```
#include "stdafx.h"
#include "funtions.h"


//绘制直线
int drawLines()
{
cv::Mat displayMat = cv::Mat::zeros(500, 500, CV_8UC3);

//0.绘制的Mat图像
//1.线段起点
//2.线段终点
//3.线段颜色
//4.线段粗细
//5.线段的连接方法，4邻接，8邻接，antialiased连接
//6.坐标点的小数点位数。

// 红色 ，宽度为3，4近邻居连接
cv::line(displayMat, cv::Point(100, 100), cv::Point(400, 105), cv::Scalar(0, 0, 200), 3, 4);
// 绿色，宽度为5，8近邻居连接
cv::line(displayMat, cv::Point(100, 200), cv::Point(400, 205), cv::Scalar(0, 200, 0), 5, 8);
// 蓝色，宽度为10，antialiased线段连接法
cv::line(displayMat, cv::Point(100, 300), cv::Point(400, 305), cv::Scalar(200, 0, 0), 10, CV_AA);

cv::namedWindow("lines", CV_WINDOW_AUTOSIZE | CV_WINDOW_FREERATIO);
cv::imshow("lines", displayMat);
cv::waitKey(0);

destroyAllWindows();

return 0;
}

int drawRectangles()
{
cv::Mat displayMat = cv::Mat::zeros(500, 500, CV_8UC3);

//0.绘制的Mat图像
//1.矩形上的顶点1
//2.顶点1的对角点
//3.线段颜色
//4.线段粗细，如果该参数是-1的话，则绘制实心的矩形
//5.线段的连接方法，4邻接，8邻接，antialiased连接
//6.坐标点的小数点位数。
// 红色 ，宽度为3，4近邻居连接
cv::rectangle(displayMat, cv::Point(200, 50), cv::Point(300, 150), cv::Scalar(0, 0, 200), 3, 4);
// 绿色，宽度为5，8近邻居连接
cv::rectangle(displayMat, cv::Point(200, 200), cv::Point(300, 300), cv::Scalar(0, 200, 0), 5, 8);
//另一种绘制方式是，输入参数使用cv::Rect，而不是用两个点
//0.绘制的Mat图像
//1.被绘制的矩形
//2.线段颜色
//3.线段粗细，如果该参数是-1的话，则绘制实心的矩形
//4.线段的连接方法，4邻接，8邻接，antialiased连接
//5.坐标点的小数点位数。
// 蓝色，矩形内部填色，antialiased线段连接法
cv::Rect rect;
rect.x = 200;
rect.y = 350;
rect.width = 50;
rect.height= 100;
cv::rectangle(displayMat, rect, cv::Scalar(200, 0, 0), -1, CV_AA);
cv::namedWindow("drawing", CV_WINDOW_AUTOSIZE | CV_WINDOW_FREERATIO);
cv::imshow("drawing", displayMat);
cv::waitKey(0);
destroyAllWindows();
return 0;
}

int drawCircles()
{
cv::Mat displayMat = cv::Mat::zeros(500, 500, CV_8UC3);

//0.绘制的Mat图像
//1.圆心坐标
//2.圆半径
//3.圆的颜色
//4.线段粗细，如果该参数是-1的话，则绘制实心圆
//5.线段的连接方法，4邻接，8邻接，antialiased连接
//6.坐标点的小数点位数。

//红色
cv::circle(displayMat, cv::Point(300, 100), 100, cv::Scalar(0, 0, 200), 3, 4);
//绿色
cv::circle(displayMat, cv::Point(200, 250), 120, cv::Scalar(0, 200, 0), 8, 8);
//蓝色
cv::circle(displayMat, cv::Point(300, 400), 80, cv::Scalar(200, 0, 0), -1, CV_AA);

cv::namedWindow("drawing", CV_WINDOW_AUTOSIZE | CV_WINDOW_FREERATIO);
cv::imshow("drawing", displayMat);
cv::waitKey(0);
destroyAllWindows();
```

```
return 0;
}

int drawEllipse()
{
cv::Mat displayMat = cv::Mat::zeros(800, 600, CV_8UC3);
double angle;

//0.绘制的Mat图像
//1.中心坐标
//2.长轴，短轴
//3.旋转角度，水平为0度，顺时针为正值
//4.圆弧开始角度，水平向右为0度，顺时针为正值
//5.圆弧结束角度
//6.颜色
//7.线段粗细，如果该参数是-1的话，则绘制实心圆
//8.线段的连接方法，4邻接，8邻接，antialiased连接
//9.坐标点的小数点位数。

cv::ellipse(displayMat, cv::Point(150, 150), cv::Size(50, 10), 0, 30, 360, cv::Scalar(0, 0, 200), 1, 4);

angle = 30;
cv::ellipse(displayMat, cv::Point(400, 150), cv::Size(200, 100), angle, angle - 100, angle + 200, cv::Scalar(0, 0, 200), 3, 4);

angle = 0;
//相当于画一个圆
cv::ellipse(displayMat, cv::Point(200, 200), cv::Size(100, 100), angle, angle, angle + 360, cv::Scalar(0, 200, 0), 5, 8);

angle = 100;
cv::ellipse(displayMat, cv::Point(200, 400), cv::Size(100, 200), angle, angle - 200, angle + 100, cv::Scalar(200, 0, 0), -
1, CV_AA);

cv::namedWindow("drawing", CV_WINDOW_AUTOSIZE | CV_WINDOW_FREERATIO);
cv::imshow("drawing", displayMat);
cv::waitKey(0);

destroyAllWindows();

return 0;
}

int drawMarkers()
{
cv::Mat displayMat = cv::Mat::zeros(800, 600, CV_8UC3);



//0.绘制的Mat图像
//1.中心坐标
//2.颜色
//3.标记类型
//4.标记的尺寸，默认20pixesl
//5.线段粗细
//6.线段的连接方法，4邻接，8邻接，antialiased连接

//加号
drawMarker(displayMat, Point(100, 50), cv::Scalar(0, 255, 255),0,20,1,8);
//叉
drawMarker(displayMat, Point(100, 100), cv::Scalar(0, 255, 255),1);
//星
drawMarker(displayMat, Point(100, 200), cv::Scalar(0, 255, 255),2);
//方片
drawMarker(displayMat, Point(100, 250), cv::Scalar(0, 255, 255),3);
//方块
drawMarker(displayMat, Point(100, 300), cv::Scalar(0, 255, 255),4);
//三角
drawMarker(displayMat, Point(100, 350), cv::Scalar(0, 255, 255),5);
//倒三角
drawMarker(displayMat, Point(100, 400), cv::Scalar(0, 255, 255),6);


cv::namedWindow("drawing", CV_WINDOW_AUTOSIZE | CV_WINDOW_FREERATIO);
cv::imshow("drawing", displayMat);
cv::waitKey(0);

destroyAllWindows();

return 0;
}


//写字
int writeText()
{
cv::Mat img = cv::Mat::zeros(500, 500, CV_8UC3);

//定义不同字体
int face[] = { cv::FONT_HERSHEY_SIMPLEX, cv::FONT_HERSHEY_PLAIN, cv::FONT_HERSHEY_DUPLEX, cv::FONT_HERSHEY_COMPLEX,
cv::FONT_HERSHEY_TRIPLEX, cv::FONT_HERSHEY_COMPLEX_SMALL, cv::FONT_HERSHEY_SCRIPT_SIMPLEX,
cv::FONT_HERSHEY_SCRIPT_COMPLEX, cv::FONT_ITALIC };

cv::String ssss;
```

```
//std::string

//0.文字绘制的Mat图像
//1.被书写的文字，String类型
//2.文字的左下角的位置
//3.字体
//4.文字的尺寸参数
//5.文字的颜色
//6.文字的线条粗细
//7.文字的连接类型

for (int i=0; i < 8; i++) {

//int 转换为 字符
stringstream ss1;
string str1;
ss1 << i;
ss1 >> str1;

cv::putText(img, str1, cv::Point(30, (i+1)*50), face[i], 1.2, cv::Scalar(255, 255, 255), 2, CV_AA);

cv::putText(img, "OpenCV", cv::Point(100, (i+1)*50), face[i], 1.2, cv::Scalar(255, 255, 255), 2, CV_AA);
}

cv::namedWindow("drawing", CV_WINDOW_AUTOSIZE | CV_WINDOW_FREERATIO);
cv::imshow("drawing", img);
cv::waitKey(0);
destroyAllWindows();

return 0;
}
```