

数字图像处理作业-第3章

计创18 181002222 连月菡

-目录-

数字图像处理作业-第3章

- 一、作业描述
- 二、代码描述
- 三、运行结果
- 报错提示:
- 0.结束
- 1.读取BMP图像
- 2.保存BMP图像
- 3.显示图像数据
- 4.计算直方图
- 5.图像增强/减暗
- 6.反色
- 7.灰度图
- 8.显示调色板(8bit)
- 四、作业代码

一、作业描述

编写一个VC++的控制台程序。要求如下：

一、打开一个BMP图像文件（图像大小不要太大）。

二、显示图像文件的数据。

对于读入的图像文件，显示位图数据(选择一部分区域显示即可)。用256色或256级灰度图像进行验证。有两种显示方式：从上到下，从下到上。

三、计算图像的直方图(可以只显示灰度级100~200的直方图，为0的不显示；大小的处理，先归一化，再乘以50)

对于256级灰度级的图像进行计算。存入一个256元素的数组中，并显示其数据。用图形的方式显示（数据为0的不显示，如果值太大，按比例缩小一下）。如下。

四、图像的增亮或减暗

输入一个数字，如果是正数对图像进行增亮，如果是负数对图像进行减暗。

用其他软件进行查看，源图像和被增亮或减暗的图像进行对比，是不是达到了预想的效果。

五、main() 函数的要求

有选择。

main()函数将指定BMP文件读入内存，将图像信息打印输出，最后又原样存入指定文件中。并进行计算需要的数据。

界面：

- 0-----结束
- 1-----读图像（8/24位）
- 2-----写图像（8/24位）
- 3-----显示图像数据

- 4-----计算直方图（100-200之间的灰度，1-10行，1-0列,1010的大小）(8位)
- 5-----对图像进行增亮或减暗（如加上50或减去50等）(8位)
- 6-----图像反色
- 7-----变成灰度图（8/24位）
- 8-----显示调色板(8位)
- 9-----24位彩色图像转换为8位灰度图像
- 10-----24位彩色图像转换为8位彩色图像

input choice:

要有操作提示。

功能 7-10 选做。

对于BMP文件的读写函数仅针对灰度图像（biBitCount=8）和彩色图像（biBitCount=24）两种格式，对于其他如biBitCount=1的图像类型，读者可以根据需要，自己对程序作简单的修改即可实现。

二、代码描述

本代码实现了0-8的功能。

分为3个部分：

- 第一部分 bitmap文件结构的定义
- 第二部分 9个功能函数的定义
- 第三部分 主函数进行功能选择

在同一个文件夹中,附有对应的.cpp文件。

运行时需注意:main()函数和对应功能函数里 图像文件路径需要按照用户电脑存储的路径进行修改,否则会报错。

三、运行结果

报错提示:

```
BMP图像处理作业菜单
0. 结束啦
1. 读取BMP图像(8/24bit)
2. 保存BMP图像(8/24bit)
3. 显示图像数据
4. 计算直方图
5. 图像增强/减暗
6. 反色
7. 灰度图(同位转换, 不改变位数)
8. 显示调色板(8bit)

输入选项: 1

并无此文件, 请检查main函数中定义的路径是否正确!
```

表示路径错误,需要查看main()函数中设置的inputFile路径是否在用户的电脑上。

0.结束

```
输入选项：0
感谢你的使用！
```

1.读取BMP图像

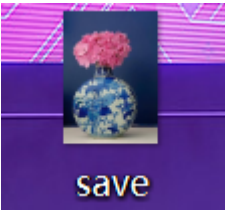
```
输入选项：1
宽度=265          长度=374
读取成功！
```

输出宽度和长度,表示读取成功。

2.保存BMP图像

```
输入选项：2
保存成功！
```

查看对应路径下是否有save.bmp文件。如果有,则运行正常。



3.显示图像数据

```
输入选项：3
109      61      37      52      52      52      52      52      52      52
52       52      52      52      52      0       52      52      52      52
52       52      52      52      56      0       0       0       52      52
52       52      52      52      52      52      28      3       0       0
52       52      52      52      52      52      52      52      1       0
0        0       52      52      52      52      52      52      52      52
3        0       0       0       52      52      52      52      52      52
52       52      27      3       0       0       52      52      52      52
52       52      52      52      118     1       0       0       52      52
52       52      52      52      52      52      9       1       0       0
显示完毕！
```

输出 10*10数据内容。

4.计算直方图

```

输入选项: 4
像素数据为:
归一化后的直方图数据为:
0      0      0      0      0      47      106      29      74      38      13      47      109      60      44      92
28      32      63      0      0      0      0      0      0      0      0      0      66      66      65      65
63      63      64      64      64      64      63      62      63      62      62      61      64      65      65      65
65      65      65      65      62      62      62      62      62      62      62      62      63      62      61      61
60      60      60      61      59      59      59      59      59      59      59      59      62      62      62      61
60      60      59      59      56      57      57      58      58      57      57      56      54      53      53      53
52      52      51      51      50      50      50      51      51      51      51      52      51      51      51      51
51      51      51      51      53      53      54      54      53      52      52      51      52      50      49      49
51      53      53      52      50      50      49      49      49      50      51      52      52      51      51      50
51      52      53      53      51      51      51      51      51      52      53      53      53      53      53      53
53      53      53      53      54      54      54      54      54      54      54      54      54      54      54      54
54      54      54      54      56      56      56      56      56      56      56      56      56      56      56      56
57      57      58      58      57      57      58      58      59      60      60      60      59      59      59      59
60      60      61      61      63      62      62      64      66      64      60      55      62      62      63      63
64      65      65      65      65      65      65      65      65      65      65      65      64      64      64      64
64      64      64      64      65      65      65      65      65      65      65      65      64      64      64      64

```

```

13 :*
28 :*
29 :*
32 :*
38 :*
44 :*
47 :**
49 :*****
50 :*****
51 :*****
52 :*****
53 :*****
54 :*****
55 :*
56 :*****
57 :*****
58 :*****
59 :*****
60 :*****
61 :*****
62 :*****
63 :*****
64 :*****
65 :*****
66 :***
74 :*
92 :*
106 :*
109 :*

```

5.图像增强/减暗

```

输入选项: 5
1. 增强
2. 减暗
输入你的选择:
1
保存成功!

```

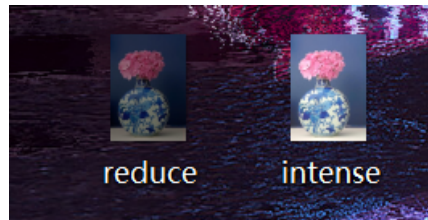
查看对应路径下是否有intense.bmp文件。如果有,则运行正常。

```

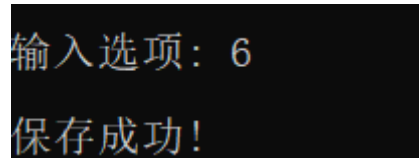
输入选项: 5
1. 增强
2. 减暗
输入你的选择:
2
保存成功!

```

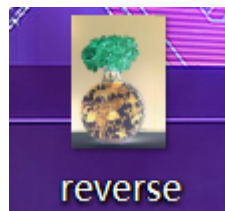
查看对应路径下是否有reduce.bmp文件。如果有,则运行正常。



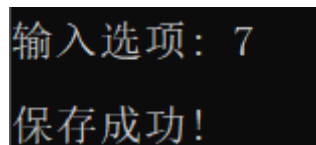
6.反色



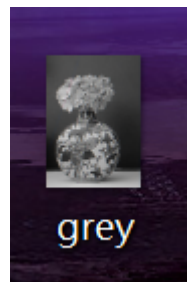
查看对应路径下是否有reverse.bmp文件。如果有,则运行正常。



7.灰度图



查看对应路径下是否有grey.bmp文件。如果有,则运行正常。



8.显示调色板(8bit)

如果该图是24bit BMP文件,则显示:



如果该图是8bit BMP文件,则显示调色板内容:

输入选项：8

0	0	0	0	0	0	0	128	0	0	128	0	0
0	128	128	0	0	128	0	0	0	128	0	128	0
128	128	0	0	192	192	192	0	192	224	192	0	0
240	202	166	0	0	32	64	0	0	32	96	0	0
0	32	128	0	0	32	160	0	0	32	192	0	0
0	32	224	0	0	64	0	0	0	64	32	0	0
0	64	64	0	0	64	96	0	0	64	128	0	0
0	64	160	0	0	64	192	0	0	64	224	0	0
0	96	0	0	0	96	32	0	0	96	64	0	0
0	96	96	0	0	96	128	0	0	96	160	0	0
0	96	192	0	0	96	224	0	0	128	0	0	0
0	128	32	0	0	128	64	0	0	128	96	0	0
0	128	128	0	0	128	160	0	0	128	192	0	0
0	128	224	0	0	160	0	0	0	160	32	0	0
0	160	64	0	0	160	96	0	0	160	128	0	0
0	160	160	0	0	160	192	0	0	160	224	0	0
0	192	0	0	0	192	32	0	0	192	64	0	0
0	192	96	0	0	192	128	0	0	192	160	0	0
0	192	192	0	0	192	224	0	0	224	0	0	0
0	224	32	0	0	224	64	0	0	224	96	0	0
0	224	128	0	0	224	160	0	0	224	192	0	0
0	224	224	0	64	0	0	0	64	0	32	0	0
64	0	64	0	64	0	96	0	64	0	128	0	0
64	0	160	0	64	0	192	0	64	0	224	0	0
64	32	0	0	64	32	32	0	64	32	64	0	0
64	32	96	0	64	32	128	0	64	32	160	0	0
64	32	192	0	64	32	224	0	64	64	0	0	0
64	64	32	0	64	64	64	0	64	64	96	0	0
64	64	128	0	64	64	160	0	64	64	192	0	0
64	64	224	0	64	96	0	0	64	96	32	0	0
64	96	64	0	64	96	96	0	64	96	128	0	0
64	96	160	0	64	96	192	0	64	96	224	0	0
64	128	0	0	64	128	32	0	64	128	64	0	0
64	128	96	0	64	128	128	0	64	128	160	0	0
64	128	192	0	64	128	224	0	64	160	0	0	0
64	160	32	0	64	160	64	0	64	160	96	0	0
64	160	128	0	64	160	160	0	64	160	192	0	0
64	160	224	0	64	192	0	0	64	192	32	0	0
64	192	64	0	64	192	96	0	64	192	128	0	0
64	192	160	0	64	192	192	0	64	192	224	0	0
64	224	0	0	64	224	32	0	64	224	64	0	0
64	224	96	0	64	224	128	0	64	224	160	0	0
64	224	192	0	64	224	224	0	128	0	0	0	0
128	0	32	0	128	0	64	0	128	0	96	0	0
128	0	128	0	128	0	160	0	128	0	192	0	0
128	0	224	0	128	32	0	0	128	32	32	0	0
128	32	64	0	128	32	96	0	128	32	128	0	0
128	32	160	0	128	32	192	0	128	32	224	0	0
128	64	0	0	128	64	32	0	128	64	64	0	0
128	64	96	0	128	64	128	0	128	64	160	0	0
128	64	192	0	128	64	224	0	128	96	0	0	0
128	96	32	0	128	96	64	0	128	96	96	0	0
128	96	128	0	128	96	160	0	128	96	192	0	0
128	96	224	0	128	128	0	0	128	128	32	0	0
128	128	64	0	128	128	96	0	128	128	128	0	0
128	128	160	0	128	128	192	0	128	128	224	0	0
128	160	0	0	128	160	32	0	128	160	64	0	0
128	160	96	0	128	160	128	0	128	160	160	0	0
128	160	192	0	128	160	224	0	128	192	0	0	0
128	192	32	0	128	192	64	0	128	192	96	0	0
128	192	128	0	128	192	160	0	128	192	192	0	0
128	192	224	0	128	224	0	0	128	224	32	0	0
128	224	64	0	128	224	96	0	128	224	128	0	0
128	224	160	0	128	224	192	0	128	224	224	0	0

显示完毕!

四、作业代码

```
1
2 #include <stdio.h>
3 #include<cmath>
4 #include <cstdint>
5 #include <iostream>
6 #include <stdio.h>
7 #include <stdlib.h>
8 #include <stdint.h>
9 #include <vector>
10 #include <fstream>
11 #include <cmath>
12 #include <map>
13 /*****
14 *=====*
15 *==BMP文件结构定义开始==*
16 *=====*
17 *****/
18 typedef unsigned char BYTE;
19 typedef unsigned short WORD;
20 typedef unsigned int DWORD;
21 typedef int LONG;
22
23 #pragma pack(push, 1)
24 typedef struct tagBITMAPFILEHEADER {
25     WORD bfType;
26     DWORD bfSize;
27     WORD bfReserved1;
28     WORD bfReserved2;
29     DWORD bfOffBits;
30 } BITMAPFILEHEADER;
31
32 typedef struct tagBITMAPINFOHEADER {
33     DWORD biSize;
34     LONG biWidth;
35     LONG biHeight;
36     WORD biPlanes;
37     WORD biBitCount;
38     DWORD biCompression;
39     DWORD biSizeImage;
40     LONG biXPelsPerMeter;
41     LONG biYPelsPerMeter;
42     DWORD biClrUsed;
43     DWORD biClrImportant;
44 } BITMAPINFOHEADER;
45 #pragma pack(pop)
46
47 typedef struct tagRGBTRIPLE {
48     BYTE rgbBlue;
49     BYTE rgbGreen;
50     BYTE rgbRed;
51 } RGBTRIPLE;
52 static FILE* inputFile;
53 static FILE* outputFile;
54 /*****
```

```

55  *=====*
56  *==BMP文件结构定义结束==*
57  *=====*
58  *****/
59
60  /*****
61  *=====*
62  *==各个功能函数定义开始==*
63  *=====*
64  *****/
65
66  int read(const char* input) { // 1. 读取图像
67      inputFile = fopen(input, "rb");
68      if (inputFile == NULL) {
69          printf("并无此文件, 请检查main函数中定义的路径是否正确!\n"); // 文件读取失败,
重新修改文件路径
70      }
71      char header[122];
72      fread(&header, sizeof(char), 122, inputFile); // 获得图像头部信息
73      int width = *(int*)&header[18]; // 获得图像宽度
74      int height = abs(*(int*)&header[22]); // 获得图像高度
75      printf("宽度=%d\t长度=%d\n", width, height); // 功能1实现
76      return 1; // 成功, 返回1
77  }
78
79  int save(const char* input) { // 2. 保存图像
80      inputFile = fopen(input, "rb");
81      if (inputFile == NULL) {
82          printf("并无此文件, 请检查main函数中定义的路径是否正确!\n"); // 文件读取失败,
重新修改文件路径
83      }
84
85      outputFile = fopen("C:/Users/yuehan lian/Desktop/save.bmp", "wb");
86      if (outputFile == NULL) {
87          printf("输出图像文件创建失败!\n"); // 文件创建失败
88          return 0; // 函数执行失败, 返回0
89      }
90      char header[122];
91      fread(&header, sizeof(char), 122, inputFile); // 获得输入图像头部信息
92      fwrite(&header, sizeof(char), 122, outputFile); // 写进输出图像头部信息
93
94      int width = *(int*)&header[18]; // 获得图像宽度
95      int height = abs(*(int*)&header[22]); // 获得图像高度
96      int stride = ((4 * ((8 * 3) * width) + 31) / 32); // 计算行的数量
97      int paddingCheck = (stride % 4);
98      int padding;
99      if (paddingCheck != 0) { // 如果行数不能被4整除
100          int padCalc = 0;
101          for (int i = 1; padCalc < stride; i++) {
102              padCalc = i * 4;
103          }
104          padding = padCalc - stride;
105      }
106      else
107          padding = 0;
108
109      char pad;
110      char pixel[3];

```



```

111     for (int j = 0; j < height; j++) {
112         for (int k = 0; k < width; k++) {
113             fread(pixel, sizeof(char), 3, inputFile); //读取RGB信息,并写入
pixel中
114             fwrite(&pixel, (sizeof(char)), 3, outputFile); //pixel中的信息写
入输出图像中
115         }
116         for (int l = 0; l < padding; l++) {
117             fread(&pad, 1, 1, inputFile); //读取末尾填充的图像信息
118             fwrite(&pad, 1, 1, outputFile); //向输出图像写入末尾填充的图像信息
119         }
120     }
121     int elms_read; //确保没有空像素被读取
122     while (1) {
123         elms_read = fread(&pad, 1, 1, inputFile);
124         if (elms_read == 0) {
125             break;
126         }
127         else {
128             fwrite(&pad, 1, 1, outputFile);
129         }
130     }
131     fclose(inputFile);
132     fclose(outputFile); //关闭文件~
133     return 1; //执行成功返回1
134 }
135
136 int show(const char* input) { //3. 显示图像信息
137     inputFile = fopen(input, "rb");
138     if (inputFile == NULL) {
139         printf("并无此文件,请检查main函数中定义的路径是否正确!\n"); //文件读取失败,
重新修改文件路径
140     }
141
142     char header[54];
143     fread(&header, sizeof(char), 54, inputFile); //获得输入图像头部信息
144
145     int width = *(int*)&header[18]; //获得图像宽度
146     int height = abs(*(int*)&header[22]); //获得图像高度
147     int stride = (4 * ((8 * 3) * width) / 32); //计算行的数量
148     int paddingCheck = (stride % 4);
149     int padding;
150     if (paddingCheck != 0) { //如果行数不能被4整除
151         int padCalc = 0;
152         for (int i = 1; padCalc < stride; i++) {
153             padCalc = i * 4;
154         }
155         padding = padCalc - stride;
156     }
157     else
158         padding = 0;
159
160     char pad;
161     char pixel[3];
162     for (int j = 0; j < height; j++) {
163         for (int k = 0; k < width; k++) {
164             fread(pixel, sizeof(char), 3, inputFile); //读取RGB信息,并写入
pixel中

```

```

165     }
166     for (int l = 0; l < padding; l++) { //pixel中的信息写入输出图像中
167         fread(&pad, 1, 1, inputFile); //读取填充的图像信息
168     }
169 }
170 int elms_read; //确保没有空像素被读取
171 while (1) {
172     elms_read = fread(&pad, 1, 1, inputFile);
173     if (elms_read == 0) {
174         break;
175     }
176 }
177
178 for (int i = 0; i < 100; ++i)
179 {
180     printf("%d\t", abs((int)pixel[i]));
181     if ((i + 1) % 10 == 0)
182         printf("\n");
183 }
184 fclose(inputFile);
185 return 1;
186 }
187
188 int reverse(const char* input) { // 6. 图像反色
189     inputFile = fopen(input, "rb");
190     if (inputFile == NULL) {
191         printf("并无此文件, 请检查main函数中定义的路径是否正确!\n"); // 文件读取失败,
重新修改文件路径
192         return 0; // 函数执行失败, 返回0
193     }
194     outputFile = fopen("C:/Users/yuehan lian/Desktop/reverse.bmp", "wb");
195     if (outputFile == NULL) {
196         printf("输出图像文件创建失败!\n"); // 文件创建失败
197         return 0; // 函数执行失败, 返回0
198     }
199     char header[54];
200     fread(&header, sizeof(char), 54, inputFile); // 获得输入图像头部信息
201     fwrite(&header, sizeof(char), 54, outputFile); // 写进输出图像头部信息
202
203     int width = *(int*)&header[18]; // 获得图像宽度
204     int height = abs(*(int*)&header[22]); // 获得图像高度
205     int stride = (4 * ((8 * 3) * width) / 32); // 计算行的数量
206     int paddingCheck = (stride % 4);
207     int padding;
208     if (paddingCheck != 0) { // 计算行的数量
209         int padCalc = 0;
210         for (int i = 1; padCalc < stride; i++) {
211             padCalc = i * 4;
212         }
213         padding = padCalc - stride;
214     }
215     else
216         padding = 0;
217
218     char pad;
219     char pixel[3];
220     for (int j = 0; j < height; j++) {
221         for (int k = 0; k < width; k++) {

```

```

222         fread(pixel, sizeof(char), 3, inputFile); //读取RGB信息,并写入
pixel中
223         //pixel[0] =~pixel[0]; //进行取反运算,得到反色后的像素值
224         //pixel[1] = ~pixel[1];
225         //pixel[2] = ~pixel[2];
226         pixel[0] = 255 - pixel[0];
227         pixel[1] = 255 - pixel[1];
228         pixel[2] = 255 - pixel[2];
229         fwrite(&pixel, (sizeof(char)), 3, outputFile); //向输出图像写入末
尾填充的图像信息
230     }
231     for (int l = 0; l < padding; l++) {
232         fread(&pad, 1, 1, inputFile); //读取末尾填充的图像信息
233         fwrite(&pad, 1, 1, outputFile); //向输出图像写入末尾填充的图像信息
234     }
235 }
236 int elms_read; //确保没有空像素被读取
237 while (1) {
238     elms_read = fread(&pad, 1, 1, inputFile);
239     if (elms_read == 0) {
240         break;
241     }
242     else {
243         fwrite(&pad, 1, 1, outputFile);
244     }
245 }
246 fclose(inputFile);
247 fclose(outputFile);
248 return 1;
249 }
250
251 int greyscale(const char* input) { //7. 获得同一位深的灰度图像
252     inputFile = fopen(input, "rb");
253     if (inputFile == NULL) {
254         printf("并无此文件,请检查main函数中定义的路径是否正确!\n"); //文件读取失败,
重新修改文件路径
255         return 0; //函数执行失败,返回0
256     }
257
258     outputFile = fopen("C:/Users/yuehan lian/Desktop/grey.bmp", "wb");
259     if (outputFile == NULL) {
260         printf("输出图像文件创建失败!\n"); //文件创建失败
261         return 0; //函数执行失败,返回0
262     }
263
264     char header[54];
265     fread(&header, sizeof(char), 54, inputFile); //获得输入图像头部信息
266     fwrite(&header, sizeof(char), 54, outputFile); //写进输出图像头部信息
267
268     int width = *(int*)&header[18]; //获得图像宽度
269     int height = abs(*(int*)&header[22]); //获得图像高度
270     int stride = (4 * ((8 * 3) * width + 31) / 32); //计算行的数量
271     int paddingCheck = (stride % 4);
272     int padding;
273     if (paddingCheck != 0) { //如果行数不能被4整除
274         int padCalc = 0;
275         for (int i = 1; padCalc < stride; i++) {
276             padCalc = i * 4;

```

```

277     }
278     padding = padCalc - stride;
279 }
280 else
281     padding = 0;
282
283     char pad;
284     unsigned char pixel[3];
285     unsigned char pixelEdit[3];
286     for (int j = 0; j < height; j++) {
287         for (int k = 0; k < width; k++) {
288             fread(pixel, sizeof(unsigned char), 3, inputFile); // 读取RGB信
信息, 并写入pixel中
289             unsigned char avg = (pixel[0] + pixel[1] + pixel[2]) /
(unsigned char)3;
290             pixelEdit[0] = pixelEdit[1] = pixelEdit[2] = avg; //如果要变灰度
图, RGB三个分量要相等
291
292             fwrite(&pixelEdit, (sizeof(unsigned char)), 3,
outputFile); //pixel中的信息写入输出图像中
293         }
294         for (int l = 0; l < padding; l++) { // 读取末尾填充的图像信息
295             fread(&pad, 1, 1, inputFile);
296             fwrite(&pad, 1, 1, outputFile); //向输出图像写入末尾填充的图像信息
297         }
298     }
299     int elms_read; //确保没有空像素被读取
300     while (1) {
301         elms_read = fread(&pad, 1, 1, inputFile);
302         if (elms_read == 0) {
303             break;
304         }
305         else {
306             fwrite(&pad, 1, 1, outputFile);
307         }
308     }
309     fclose(inputFile);
310     fclose(outputFile); //关闭文件~
311     return 1; //执行成功返回1
312 }
313
314 int intense(const char* input) { //5. 图像增强或减暗
315     inputFile = fopen(input, "rb");
316     if (inputFile == NULL) {
317         printf("并无此文件, 请检查main函数中定义的路径是否正确!\n"); //文件读取失败,
重新修改文件路径
318     }
319
320     char header[122];
321     fread(header, sizeof(char), 122, inputFile); //获得输入图像头部信息
322
323     int width = *(int*)&header[18]; //获得图像宽度
324     int height = abs(*(int*)&header[22]); //获得图像高度
325     int stride = (4 * ((8 * 3) * width) / 32); //计算行的数量
326     int paddingcheck = (stride % 4);
327     int padding;
328     if (paddingcheck != 0) { //如果行数不能被4整除
329         int padcalc = 0;

```

```

330         for (int i = 1; padcalc < stride; i++) {
331             padcalc = i * 4;
332         }
333         padding = padcalc - stride;
334     }
335     else
336         padding = 0;
337     int op;
338     printf("1.增强\n");
339     printf("2.减暗\n");
340     printf("输入你的选择: \n");
341     scanf("%d", &op);
342     if(op==1)
343         outputFile = fopen("C:/Users/yuehan lian/Desktop/intense.bmp",
344 "wb");//需要修改路径
345         if (op == 2)
346             outputFile = fopen("C:/Users/yuehan lian/Desktop/reduce.bmp",
347 "wb");//需要修改路径
348         if (outputFile == NULL) {
349             printf("the file cannot be found\n");
350         }
351         fwrite(header, sizeof(char), 122, outputFile);
352         char pad;
353         unsigned char pixel[3];
354         unsigned char pixeledit[3];
355         for (int j = 0; j < height; j++) {
356             for (int k = 0; k < width; k++) {
357                 fread(pixel, sizeof(unsigned char), 3, inputFile); //读取RGB信
358                 息,并写入pixel中
359                 if (op == 1) { //图像增强    RGB分量 +50,如果超出255,则等于255
360                     pixel[0] +50 >=255 ? pixel[0]=255 : pixel[0]= pixel[0] +
361                     50;
362                     pixel[1] +50>= 255 ? pixel[1] = 255 : pixel[1] = pixel[1]
363                     + 50;
364                     pixel[2] +50 >=255 ? pixel[2] = 255 : pixel[2] = pixel[2]
365                     + 50;
366                 }
367                 if (op == 2) { //图像减暗    RGB分量 *0.6
368                     pixel[0] *= 0.6;
369                     pixel[1] *= 0.6;
370                     pixel[2] *= 0.6;
371                 }
372                 fwrite(&pixel, (sizeof(char)), 3, outputFile); //pixel中的信息写
373                 入输出图像中
374             }
375             for (int l = 0; l < padding; l++) {
376                 fread(&pad, 1, 1, inputFile); //读取末尾填充的图像信息
377                 fwrite(&pad, 1, 1, outputFile); //向输出图像写入末尾填充的图像信息
378             }
379         }
380     int elms_read; //确保没有空像素被读取
381     while (1) {
382         elms_read = fread(&pad, 1, 1, inputFile);
383         if (elms_read == 0) {
384             break;
385         }
386         else {
387             fwrite(&pad, 1, 1, outputFile);
388         }
389     }

```

```

381     }
382 }
383 fclose(inputFile);
384 fclose(outputFile);
385 return 1;
386 }
387
388 int map(int r, int c, int width) {
389     return r * width + c;
390 }
391
392 int pixeldata(const char* input) {
393     inputFile = fopen(input, "rb");
394     if (inputFile == NULL) {
395         printf("并无此文件,请检查main函数中定义的路径是否正确!\n");//文件读取失败,
重新修改文件路径
396         return 0;
397     }
398
399     char header[54];
400     fread(&header, sizeof(char), 54, inputFile); //获得输入图像头部信息
401
402     int width = *(int*)&header[18]; //获得图像宽度
403     int height = abs(*(int*)&header[22]); //获得图像高度
404     int stride = (4 * ((8 * 3) * width) / 32); //计算行的数量
405     int paddingCheck = (stride % 4);
406     int padding;
407     if (paddingCheck != 0) { //如果行数不能被4整除
408         int padCalc = 0;
409         for (int i = 1; padCalc < stride; i++) {
410             padCalc = i * 4;
411         }
412         padding = padCalc - stride;
413     }
414     else
415         padding = 0;
416
417     char pad;
418     char pixel[3];
419     int merge[256];
420     int count[256];
421     for (int i = 0; i < 256; ++i)
422     {
423         count[i] = 0;
424     }
425     printf("像素数据为: \n");
426     int cnt = 0; bool flag = 0;
427     for (int j = 0; j < height; j++) {
428         if (flag) break;
429         for (int k = 0; k < width; k++) {
430             fread(pixel, sizeof(char), 3, inputFile); //读取RGB信息,并写入
pixel中
431             if (cnt < 256)
432             {
433                 merge[cnt]= abs((int)pixel[0])*0.3+
abs((int)pixel[1])*0.59+abs((int)pixel[2])*0.11;//归一化
434                 cnt++;
435             }

```

```

436         else {
437             flag = 1;
438             break;
439         }
440
441     }
442     for (int l = 0; l < padding; l++) { //读取末尾填充的图像信息
443         fread(&pad, 1, 1, inputFile);
444     }
445 }
446 int elms_read; //确保没有空像素被读取
447 while (1) {
448     elms_read = fread(&pad, 1, 1, inputFile);
449     if (elms_read == 0) {
450         break;
451     }
452 }
453
454 printf("\n归一化后的直方图数据为: \n");
455 for (int i = 0; i < 256; ++i)
456 {
457     printf("%3d\t", merge[i]);
458     count[merge[i]]++;
459     if ((i + 1) % 16 == 0) //每输出16个,就换行
460         printf("\n");
461 }
462 printf("\n");
463 /*直方图*/
464 for (int i = 1; i < 256; ++i)
465 {
466     if (count[i] == 0) //只输出大于0的直方图
467         continue;
468     printf("%d :", i);
469     for (int j = 1; j <= count[i]; ++j)
470     {
471         printf("*");
472     }
473     printf("\n");
474 }
475
476 fclose(inputFile);
477 return 1;
478 }
479
480 int showcolorpad(const char* input) { //8/ 显示调色板
481     inputFile = fopen(input, "rb");
482     if (inputFile == NULL) {
483         printf("并无此文件,请检查main函数中定义的路径是否正确!\n"); //文件读取失败,
484         //重新修改文件路径
485         return 0;
486     }
487     fseek(inputFile, sizeof(BITMAPFILEHEADER), 0);
488     BITMAPINFOHEADER head;
489     fread(&head, sizeof(BITMAPINFOHEADER), 1, inputFile);
490     unsigned int width = head.biWidth;
491     unsigned int height = head.biHeight;
492     unsigned int bitcount = head.biBitCount;
493     int linebyte = (width * bitcount / 8 + 3) / 4 * 4;

```

```

493     RGBTRIPLE* pcolortable = new RGBTRIPLE[256];
494     if (bitcount == 8) {
495         fread(pcolortable, sizeof(RGBTRIPLE), 256, inputFile);
496     }
497     else {
498         printf("该图不是8位BMP图像,没有调色盘!\n");
499         return 0;
500     }
501     for (size_t i = 0; i < 256; ++i)
502     {
503         printf("%3d %3d %3d\t", (unsigned int)(pcolortable[i].rgbBlue),
504             (unsigned int)(pcolortable[i].rgbGreen), (unsigned int)
505             (pcolortable[i].rgbRed));
506         if ((i + 1) % 4 == 0) printf("\n"); //每输出4个数据,就换行
507     }
508     fclose(inputFile);
509     return 1;
510 }
511
512 int main(void) { //主函数
513     while (1)
514     {
515         int op;
516         printf("BMP图像处理作业菜单\n");
517         char infile[] = "C:/Users/yuehan lian/Desktop/a1.bmp"; //需要修改文件路
518         径
519         // 请输入BMP图像所在位置(例如:"C:/Users/yuehan lian/Desktop/1.bmp
520         printf("0. 结束啦\n");
521         printf("1. 读取BMP图像(8/24bit)\n");
522         printf("2. 保存BMP图像(8/24bit)\n");
523         printf("3. 显示图像数据\n");
524         printf("4. 计算直方图\n");
525         printf("5. 图像增强/减暗\n");
526         printf("6. 反色\n");
527         printf("7. 灰度图(同位转换,不改变位数)\n");
528         printf("8. 显示调色板(8bit)\n");
529
530         printf("\n输入选项: ");
531         scanf("%d", &op);
532         printf("\n");
533         switch (op)
534         {
535             case 0: {
536                 printf("感谢你的使用!\n");
537                 return 0;
538                 break;
539             }
540             case 1: {
541                 if (read(infile) == 0) {
542                     return 0; // 1 读图像
543                 }
544                 printf("读取成功!\n");
545                 break; }
546             case 2: {
547                 if (save(infile) == 0) {
548                     return 0; // 2 保存图像
549                 }
550                 printf("保存成功!\n");
551                 break; }
552         }
553     }
554 }

```



```
548     case 3: {
549         if (show(infile) == 0) {
550             return 0; } // 3 显示图像数据
551         printf("显示完毕!\n");
552         break; }
553     case 4: {
554         if (pixeldata(infile) == 0) {
555             return 0; } // 4 直方图
556         printf("显示完毕!\n");
557         break; }
558     case 5: {
559         if (intense(infile) == 0) {
560             return 0; } // 5 图像增强/减暗
561         printf("保存成功!\n");
562         break; }
563     case 6: {
564         if (reverse(infile) == 0) {
565             return 0; } // 6 反色
566         printf("保存成功!\n");
567         break; }
568     case 7: {
569         if (greyScale(infile) == 0) {
570             return 0; } // 7 灰度图
571         printf("保存成功!\n");
572         break; }
573     case 8: {
574         if (showcolorpad(infile) == 0) {
575             return 0; } // 8 显示调色板
576         printf("显示完毕!\n");
577         break; }
578     default:
579         break;
580     }
581 }
582
583 return 0;
584 }
```