2021-2022-1 编程水平测试 1

【题目】根据要求对原始两个 24 位 bmp 图像进行处理，生成指定的文件同时显示到屏幕输出。原始图片如下：

tiger.bmp

Spring.bmp

1) 截取原图 tiger.bmp 左下角 148 * 200 的部分保存为 cut.bmp，同时输出到屏幕显示；

2) 将原图 tiger.bmp 进行 90 度翻转,生成 invert.bmp，同时输出到屏幕显示；

3) 给原图 tiger.bmp 加边框，边框宽度不小于 6，颜色可根据 bmp 配色图自行确定，要求清晰可见。加边框的图保存为 add.bmp，同时输出到屏幕显示；

4) 将原图 spring.bmp 叠加到 tiger.bmp 的左下角或者右上角。生成图 2in1.bmp，并输出到屏幕显示。

cut.bmp

invert.bmp

add.bmp

2in1.bmp(左下角)

2in1.bmp(右上角)

【要求】下载源文件，不要修改已完成的函数和代码段。认真阅读题目和源文件中的注释信息，根据要求实现需要完成的函数。

上传结果时请**提交 2 个文档**：

1） 将源文件复制粘贴到 word 文档中，并将运行结果图(四张)插入在程序后面，保存为“编程水平测试 1.docx”文件，上传到“源代码”目录下

2） 将结果图(四张)打包成压缩文档"编程水平测试 1.zip"上传到“运行结果”目录下。

【提示】若一幅 24 位 bmp 图像高为 h，宽为 w，那么可以考虑用 h 行 w 列的三个 char 型二维数组 b、g、r 分别存储每个像素点的 blue、green 和 red 分量的值，三个数组中同一个位置对应的就是该位置的像素点。那么处理图像就可以分解成：

1） 将 buf 指针所指像素流分解成 b、g、r 三个数组；

2） 根据要求对 b、g、r 三个数组进行处理；

3） 将三个数组按照 b、g、r 的顺序再写回 buf 或者新的缓存里。

需要注意的是，bmp 文件的存放是从下往上、从左向右进行的。

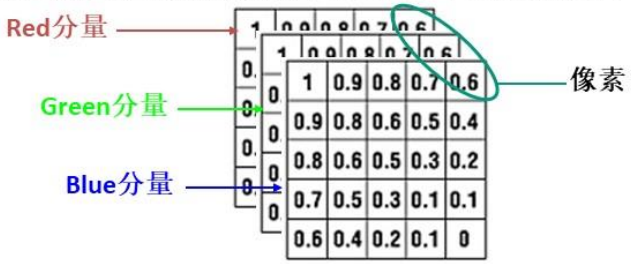
*24 位 bmp 图文件的具体格式详见附件一。

*常用 RGB 颜色表见附件二。

附件 bmp 文件

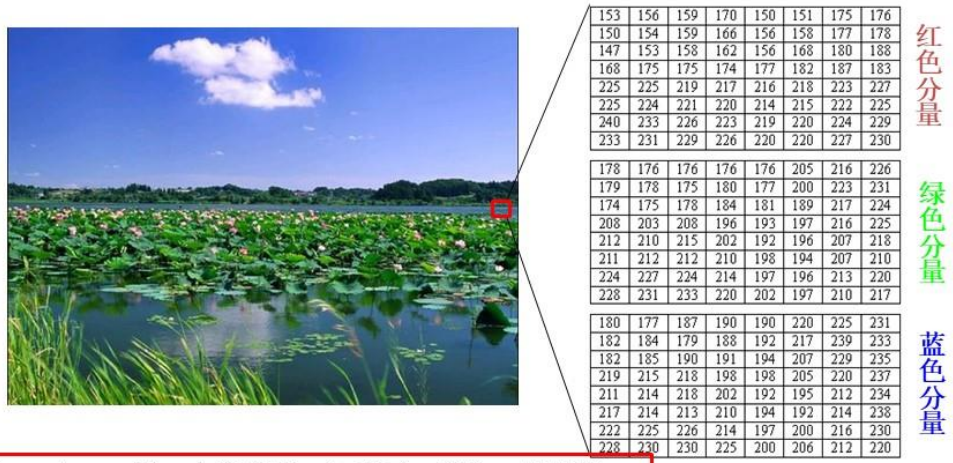
数字(取样)图像的组成

- 数字取样图像由M(列) × N(行)个取样点组成
- 取样点是组成数字取样图像的基本单位，称为“像素”
- 彩色图像的像素通常由3个彩色分量组成
- 灰度图像和黑白图像的像素只包含1个亮度分量



彩色图象的表示

■ 彩色图像的每个像素有三个分量，分别表示三个基色的亮度，假设3个分量分别用n,m,k个二进位表示，则可表示2^{n+m+k}种不同的颜色



取n=m=k=8，则一个像素用24bit表示，称为24位位图。

文件操作

– 打开文件（生成文件指针，指向对应内存）

– 读/写文件，（读入缓冲区进行操作）

– 关闭文件

打开和关闭文件

- 文件指针 = fopen(文件名，打开方式)
  - “r”以只读方式打开文本文件，“rb”只读打开二进制文件
  - “w”以只写方式打开文本文件，已存在的将被覆盖，“wb”写模式打开二进制文件
  - “a”以只写方式打开文本，指针指向文件尾，原文件内容保留，“ab”写模式打开二进制文件

打开/关闭文件

- **打开文件成功，返回文件指针，如果返回值为NULL，打开不成功。**

```
if ((fp = fopen(fileName, "rb")) == NULL)
{
    cout << "文件未找到！ ";
    exit(0);
}
```

- 关闭文件——fclose（文件指针）

三. 文件的读写

fread()

```
头文件：#include<stdio.h>
功能：是用于读取二进制数据
```

原型：

```
size_t fread(void*buffer,size_t size,size_t count,FILE*stream);
1.buffer:  是读取的数据存放的内存的指针，
         （可以是数组，也可以是新开辟的空间）
    ps:   是一个指向用于保存数据的内存位置的指针（为指向缓冲区
          保存或读取的数据或者是用于接收数据的内存地址）
2.size:   是每次读取的字节数
3.count:  是读取的次数
4.stream:  是要读取的文件的指针
      ps:  是数据读取的流（输入流）
```

fwrite()

```
功能：是用于写入二进制数据
头文件：#include<stdio.h>
```

原型：

```
size_t fwrite(void*buffer,size_ size,size_t count,FILE*stream)
1.buffer：是一个指向用于保存数据的内存位置的指针
        （是一个指针，对于fwrite来说，是要获取数据的地址）
2.size:    是每次读取的字节数
3.count:   是读取的次数
4.stream:  是数据写入的流（目标指针的文件）
```


读写文件

– nt fseek(FILE *stream, long offset, int origin);

– stream为文件指针

– offset为偏移量，整数表示正向偏移，负数表示负向偏移

– origin设定从文件的哪里开始偏移,可能取值为：

bmp文件格式

BMP文件由4部分组成：

1. 位图文件头(bitmap-file header)
2. 位图信息头(bitmap-informationheader)
3. 颜色表(color table)
4. 颜色点阵数据(bits data)

24位真彩色位图没有颜色表，所以只有1、2、4这三部分。

bmp文件格式

1、 1位图文件头（BITMAPFILEHEADER）

位图文件头分4部分，共**14**字节：

名称	占用空间	内容
bfType	2字节	标识，就是“BM”二字
bfSize	4字节	整个BMP文件的大小
bfReserved1/2	4字节	保留字，没用
bfOffBits	4字节	偏移数，即 位图文件头+位图信息头+调色板的大小

bmp文件格式

```
typedef struct tagBITMAPFILEHEADER {
WORD    bfType;
DWORD   bfSize;
WORD    bfReserved1;
WORD    bfReserved2;
DWORD   bfOffBits;
} BITMAPFILEHEADER;
```


1、2位图信息头（BITMAPINFOHEADER ）

位图信息头共**40**字节：

名称	占用空间	内容
biSize	4字节	位图信息头的大小，为40
biWidth	4字节	位图的宽度，单位是像素
biHeight	4字节	位图的高度，单位是像素
biPlanes	2字节	固定值1
biBitCount	2字节	每个像素的位数 1-黑白图，4-16色，8-256色，24-真彩色
biCompression	4字节	压缩方式，BI_RGB(0)为不压缩
biSizeImage	4字节	位图全部像素占用的字节数，BI_RGB时可设为0
biXPelsPerMeter	4字节	水平分辨率(像素/米)
biYPelsPerMeter	4字节	垂直分辨率(像素/米)
biClrUsed	4字节	位图使用的颜色数 如果为0，则颜色数为2的biBitCount次方
biClrImportant	4字节	重要的颜色数，0代表所有颜色都重要

bmp文件格式

```
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;
```


bmp文件格式

1、3颜色表

24位真彩色位图没有颜色表。为了简化，只讨论24位真彩色位图。

1、4颜色点阵数据

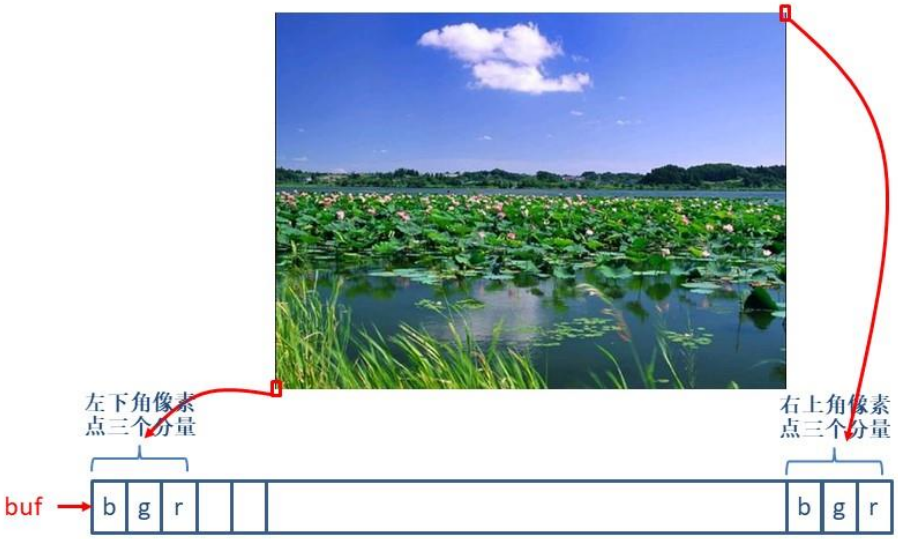
位图全部的像素，是按照自下向上，自左向右的顺序排列的。

RGB数据也是倒着念的，原始数据是按B、G、R的顺序排列的。

bmp文件格式

位图全部的像素，是按照自下向上，自左向右的顺序排列的。

RGB数据也是倒着念的，原始数据是按B、G、R的顺序排列的。



附件二.常用 RGB 颜色表（一）

	R	G	B	值
黑色	0	0	0	#000000
象牙黑	41	36	33	#292421
灰色	192	192	192	#C0C0C0
冷灰	128	138	135	#808A87
石板灰	112	128	105	#708069
暖灰色	128	128	105	#808069
白色	225	225	225	#FFFFFF
古董白	250	235	215	#FAEBD7
天蓝色	240	255	255	#F0FFFF
白烟	245	245	245	#F5F5F5
白杏仁	255	235	205	#FFFFCD
cornsilk	255	248	220	#FFF8DC
蛋壳色	252	230	201	#FCE6C9

	R	G	B	值
黄色	255	255	0	#FFFF00
香蕉色	227	207	87	#E3CF57
镉黄	255	153	18	#FF9912
dougello	235	142	85	#EB8E55
forum gold	255	227	132	#FFE384
金黄色	255	215	0	#FFD700
黄花色	218	165	105	#DAA569
瓜色	227	168	105	#E3A869
橙色	255	97	0	#FF6100
镉橙	255	97	3	#FF6103
胡萝卜色	237	145	33	#ED9121
桔黄	255	128	0	#FF8000
淡黄色	245	222	179	#F5DEB3

	R	G	B	值
浅灰蓝色	176	224	230	#B0E0E6
品蓝	65	105	225	#4169E1
石板蓝	106	90	205	#6A5ACD
天蓝	135	206	235	#87CEEB
青色	0	255	255	#00FFFF
绿土	56	94	15	#385E0F
靛青	8	46	84	#082E54
碧绿色	127	255	212	#7FFFD4
青绿色	64	224	208	#40E0D0
绿色	0	255	0	#00FF00
黄绿色	127	255	0	#7FFF00
钴绿色	61	145	64	#3D9140
翠绿色	0	201	87	#00C957