Сб 01:07
BmpEditor
Tools
Draw Rectangle
Replace common color
Swap regions
Color
Change Color
About..
BmpEditor
Author: Nguyen Quang Huy
Group: 7304
OK


Activities
Terminal
Сб 01:48
levi@levi-SVE15115FXS: ~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug
File Edit View Search Terminal Help
levi@levi-SVE15115FXS:~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug$ ./BMP_Editor --help
BMP Editor
Usage: bmp_editor.out -r [-f -a -b -c -t] fileName
or: bmp_editor.out -p [-c] fileName
or: bmp_editor.out -s [-a -b] fileName
Arguments:
-r --rectangle Set mode to drawing rectangle
-p --replace Set mode to replacing the most common color to the chosen color, if -c not listed then the default color (black) is used
-s --swap Swap 4 pieces of area. The rectangular area, selected by the user through the arguments -a and -b, is divided into 4 parts and these parts are interchanged.
-h --help Print help (this message) then exit
-a --start x y Coordinate of the upper left corner of the selected area.
-b --end x y Coordinate of the bottom right corner of the selected area
-f --filled Put this option to draw a filled rectangle in mode -r
-c --color color-name The specified color to use in mode -r and -p. Supported colors (red, blue, green)
-c --color r g b The specified color to use in mode -r and -p. r,g,b in the order are color code for red, green, blue channels. 0<=r, g,b<=255
-t --thickness The thickness of the line used to draw the rectangle in mode -r
levi@levi-SVE15115FXS:~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug$ ./BMP_Editor --rectangle --start 10 10 --end 50 50 --color red --filled test1.bmp
Top_left: 10 10
Bottom_right: 50 50
levi@levi-SVE15115FXS:~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug$ ./BMP_Editor --replace --color blue
File not found
levi@levi-SVE15115FXS:~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug$ ./BMP_Editor --replace --color blue test1.bmp
levi@levi-SVE15115FXS:~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug$ ./BMP_Editor
levi@levi-SVE15115FXS:~/Desktop/bmp_editor/build-BMP_Editor-Desktop_Qt_5_10_1_GCC_64bit-Debug$

Заключение

В этой работе была исследована структура bmp-файла и то, как выполнять операцию в файле. Результатом является полностью работающая программа.