2018级课程设计任务书

一、设计题目　　数字图像处理—旋转图像

二、设计内容

【题目描述】

读入一幅灰度图像，给定旋转角度，将该图像绕中心点旋转该角度。

【题目要求】

（1）读入bmp格式图像；

（2）判断是彩色还是灰度图像，若是彩色转化为灰度；

（3）给定角度，通过计算，原图中的某个位置上的像素点变成目标图中的另一个位置点。

（4）将旋转后的数据保存成bmp格式文件。

【输入/输出要求】

（1）应用程序运行后，先显示一个菜单，然后用户根据需要选择相应的操作项目。进入每个操作后，根据程序的提示输入相应的信息；

（2）输出每个功能的效果图。

三、基本要求

1、编写源程序的要求：

(1)能够实现任务书中的功能；

(2)尽可能使界面友好、直观、易操作

(3)源程序要有适当的注释，使程序容易阅读。

2、撰写“课程设计报告”，要求如下：

（1）封面：统一采用《常州大学课程设计说明书》封面格式

（2）任务书

（3）目录

（4）“课程设计报告”正文

3、课程设计验收要求：

(1)运行所设计的系统；

(2)回答有关问题；

(3)提交课程设计报告；

(4)提交源程序。

四、进度安排

1、系统分析、设计准备阶段：4学时

2、编程调试阶段：30学时

3、总结和书写课程设计报告阶段：2学时

4、机房考核阶段：4学时

目录

1. 需求分析

1.1 系统的功能需求

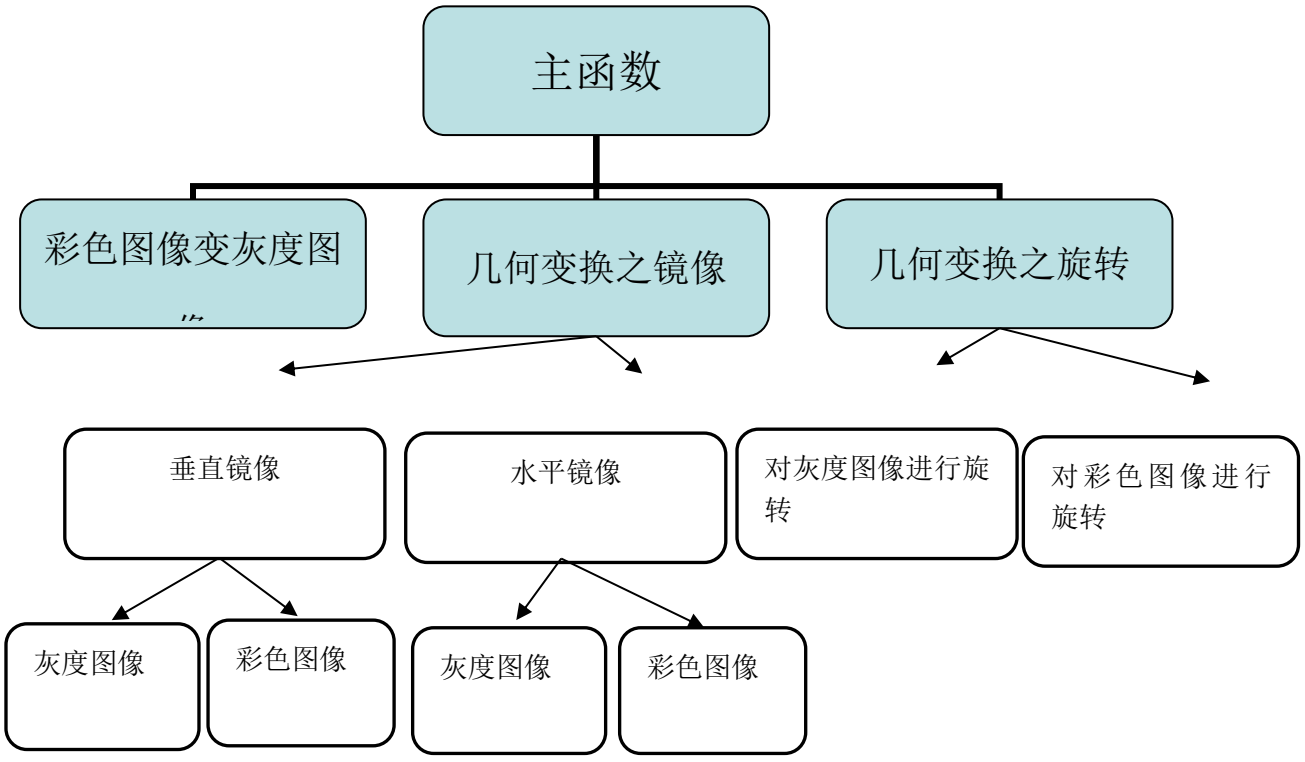
读入一幅彩色的数字图像，完成一系列的几何运算，并输出每个运算的效果图

1. 将彩色图像变为灰度图像
2. 将灰度图像旋转任意角度；并对彩色图像进行相应旋转

1.2 用户操作流程

编译链接后出现对话框和原始图片，按任意键即可在对话框中输入，根据对话框提示输入角度完成对应功能，出现功能图像按任意键摧毁窗口，即可继续输入完成对应功能。输入 0 结束变换，按任意键退出对话框。

2. 概要设计



3. 详细设计

C 语言结构体定义:

```
struct bmp_file     //BMP 文件头结构
{
   char type[2];            //位图文件的类型，必须为 BM，我这里类型
不对，所以显示有误。
   unsigned int size;       //位图文件的大小，以字节为单位
   short rd1;               // 位图文件保留字，必须为 0
   short rd2;               // 位图文件保留字，必须为 0
   unsigned int offset;    // 位图数据的起始位置，以相对于位图
};
struct bmp_info            //图像信息区
{
   unsigned int bsize;      //本结构体所占用字节数,即 40 个字节
   int width;              // 位图的宽度，以像素为单位，像素数量是 4
字节对齐的
   int height;              // 位图的高度，以像素为单位
   unsigned short planes;       // 目标设备的级别，必须为 1
   unsigned short count;        // 每个像素所需的位数,必须是 1(双色）
// 4(16 色)，8(256 色)或 24(真彩色)之一
   unsigned int compression;    // 位图压缩类型，必须是 0(不压
```

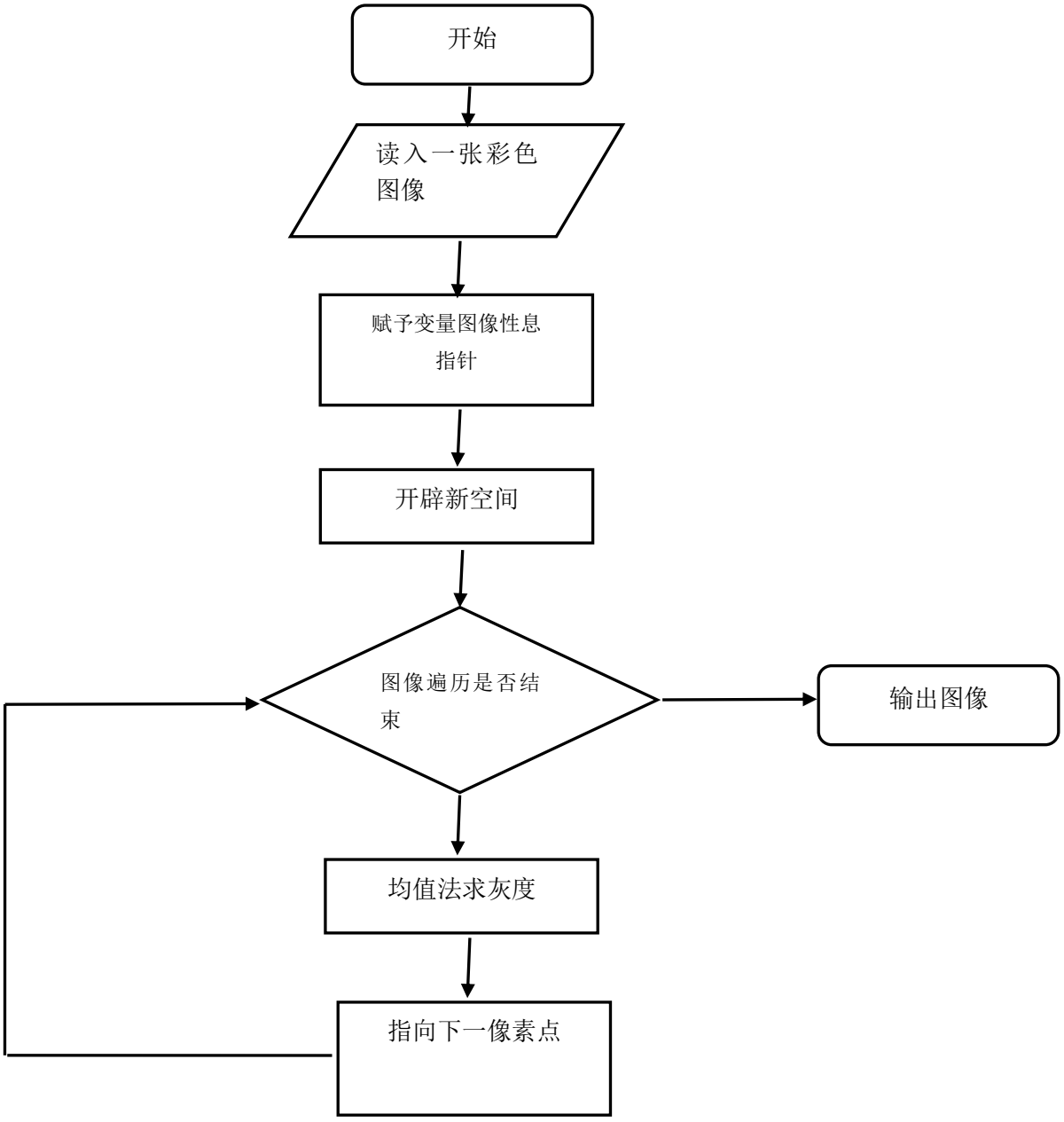
```
缩),// 1(BI_RLE8 压缩类型)或 2(BI_RLE4 压缩类型)之一
   unsigned int sizeimage;      // 位图的大小，以字节为单位
   unsigned int xmeter;        // 位图水平分辨率，每米像素数
   unsigned int ymeter;         // 位图垂直分辨率，每米像素数
   unsigned int cused;         // 位图实际使用的颜色表中的颜色数
   unsigned int cimportant;      // 位图显示过程中重要的颜色数
};
struct bmp_head {
   struct bmp_file file;
   struct bmp_info info;
};
  struct bmp_attr {
   struct bmp_head head;
   int xsize, ysize,sizeimage;
   int pixel_size;
   int line_length;
   unsigned int offset;
};
struct bmp_attr *bmp = (struct bmp_attr *)calloc(1, sizeof(struct
bmp_attr));
   //读取 bmp 的文件头
   fread(bmp, sizeof(struct bmp_attr), 1, fp);
```

```
    bmp->xsize = (bmp->head.info.width * 3 + 3) / 4 * 4; // 4 字节补齐
    bmp->ysize = bmp->head.info.height;
    bmp->sizeimage = bmp->head.info.sizeimage;
    bmp->offset = bmp->head.file.offset;
    printf("bmp xsize = %d ysize = %d sizeimage = %d offset = %d\n",bmp->xsize,bmp->ysize,bmp->sizeimage,bmp->offset);
```

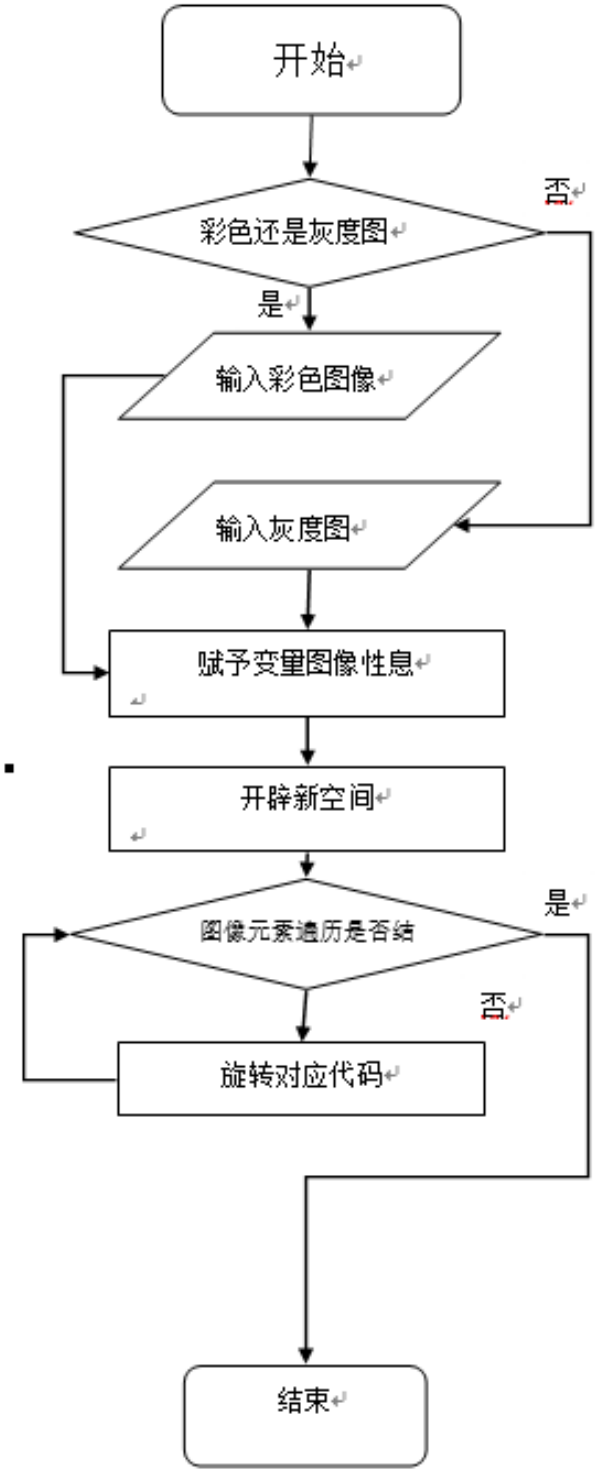
下面关于图片数据信息的获取

```
for (ix = 0; ix < bmp->ysize; ++ix) {
        //因为 bmp 文件的原点是左下角，所以 bmp 图片需要顺着 y 轴的反方向来读取
        fseek(fp, bmp->head.file.offset + (bmp->ysize -1 - ix) * bmp->line_length+1, SEEK_SET); //最后一行开始读取
        //读取一行像素数
        fread(buf, 1, bmp->line_length, fp);
```

3.1 彩色图像变成灰度图像

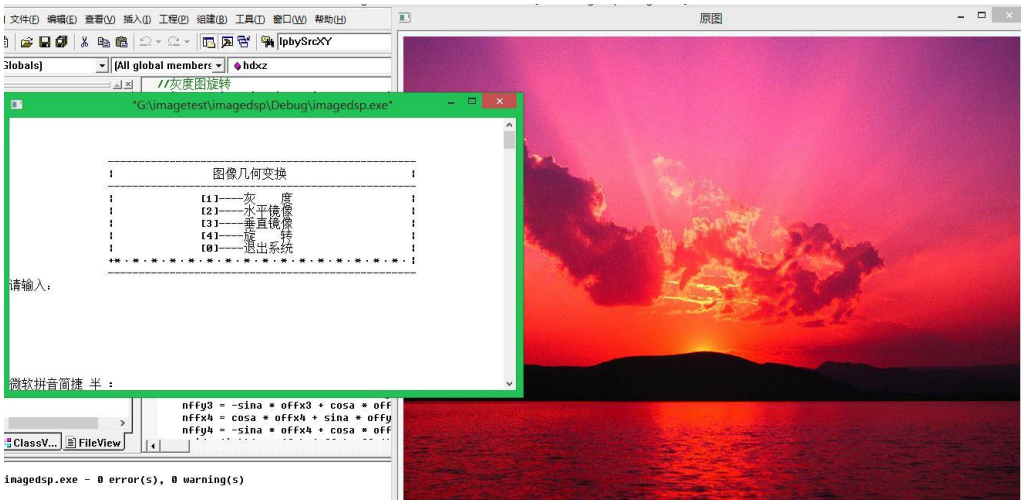


3.2 几何变换--旋转

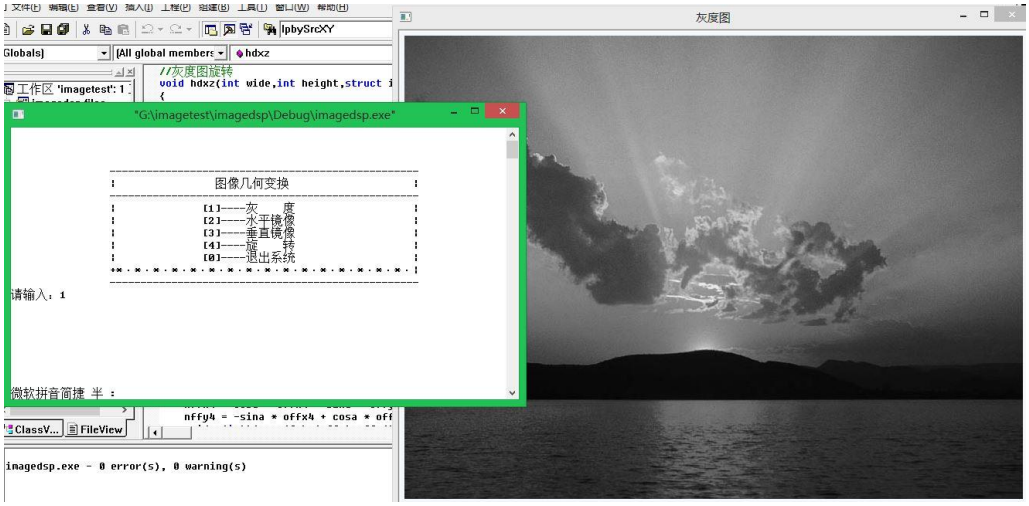

开始
彩色还是灰度图
否
是
输入彩色图像
输入灰度图
赋予变里图像性息
开辟新空间
图像元素遍历是否结
是
否
旋转对应代码
结束

4、调试运行

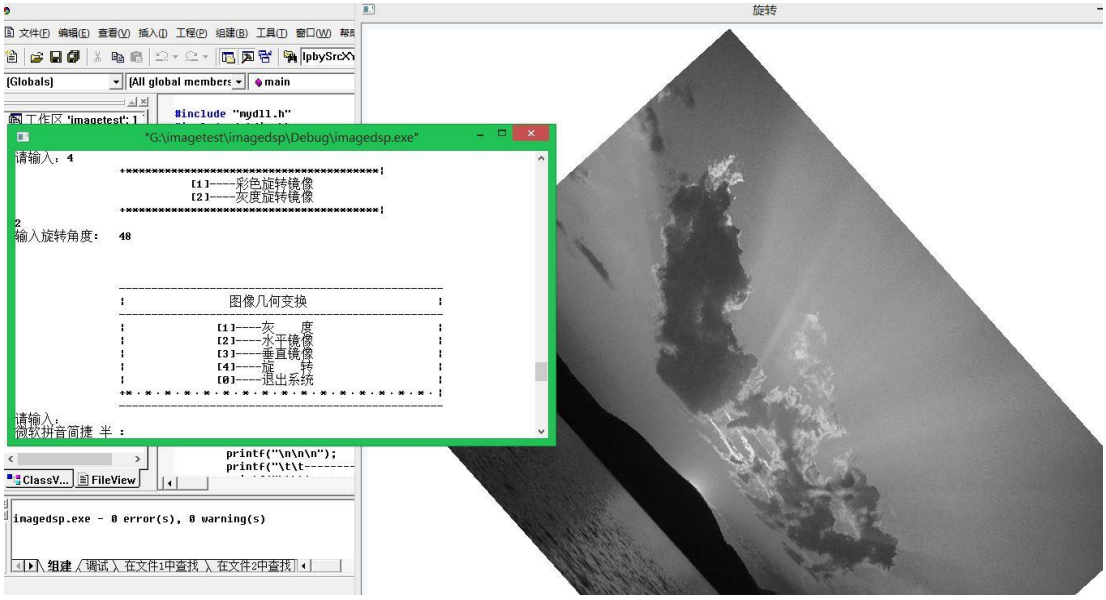
4.1 原图



4.2 彩色图像变为灰度图像



4.3 灰度图像旋转(48)



5、心得体会

刚开始并不了解如何用 vc 来进行几何变换，后来经过初步认识，了解到各门学科的联系，c 语言中用到了很多算法和数学有着紧密的联系，感受到数组与指针的强大，尤其是矩阵对变化的作用。但对比 openCV，仅用 C 语言效率太低，因此以后对库的更新要保持格外敏感。

编程任然需要注意细节，不然犯错了在去找问题所在很难找出问题错在哪里。基础知识要熟练掌握，要养成作注解的习惯，不然容易生疏忘记，也方便查找问题。多上机，发现问题解决问题，才能积累经验，在今后编程中少犯错误。

6、实习日志

6 月 17 日
安排：学习图像基本知识，完成彩色图像变灰度图像
进度：完成了图像导入，但并没有完成真正灰度
6 月 18 日
安排：解决上次问题，学习彩图转灰度
进度：解决了上次问题，完成灰度图像的转换
遇到的问题：彩色图像转灰度后比例出现问题
解决办法：调整指针指向
6 月 19 日
安排：解决上次问题，学习图像旋转基础知识。
进度：完成良好
6 月 20 日
安排：完成旋转
进度：只完成特殊值旋转
遇到问题：地址访问错误
解决办法：调试逐句查看，查看哪里地址访问错误
6 月 21 日
安排：完成界面设计，图像旋转
进度：界面设计完成良好

7、参考文献

《VC++数字图像处理》何斌 人民邮电出版社

8、C 语言核心源程序代码

```
#include "mydll.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#define PI 3.14159265
void main()
{
    double maxa(double a,double b);//最大值
    void hd(int wide,int height,unsigned char*pdata,unsigned char*gray);   //灰度
    void spjx(int wide,int height,unsigned char*pdata,int numcolors);       //水平镜像
    void czjx(int wide,int height,unsigned char*pdata,int numcolors);       //垂直镜像
    void hdxz(int wide,int height,struct image*p1,struct image *p2,int k); //旋转
    struct image img1,img2,*p2=&img2,*p1=&img1;
    unsigned char *pdata,*gray;
    int i,k,n=3,m;
    int wide,height;
    imgfiletomat("G:\\ 1.jpg", &img1); //显示原始图像
    createwindow("原图");
    imgshow("原图",&img1);
    wait(0); //等待按键
    destroywindow("原图");
    height=img1.height;//像素高度
    wide=img1.width;    //像素宽度
    for(i=0;(n!=0);i++)
    {
        printf("\n\n\n");
        printf("\t\t-------------------------------------------------------------------------\n");
        printf("\t\t|                          图像几何变换                                 |\n");
        printf("\t\t-------------------------------------------------------------------------\n");
        printf("\t\t|                      [1]----灰      度                                |\n");
        printf("\t\t|                      [2]----水平镜像                                  |\n");
        printf("\t\t|                      [3]----垂直镜像                                  |\n");
        printf("\t\t|                      [4]----旋      转                                |\n");
        printf("\t\t|                      [0]----退出系统                                  |\n");
        printf("\t\t+*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*.*|\n");
        printf("\t\t-------------------------------------------------------------------------\n");
        printf("请输入： ");
        scanf("%d",&n);
        switch(n)
```

```
{
    case 0: break;                                                    ~7~
    case 1:  imgfiletomat("G:\\ 1.jpg", &img1);pdata=img1.ptr;
             if(img1.numcolors==3)
             {
                 img1.numcolors=1;img1.ptr=gray=(unsignedchar*)malloc(wide*height);
                 hd(wide,height,pdata,gray);
                 createwindow("灰度图");
                 imgshow("灰度图",&img1);
                 wait(0); //等待按键
                 mattoimgfile("G:\\ c1.jpg",&img1);
                 free(gray);
                 gray=NULL;
                 destroywindow("灰度图");
             }
             else
             {
                 printf("\t\t+--------------------------------------|\n");
                 printf("                    请换彩色图                    \n");
                 printf("\t\t+--------------------------------------|\n");
             }
             break;
    case 2:  printf("\t\t+--------------------------------------|\n");
             printf("\t\t          [1]----彩色水平镜像          \n");
             printf("\t\t          [2]----灰度水平镜像          \n");
             printf("\t\t+--------------------------------------|\n");
             printf("请输入：");
             scanf("%d",&m);
             if(m==1)
             {imgfiletomat("G:\\ 1.jpg", &img1);pdata=img1.ptr;}
             else
             {imgfiletomat("G:\\ c1.jpg", &img1);pdata=img1.ptr;}
             spjx(wide,height,pdata,img1.numcolors);
             createwindow("水平镜像图");
             imgshow("水平镜像图",&img1);
             wait(0); //等待按键
             destroywindow("水平镜像图");
             break;
    case 3:  printf("\t\t+--------------------------------------|\n");
             printf("\t\t          [1]----彩色垂直镜像          \n");
             printf("\t\t          [2]----灰度垂直镜像          \n");
             printf("\t\t+--------------------------------------|\n");
             printf("请输入：");
             scanf("%d",&m);
```

```
                    if(m==1)
                    {imgfiletomat("G:\\ 1.jpg", &img1);pdata=img1.ptr;}
                    else
                    {imgfiletomat("G:\\ c1.jpg", &img1);pdata=img1.ptr;}
                    czjx(wide,height,pdata,img1.numcolors);
                    createwindow("垂直镜像图");
                    imgshow("垂直镜像图",&img1);
                    wait(0); //等待按键
                    destroywindow("垂直镜像图");
                    break;
            case 4: printf("\t\t+****************************************|\n");
                    printf("\t\t          [1]----彩色旋转镜像          \n");
                    printf("\t\t          [2]----灰度旋转镜像          \n");
                    printf("\t\t+****************************************|\n");
                    printf("请输入： ");
                    scanf("%d",&m);
                    if(m==1)
                    {imgfiletomat("G:\\ 1.jpg", &img1);pdata=img1.ptr;}
                    else
                    {imgfiletomat("G:\\ c1.jpg", &img1);pdata=img1.ptr;}
                    printf("输入旋转角度:\t");
                    scanf("%d",&k);
                    hdxz(wide,height,p1,p2,k);
                    createwindow("旋转");
                    imgshow("旋转",&img2);
                    wait(0); //等待按键
                    destroywindow("旋转");
                    break;
            default:
                printf("\t\t+****************************************|\n");
                printf("                  输入错误                  \n");
                printf("\t\t+****************************************|\n");
        }
    }
    destroyallwindows();
}
double maxa(double a,double b)
{
    return (a>b?a:b);
}
//灰度
void hd(int wide,int height,unsigned char*pdata,unsigned char*gray)
{
    int i,j,k=0;
```

```
    for(i=0;i<height;i++)//进行遍历元素                                    ~9~
        for(j=0;j<wide;j++)
        {//将原图 RGB 的值根据公式赋给新开辟的空间
            *(gray++)=(*(pdata+k)*11+*(pdata+k+1)*59+*(pdata+k+2)*30)/100;
            k+=3;
        }
}
//水平镜像
void spjx(int wide,int height,unsigned char*pdata,int numcolors)
{
    unsigned char*temp,*ogray,*ngray,*p=pdata;
    int i,j,k,n=numcolors,m;
    temp=(unsigned char*)malloc(wide*height*numcolors);    //开辟一个空间用以存放数据
    for(i=0;i<height;i++)//对元素进行遍历
    {
        k=wide*n-1;
        for(j=0;j<wide;j++)
        {
            for(m=numcolors;m>0;m--)//灰度与彩色的转换
            {
                ogray=pdata++;//遍历原图每一个值
                ngray=temp+wide*i+k-j*numcolors-(m-1);//对应新开辟空间的地址
                *ngray=*ogray;//把数据存放在新开辟的空间里
            }
        }
        if(numcolors==3)
                n+=2;
    }
    memcpy(p,temp,wide*height*numcolors);//数据拷贝
    free(temp);//释放开辟的空间
    temp=NULL;//防止野指针
}
//垂直镜像
void czjx(int wide,int height,unsigned char*pdata,int numcolors)
{
    unsigned char*temp,*lpdst,*lpsrc;
    int i,j;
    temp=(unsigned char*)malloc(wide*height*numcolors);//开辟一个空间用以存放数据
    for(i=0;i<wide*numcolors;i++)//遍历数据并把数据存放在新开辟的空间里
        for(j=0;j<height;j++)
        {
            lpsrc=pdata+wide*numcolors*j+i;//数据地址
            lpdst=temp+wide*numcolors*(height-1-j)+i;//相应新开辟的空间的地址
```

```
                *lpdst=*lpsrc;//把数据存放在新开辟的空间里
            }                                                                    ~ 10 ~
        memcpy(pdata,temp,wide*height*numcolors);//数据拷贝
        free(temp);//释放开辟的空间
        temp=NULL;//防止野指针
}
//灰度图旋转
void hdxz(int wide,int height,struct image*p1,struct image *p2,int k)
{
        unsigned char*temp,*pdata;
        double offx1,offy1,offx2,offy2;//原图四个角的坐标
        double offx3,offy3,offx4,offy4;
        double nffx1,nffy1,nffx2,nffy2;//新图四个角的坐标
        double nffx3,nffy3,nffx4,nffy4;
        double cosa,sina,a,b;
        int x0,y0,x1,y1,nwide,nheight,n=0,m=0;
        pdata=p1->ptr;
        cosa=cos(PI*k/180.0); //角度变弧度
        sina=sin(PI*k/180.0);
        offx1 = -0.5 * wide;//以中心为原点计算原图四个角的坐标
        offy1 = 0.5 * height;
        offx2 = 0.5 * wide;
        offy2 = 0.5 * height;
        offx3 = -0.5 * wide;
        offy3 = -0.5 * height;
        offx4 = 0.5 * wide;
        offy4 = -0.5 * height;
        nffx1 = cosa * offx1 + sina * offy1;//旋转后新图四个角的坐标
        nffy1 = -sina * offx1 + cosa * offy1;
        nffx2 = cosa * offx2 + sina * offy2;
        nffy2 = -sina * offx2 + cosa * offy2;
        nffx3 = cosa * offx3 + sina * offy3;
        nffy3 = -sina * offx3 + cosa * offy3;
        nffx4 = cosa * offx4 + sina * offy4;
        nffy4 = -sina * offx4 + cosa * offy4;
        nwide=(int)(maxa(fabs(nffx4-nffx1),fabs(nffx3-nffx2))+0.5);//新图的宽和高
        nheight=(int)(maxa(fabs(nffy4-nffy1),fabs(nffy3-nffy2))+0.5);
        p2->width=nwide;//给 img2 赋值
        p2->height=nheight;//给 img2 赋值
        p2->numcolors=p1->numcolors;//给 img2 赋值
        //开辟一个空间用以存放数据
        p2->ptr=temp=(unsigned char *)malloc(nwide*nheight*(p2->numcolors));
        a = - 0.5 * nwide * cosa - 0.5 * nheight * sina + 0.5 * wide;//旋转常值
```

```
    b =   0.5 * nwide * sina - 0.5 * nheight * cosa + 0.5 * height;//旋转常值
    if(p2->numcolors==3)//判断灰度图，彩色图
    {                                                                    ~ 11 ~
        for(y1=0;y1<nheight;y1++)//进行彩色图元素遍历
        {
            for(x1=0;x1<nwide;x1++)
            {
                x0=(int)(x1*cosa+y1*sina+a);//算出新坐标对应的原图坐标
                y0=(int)(-x1*sina+y1*cosa+b);
              //判断是否超过原图，超过赋 255
                if((x0<wide)&&(x0>=0)&&(y0<height)&&(y0>=0))
                {    //将原图 RGB 值赋予新开辟空间对应的位置
                    *(temp++)=*(pdata+y0*wide*p2->numcolors+m);
                    *(temp++)=*(pdata+y0*wide*p2->numcolors+m+1);
                    *(temp++)=*(pdata+y0*wide*p2->numcolors+m+2);
                }
                else
                {
                    *(temp++)=255;//将没有对应点的值赋 255，使其变白色
                    *(temp++)=255;
                    *(temp++)=255;
                }
            }
            m+=3;
        }
    }
    else
    {    //进行灰度图元素遍历
        for(y1=0;y1<nheight;y1++)
        for(x1=0;x1<nwide;x1++)
       {
            x0=(int)(x1*cosa+y1*sina+a);//算出新坐标对应的原图坐标
            y0=(int)(-x1*sina+y1*cosa+b);
          //判断是否超过原图，超过赋 255
            if((x0<wide)&&(x0>=0)&&(y0<height)&&(y0>=0))
              *(temp++)=*(pdata+y0*wide+x0);//将原图的值赋予新开辟空间对应的位置
            else
              *(temp++)=255;//将没有对应点的值赋 255，使其变白色
       }
    }
}
```