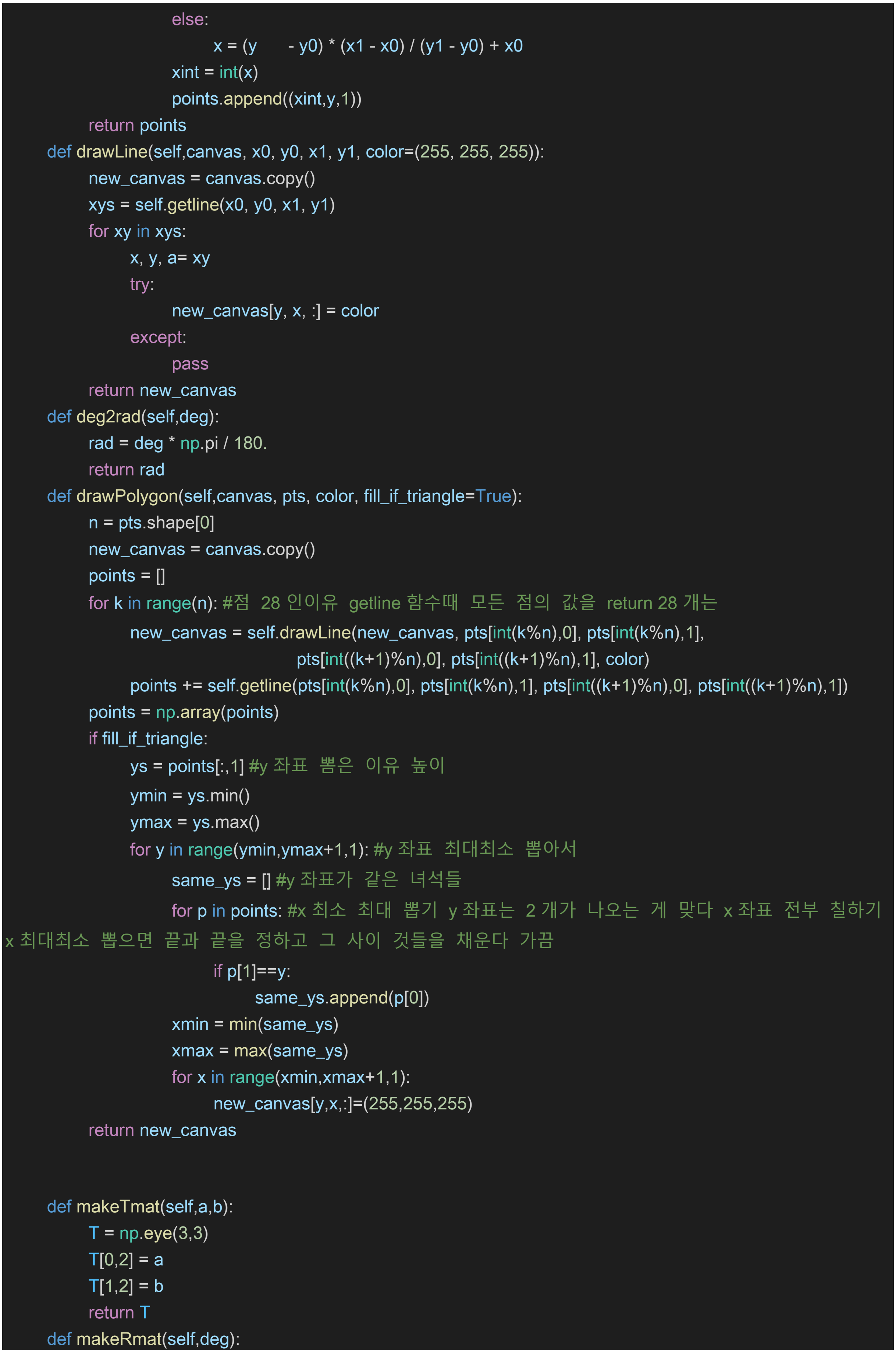
Midterm Catch Up

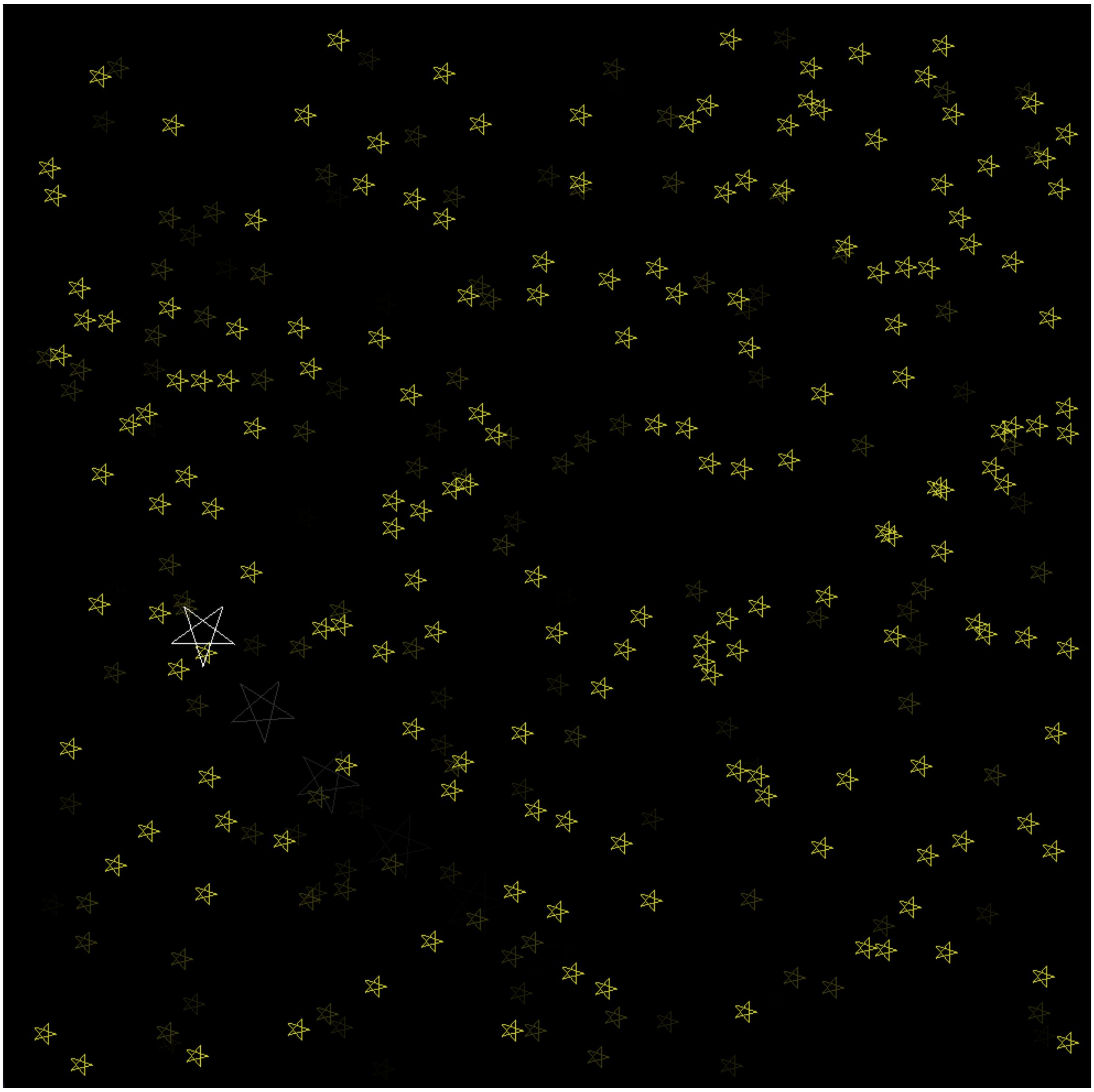
20211030 Jungwon Park

1. Twinkle Star

```
import numpy as np
import cv2


class Night_Star(object):
    def __init__(self , canvas_width = 1500 , canvas_height=1000 , rect_w = 100 ,rect_h = 100): #가로  세로
        self.canvas_width = canvas_width
        self.canvas_height = canvas_height
        self.rect_width = rect_w
        self.rect_height = rect_h
        self.canvas = np.zeros((self.canvas_height,self.canvas_width,3),dtype=np.uint8)
        self.pre_canvas1 = np.zeros((self.canvas_height,self.canvas_width,3),dtype=np.uint8)
        self.pre_canvas2 = np.zeros((self.canvas_height,self.canvas_width,3),dtype=np.uint8)
        self.pre_canvas = np.zeros((self.canvas_height,self.canvas_width,3),dtype=np.uint8)



        self.delta_theta = 15
    def getline(self,x0,y0,x1,y1):
        points = []
        if (x0==x1):
            x = x0
            if y0>y1:
                for y in range(y0,y1+1,-1):
                    points.append((x,y,1))
            else:
                for y in range(y0,y1+1):
                    points.append((x,y,1))
            # print(points)
        elif abs((y1-y0)/(x1-x0)) < 1:
            for x in range(x0,x1+1,1 if x0<x1 else -1):
                if x0 == x1:
                    y = y0
                else:
                    y = (x - x0) * (y1 - y0) / (x1 - x0) + y0
                yint = int(y)
                points.append((x,yint,1))
        else:
            for y in range(y0,y1+1,1 if y0<y1 else -1):
                if y0 == y1:
                    x = x0
```

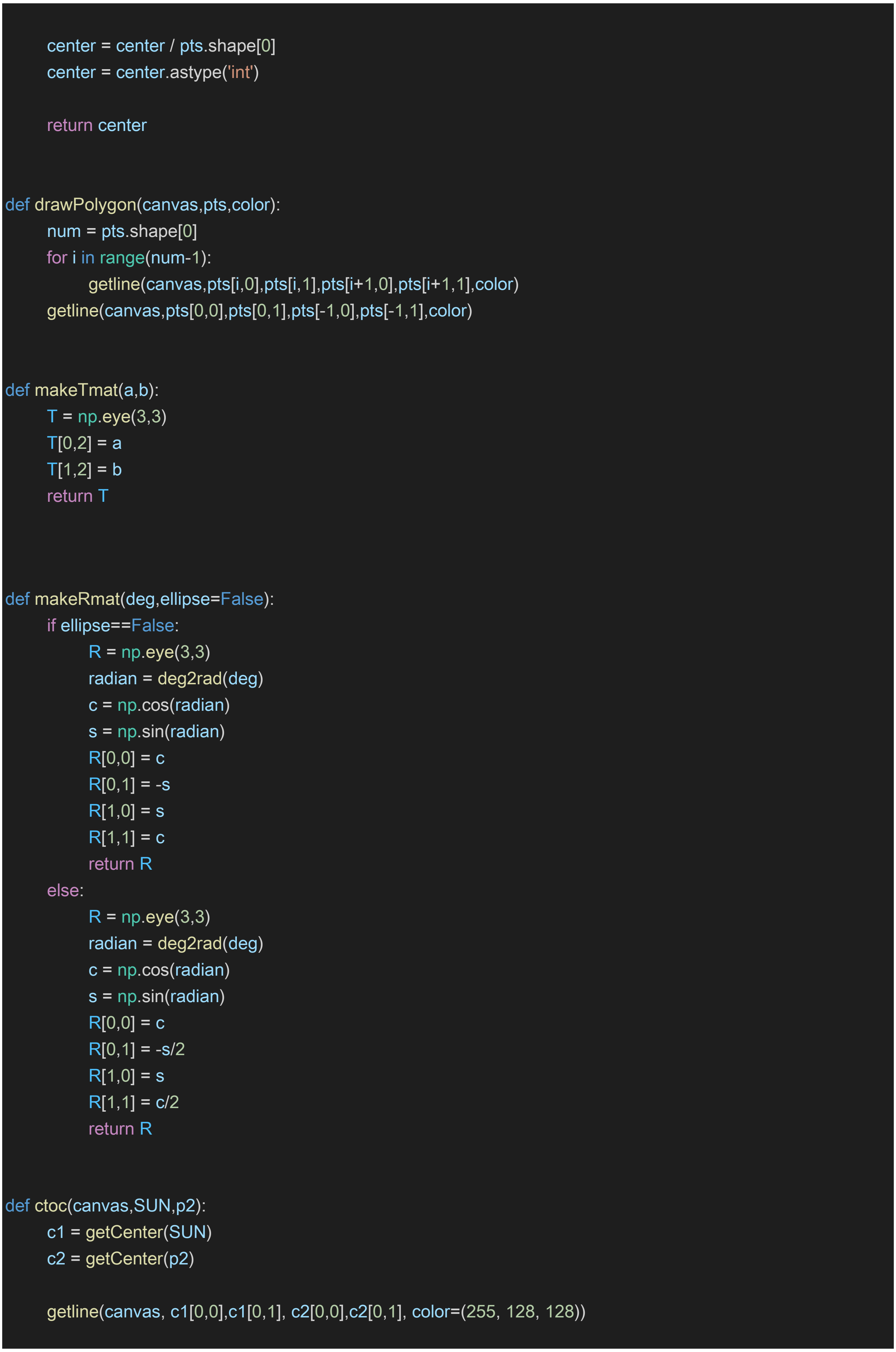


```
                else:
                    x = (y      - y0) * (x1 - x0) / (y1 - y0) + x0
                xint = int(x)
                points.append((xint,y,1))
        return points
    def drawLine(self,canvas, x0, y0, x1, y1, color=(255, 255, 255)):
        new_canvas = canvas.copy()
        xys = self.getline(x0, y0, x1, y1)
        for xy in xys:
            x, y, a= xy
            try:
                new_canvas[y, x, :] = color
            except:
                pass
        return new_canvas
    def deg2rad(self,deg):
        rad = deg * np.pi / 180.
        return rad
    def drawPolygon(self,canvas, pts, color, fill_if_triangle=True):
        n = pts.shape[0]
        new_canvas = canvas.copy()
        points = []
        for k in range(n): #점 28 인이유 getline 함수때 모든 점의 값을 return 28 개는
            new_canvas = self.drawLine(new_canvas, pts[int(k%n),0], pts[int(k%n),1],
                                        pts[int((k+1)%n),0], pts[int((k+1)%n),1], color)
            points += self.getline(pts[int(k%n),0], pts[int(k%n),1], pts[int((k+1)%n),0], pts[int((k+1)%n),1])
        points = np.array(points)
        if fill_if_triangle:
            ys = points[:,1] #y 좌표 뽑은 이유 높이
            ymin = ys.min()
            ymax = ys.max()
            for y in range(ymin,ymax+1,1): #y 좌표 최대최소 뽑아서
                same_ys = [] #y 좌표가 같은 녀석들
                for p in points: #x 최소 최대 뽑기 y 좌표는 2 개가 나오는 게 맞다 x 좌표 전부 칠하기
x 최대최소 뽑으면 끝과 끝을 정하고 그 사이 것들을 채운다 가끔
                    if p[1]==y:
                        same_ys.append(p[0])
                xmin = min(same_ys)
                xmax = max(same_ys)
                for x in range(xmin,xmax+1,1):
                    new_canvas[y,x,:]=(255,255,255)
        return new_canvas


    def makeTmat(self,a,b):
        T = np.eye(3,3)
        T[0,2] = a
        T[1,2] = b
        return T
    def makeRmat(self,deg):
        rad = self.deg2rad(deg)
        c = np.cos(rad)
        s = np.sin(rad)
        R =    np.eye(3,3)
        R[0,0] = c
        R[0,1] = -s
        R[1,0] = s
        R[1,1] = c
        return R
    def getrectangle(self,width,height):
        points = []
        points.append((0,0,1))
        points.append((width,0,1))
        points.append((width,height,1))
        points.append((0,height,1))
        points = np.array(points)
        return points
    def gettriangle(self,):
        points=[]
        points.append((0,0,1))
        points.append((200,-50,1))
        points.append((200,50,1))
        return np.array(points)

    def getRegularNgon(self,ngon,size=10): #별을 그리기 위해 여러 각을 그릴 수 있는 getregularNgon
함수를 만들었다
        vertices = []
        delta = 360.0/ngon
        for i in range(ngon):
            degree = delta * i
            radian = self.deg2rad(degree)
            x = np.cos(radian)
            y = np.sin(radian)
            vertices.append((x,y,1/size))
        vertices = np.array(vertices)
        return vertices*size
    def draw_diag(self,canvas,polygon,on_off = 1.0,color=(50,200,200)):
        if on_off>=0.4: #on off 40 이상일때만  그려라
            num = polygon.shape[0] #몇각형인지  //5 각형이면  5
            for i in range(num): #0-4
                for count in range(num):
                    if(2<=abs(i-count)<num-1):
                        canvas =
self.drawLine(canvas,polygon[count,0],polygon[count,1],polygon[i,0],polygon[i,1],color=color)
            return canvas
        else:
            return canvas


    def main(self):
        self.stars = [] #잔상 표현을 위해 self.star 자기자신을 설정하고 self.movingstar 을 이용해 잔상을 표현하고자 하였다.
        self.moving_star = self.getRegularNgon(5,30)
        self.moving_star_T_mat = self.makeTmat(0,-self.canvas_height)
        for i in range(300):
            a,b = np.random.randint(30,self.canvas_width-30),np.random.randint(30,self.canvas_height-30)
            Tmat = self.makeTmat(a,b)
            star = self.getRegularNgon(5)
            star = ((Tmat@star.T).T).astype('int')
            self.stars.append(star)
        theta1 = 270
        while True:
            new_canvas = self.canvas.copy()
            new_canvas = cv2.addWeighted(new_canvas,0.7,self.pre_canvas,0.3,0)


            for star in self.stars:
                new_canvas = self.draw_diag(new_canvas,star,np.random.random())

            Rmat = self.makeRmat(theta1)
            new_star = ((Rmat@self.moving_star_T_mat@self.moving_star.T).T).astype('int')
            new_canvas = self.draw_diag(new_canvas,new_star,color=(255,255,255))

            cv2.imshow("canvas",new_canvas)
            cv2.waitKeyEx(5)

            theta1+=5
            theta1%=360

            self.pre_canvas2 = self.pre_canvas1.copy() #잔상 보여준 캔버스 기준으로 한 칸 뒤로 민것 pre1 에 잇던거 2 로 원래 캔버스 위에서 copy 되면 서 만들어짐 imshow 가 최종
            self.pre_canvas1 = new_canvas.copy()
            self.pre_canvas = cv2.addWeighted(self.pre_canvas1,0.7,self.pre_canvas2,0.3,0) #잔상 1. 잔상 2. 비율









if __name__=="__main__":
    robot_arm = Night_Star(1000,1000,150,80) #가로길이
    robot_arm.main()
```



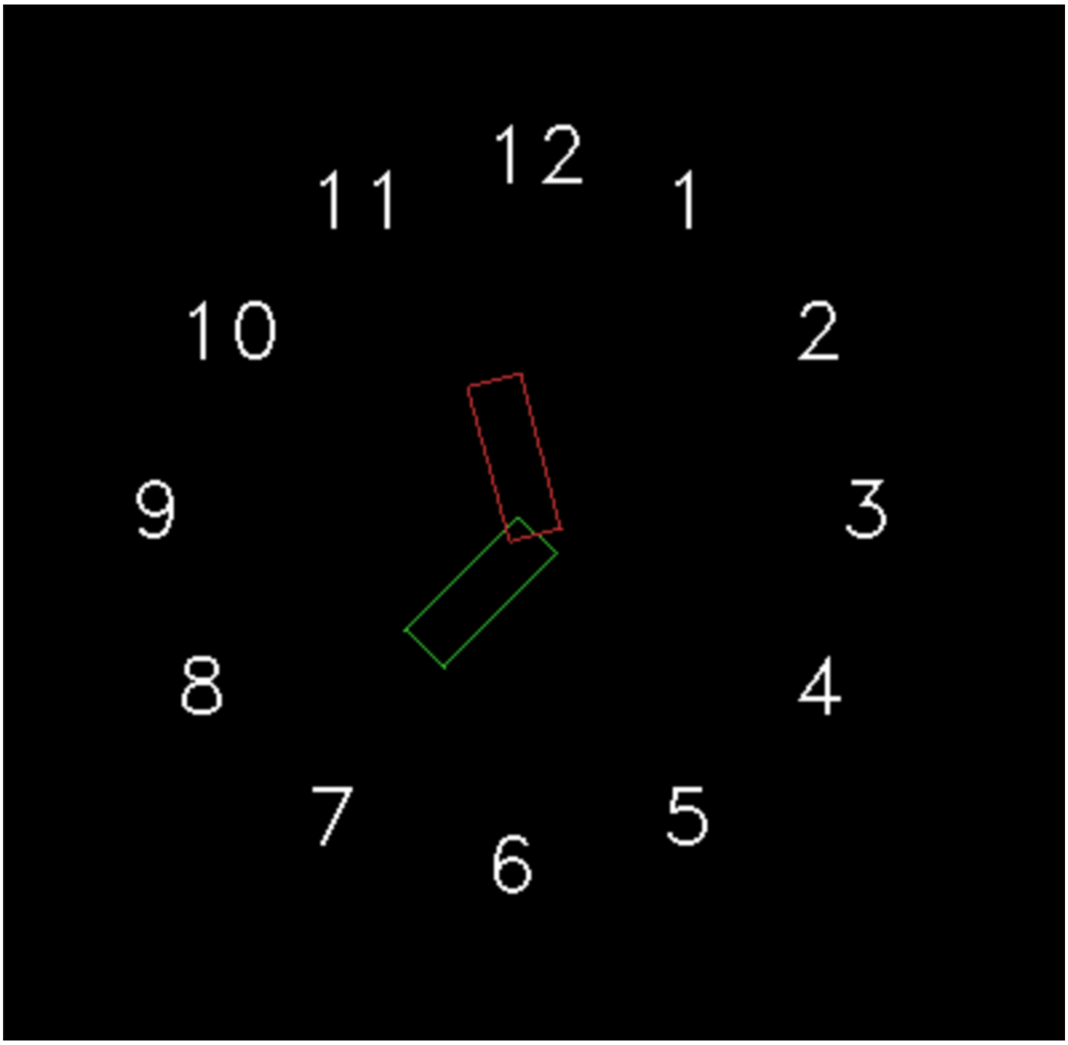
2. Clock

```
import cv2
import numpy as np

def deg2rad(degree):
    rad = degree * np.pi / 180.0
    return rad

def getRegularNgon(ngon):
        vertices = []
        delta = 360.0/ngon
        for i in range(ngon):
            degree = delta * i
            radian = deg2rad(degree)
            x = np.cos(radian)
            y = np.sin(radian)
            vertices.append((x,y))
        vertices = np.array(vertices)
        return vertices

def getline(canvas,x0,y0,x1,y1,color):
    # |기울기|<1 --> y =    (x-x0) * (y1-y0) / (x1-x0) +y0
    # |기울기|>1 --> x = (y-y0) * (x1-x0)/(y1-y0) +x0
    if(abs(x1-x0)<abs(y1-y0)): #|기울기|>1 인 경우
        if(y1==y0):
            y = y0
            if(x0<x1):
                for x in range(x0,x1+1):
                    canvas[int(y),int(x)] = color
            else:
                for x in range(x0,x1-1,-1):
                    canvas[int(y),int(x)] = color
        else:
            if(y0<y1):
                for y in range(y0,y1+1):
                    x = (y-y0) * (x1-x0)/(y1-y0) +x0
                    canvas[int(y),int(x)] = color
            else:
                for y in range(y0,y1-1,-1):
                    x = (y-y0) * (x1-x0)/(y1-y0) +x0
                    canvas[int(y),int(x)] = color

    else:#|기울기|<=1 인 경우
        if(x1==x0):
            x = x0
            if(y0<y1):
                for y in range(y0,y1+1):
                    canvas[int(y),int(x)] = color
            else:
                for y in range(y0,y1-1,-1):
                    canvas[int(y),int(x)] = color

        else:
            if(x0<x1):
                for x in range(x0,x1+1):
                    y =    (x-x0) * (y1-y0) / (x1-x0) +y0
                    canvas[int(y),int(x)] = color

            else:
                for x in range(x0,x1-1,-1):
                    y =    (x-x0) * (y1-y0) / (x1-x0) +y0
                    canvas[int(y),int(x)] = color

def getCenter(pts): #중심좌표를 불러오는 함수
    center = np.array([[0.0,0.0]])
    for p in pts:
        center+=p
```



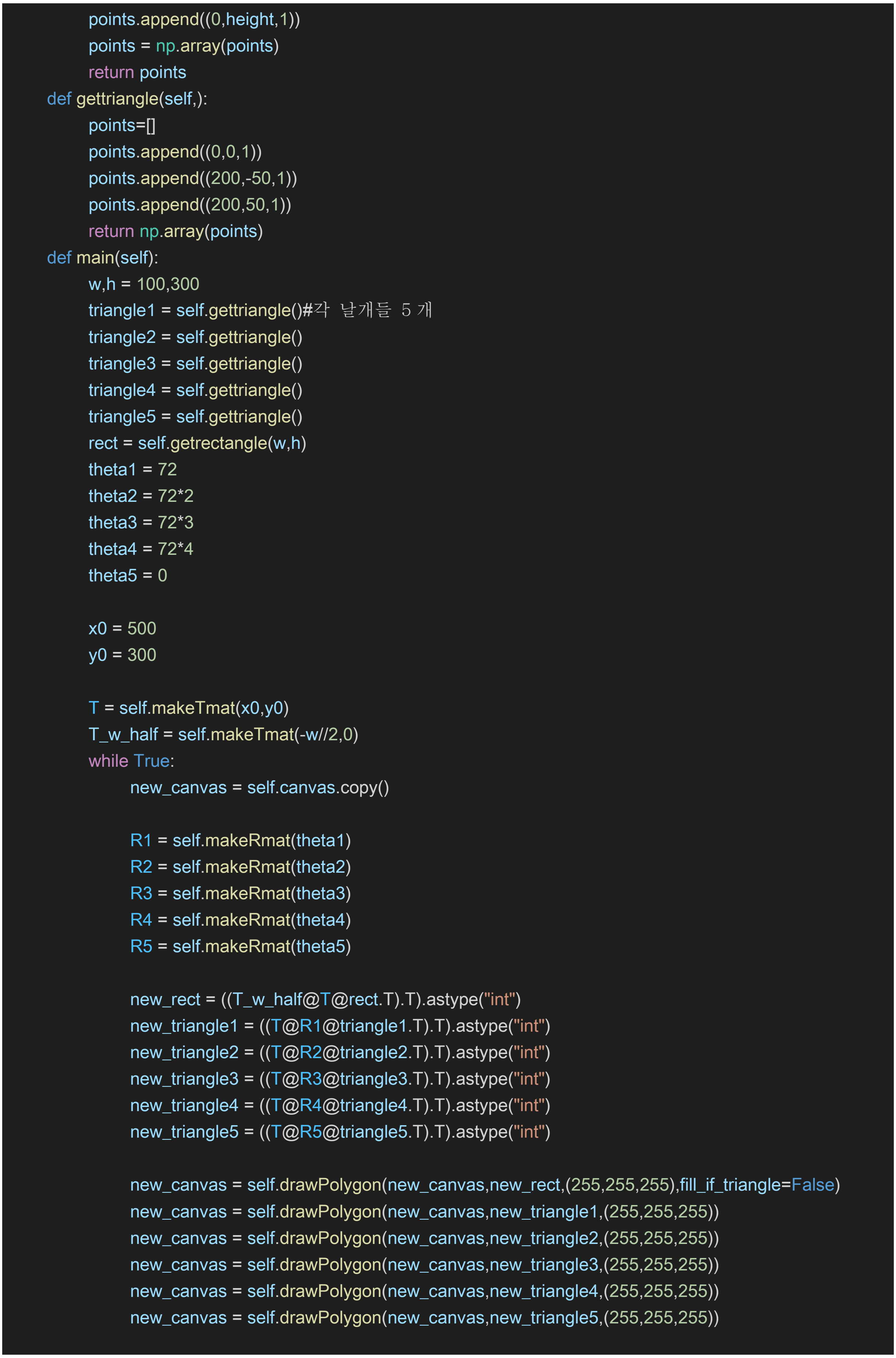
```
    center = center / pts.shape[0]
    center = center.astype('int')

    return center


def drawPolygon(canvas,pts,color):
    num = pts.shape[0]
    for i in range(num-1):
        getline(canvas,pts[i,0],pts[i,1],pts[i+1,0],pts[i+1,1],color)
    getline(canvas,pts[0,0],pts[0,1],pts[-1,0],pts[-1,1],color)


def makeTmat(a,b):
    T = np.eye(3,3)
    T[0,2] = a
    T[1,2] = b
    return T



def makeRmat(deg,ellipse=False):
    if ellipse==False:
        R = np.eye(3,3)
        radian = deg2rad(deg)
        c = np.cos(radian)
        s = np.sin(radian)
        R[0,0] = c
        R[0,1] = -s
        R[1,0] = s
        R[1,1] = c
        return R
    else:
        R = np.eye(3,3)
        radian = deg2rad(deg)
        c = np.cos(radian)
        s = np.sin(radian)
        R[0,0] = c
        R[0,1] = -s/2
        R[1,0] = s
        R[1,1] = c/2
        return R


def ctoc(canvas,SUN,p2):
    c1 = getCenter(SUN)
    c2 = getCenter(p2)

    getline(canvas, c1[0,0],c1[0,1], c2[0,0],c2[0,1], color=(255, 128, 128))
```

```
def draw_diag(canvas,polygon,axis=False):
    num = polygon.shape[0] #몇각형인지 //5 각형이면 5
    for i in range(num): #0-4
        for count in range(num):
            if(2<=abs(i-count)<num-1):
                getline(canvas,polygon[count,0],polygon[count,1],polygon[i,0],polygon[i,1],color=(255, 255, 255))
    if axis==True:
        Center = getCenter(polygon)
        getline(canvas, Center[0,0],Center[0,1], polygon[0,0],polygon[0,1], color=(255, 128, 128))

def draw_nums(canvas,): #300 부터 1 을 그린다
    h,w,_ = canvas.shape
    r = min(h//2,w//2)*2//3
    theta = 270
    for i in range(1,13):
        x,y = r*np.cos(deg2rad((theta+i*30)%360)),r*np.sin(deg2rad((theta+i*30)%360))
        x = x+(w//2)
        y = y+(h//2)
        canvas = cv2.putText(canvas,str(i),(int(x)-20,int(y)),cv2.FONT_HERSHEY_DUPLEX,1,(255,255,255))#-20 좌표를 중간으로 옮기기 위해
    return canvas

def main():
    width,height = 400,400
    center_x = width//2
    center_y = height//2
    w,h = 6*10,2*10
    rectangle = np.array([[0,0],[w,0],[w,h],[0,h]])
    base_canvas = draw_nums(np.zeros((height,width,3),dtype=np.uint8)) #캔버스 만들고 drawnums 때문에
    theta_hour = 90
    theta_minute = 270
    l = np.ones(4).reshape(4,1)
    while True:
        canvas = base_canvas.copy()
        rect_hour = rectangle.copy()
        rect_minute = rectangle.copy()
        rect_hour = np.concatenate([rect_hour,l],axis=1) #concatenate array2 개를 엮는다
        rect_minute = np.concatenate([rect_minute,l],axis=1)
        rect_hour = makeTmat(center_x,center_y) @ makeRmat(theta_hour) @ makeTmat(0,-h//2) @ rect_hour.T
        rect_minute = makeTmat(center_x,center_y) @ makeRmat(theta_minute) @ makeTmat(0,-h//2) @ rect_minute.T
        rect_hour = rect_hour[:2,:].T
        rect_minute = rect_minute[:2,:].T #1 을 빼는 것
        drawPolygon(canvas,rect_hour.astype('int'),(50,200,50))
        drawPolygon(canvas,rect_minute.astype('int'),(50,50,200))
        cv2.imshow("canvas",canvas)
        if cv2.waitKey(20) == 27:
            break
        theta_hour+=3*(15/360)
        theta_minute+=3

if __name__ =="__main__":
    main()
```

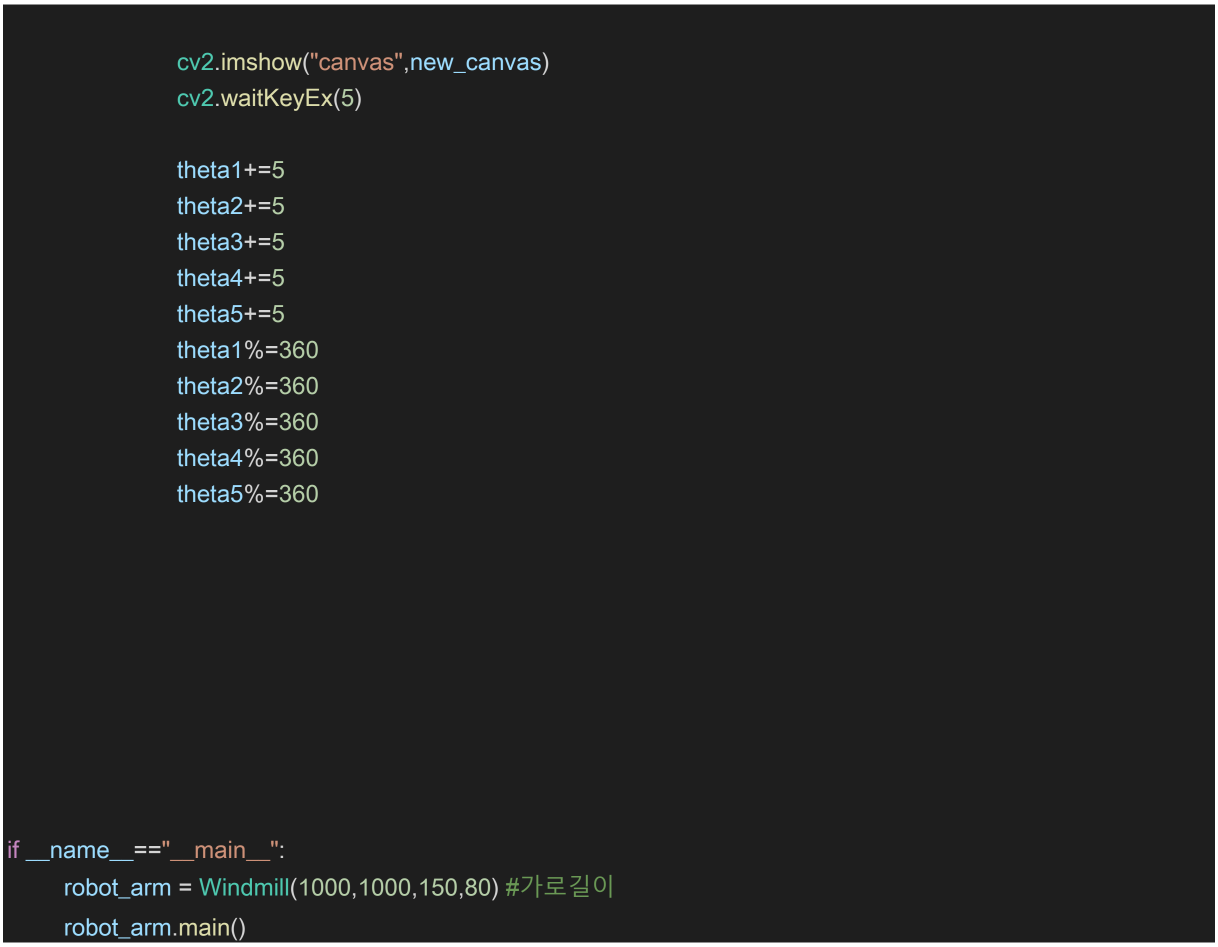


3. Windmill

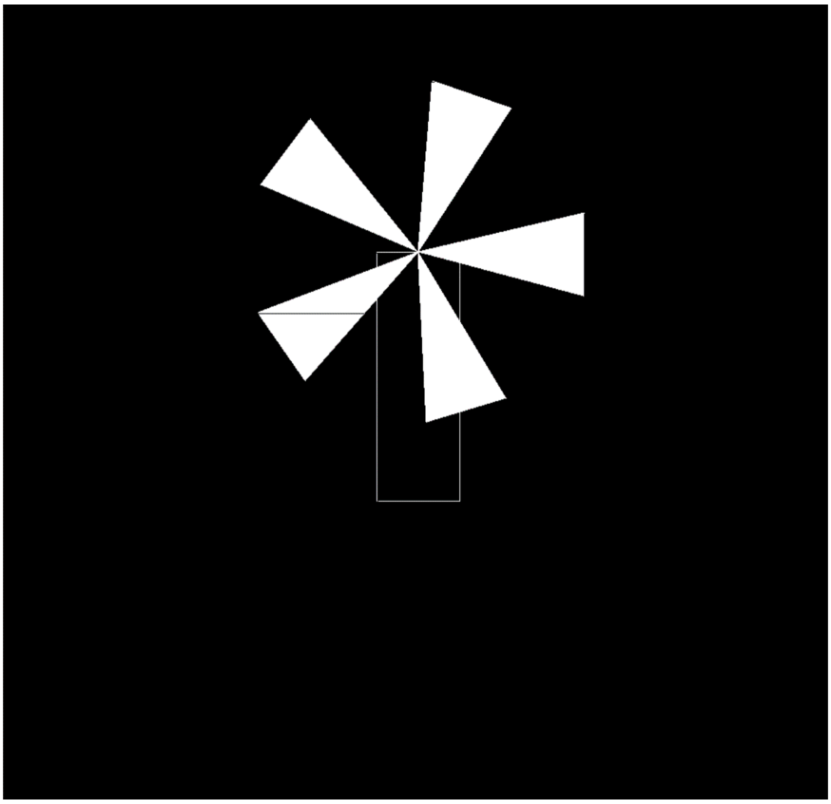
```
import numpy as np
import cv2


class Windmill(object):
    def __init__(self , canvas_width = 1500 , canvas_height=1000 , rect_w = 100 ,rect_h = 100): #가로  세로
        self.canvas_width = canvas_width
        self.canvas_height = canvas_height
        self.rect_width = rect_w
        self.rect_height = rect_h
        self.canvas = np.zeros((self.canvas_height,self.canvas_width,3),dtype=np.uint8)
        self.delta_theta = 15
    def getline(self,x0,y0,x1,y1):
        points = []
        if (x0==x1):
            x = x0
            if y0>y1:
                for y in range(y0,y1+1,-1):
                    points.append((x,y,1))
            else:
                for y in range(y0,y1+1):
                    points.append((x,y,1))
            # print(points)
        elif abs((y1-y0)/(x1-x0)) < 1:
            for x in range(x0,x1+1,1 if x0<x1 else -1):
                if x0 == x1:
                    y = y0
                else:
                    y = (x - x0) * (y1 - y0) / (x1 - x0) + y0
                yint = int(y)
                points.append((x,yint,1))
        else:
            for y in range(y0,y1+1,1 if y0<y1 else -1):
                if y0 == y1:
                    x = x0
                else:
                    x = (y      - y0) * (x1 - x0) / (y1 - y0) + x0
                xint = int(x)
                points.append((xint,y,1))
        return points
    def drawLine(self,canvas, x0, y0, x1, y1, color=(255, 255, 255)):
        new_canvas = canvas.copy()
        xys = self.getline(x0, y0, x1, y1)
        for xy in xys:
            x, y, a= xy
            new_canvas[y, x, :] = color
        return new_canvas
    def deg2rad(self,deg):
        rad = deg * np.pi / 180.
        return rad
    def drawPolygon(self,canvas, pts, color, fill_if_triangle=True):
        n = pts.shape[0]
        new_canvas = canvas.copy()
        points = []
        for k in range(n): #점 28 인이유 getline 함수때 모든 점의 값을 return 28 개는
            new_canvas = self.drawLine(new_canvas, pts[int(k%n),0], pts[int(k%n),1],
                                        pts[int((k+1)%n),0], pts[int((k+1)%n),1], color)
            points += self.getline(pts[int(k%n),0], pts[int(k%n),1], pts[int((k+1)%n),0], pts[int((k+1)%n),1])
        points = np.array(points)
        if fill_if_triangle:
            ys = points[:,1] #y 좌표 뽑은 이유 높이
            ymin = ys.min()
            ymax = ys.max()
            for y in range(ymin,ymax+1,1): #y 좌표 최대최소 뽑아서
                same_ys = [] #y 좌표가 같은 녀석들
                for p in points: #x 최소 최대 뽑기 y 좌표는 2 개가 나오는 게 맞다 x 좌표 전부 칠하기
x 최대최소 뽑으면 끝과 끝을 정하고 그 사이 것들을 채운다 가끔
                    if p[1]==y:
                        same_ys.append(p[0])
                xmin = min(same_ys)
                xmax = max(same_ys)
                for x in range(xmin,xmax+1,1):
                    new_canvas[y,x,:]=(255,255,255)
        return new_canvas


    def makeTmat(self,a,b):
        T = np.eye(3,3)
        T[0,2] = a
        T[1,2] = b
        return T
    def makeRmat(self,deg):
        rad = self.deg2rad(deg)
        c = np.cos(rad)
        s = np.sin(rad)
        R =   np.eye(3,3)
        R[0,0] = c
        R[0,1] = -s
        R[1,0] = s
        R[1,1] = c
        return R
    def getrectangle(self,width,height): #사각형 날개를 만드는 함수
        points = []
        points.append((0,0,1))
        points.append((width,0,1))
        points.append((width,height,1))
```



```
            points.append((0,height,1))
            points = np.array(points)
            return points
    def gettriangle(self,):
            points=[]
            points.append((0,0,1))
            points.append((200,-50,1))
            points.append((200,50,1))
            return np.array(points)
    def main(self):
            w,h = 100,300
            triangle1 = self.gettriangle()#각 날개들 5 개
            triangle2 = self.gettriangle()
            triangle3 = self.gettriangle()
            triangle4 = self.gettriangle()
            triangle5 = self.gettriangle()
            rect = self.getrectangle(w,h)
            theta1 = 72
            theta2 = 72*2
            theta3 = 72*3
            theta4 = 72*4
            theta5 = 0

            x0 = 500
            y0 = 300

            T = self.makeTmat(x0,y0)
            T_w_half = self.makeTmat(-w//2,0)
            while True:
                    new_canvas = self.canvas.copy()

                    R1 = self.makeRmat(theta1)
                    R2 = self.makeRmat(theta2)
                    R3 = self.makeRmat(theta3)
                    R4 = self.makeRmat(theta4)
                    R5 = self.makeRmat(theta5)

                    new_rect = ((T_w_half@T@rect.T).T).astype("int")
                    new_triangle1 = ((T@R1@triangle1.T).T).astype("int")
                    new_triangle2 = ((T@R2@triangle2.T).T).astype("int")
                    new_triangle3 = ((T@R3@triangle3.T).T).astype("int")
                    new_triangle4 = ((T@R4@triangle4.T).T).astype("int")
                    new_triangle5 = ((T@R5@triangle5.T).T).astype("int")

                    new_canvas = self.drawPolygon(new_canvas,new_rect,(255,255,255),fill_if_triangle=False)
                    new_canvas = self.drawPolygon(new_canvas,new_triangle1,(255,255,255))
                    new_canvas = self.drawPolygon(new_canvas,new_triangle2,(255,255,255))
                    new_canvas = self.drawPolygon(new_canvas,new_triangle3,(255,255,255))
                    new_canvas = self.drawPolygon(new_canvas,new_triangle4,(255,255,255))
                    new_canvas = self.drawPolygon(new_canvas,new_triangle5,(255,255,255))
```



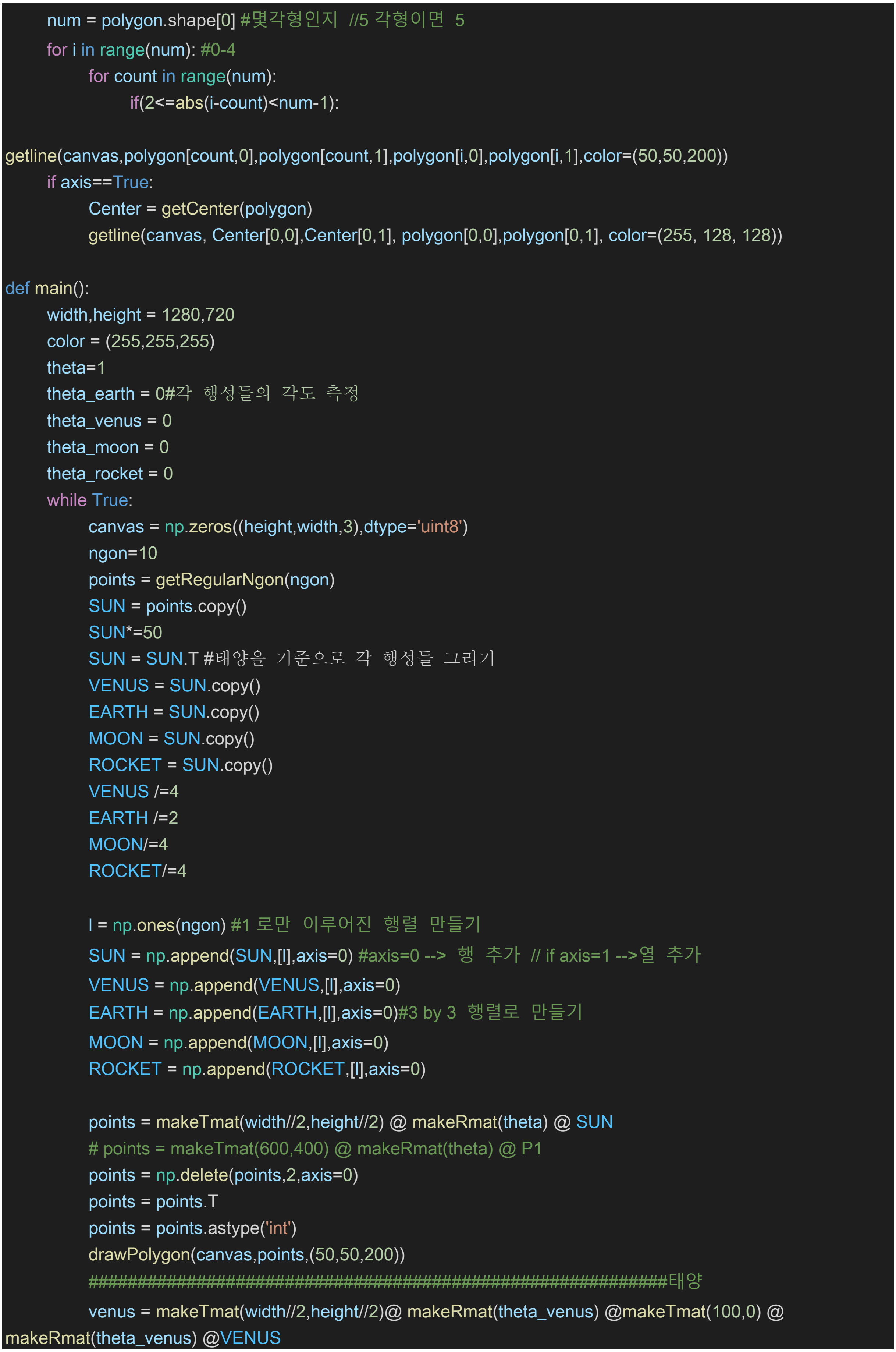
```
            cv2.imshow("canvas",new_canvas)
            cv2.waitKeyEx(5)

            theta1+=5
            theta2+=5
            theta3+=5
            theta4+=5
            theta5+=5
            theta1%=360
            theta2%=360
            theta3%=360
            theta4%=360
            theta5%=360









if __name__=="__main__":
    robot_arm = Windmill(1000,1000,150,80) #가로길이
    robot_arm.main()
```



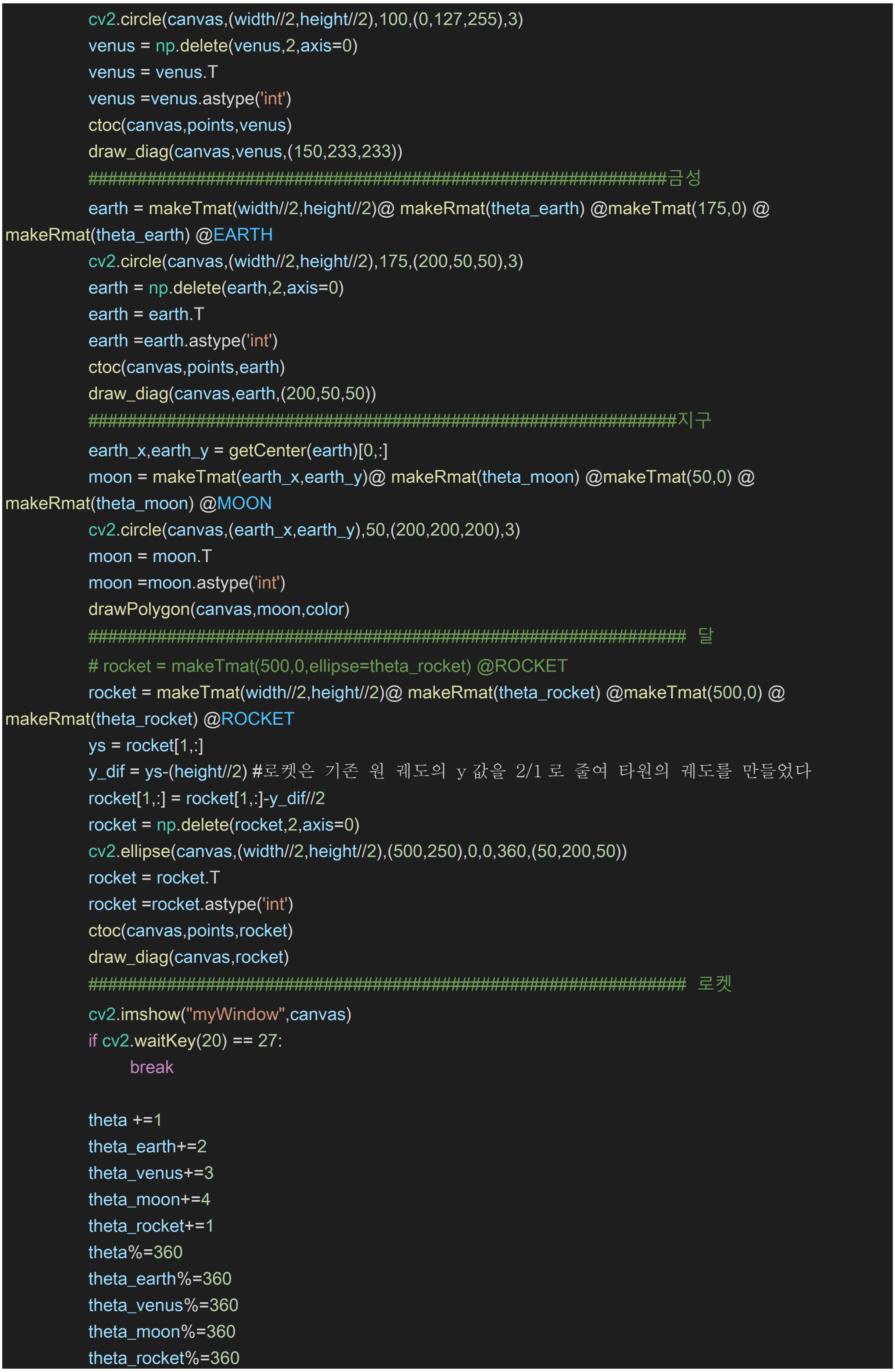
4. Solar System

```
##A solar system with the sun, venus, earth, moon, and a rocket wondering around.
import cv2
import numpy as np

def deg2rad(degree):
    rad = degree * np.pi / 180.0
    return rad

def getRegularNgon(ngon):
        vertices = []
        delta = 360.0/ngon
        for i in range(ngon):
            degree = delta * i
            radian = deg2rad(degree)
            x = np.cos(radian)
            y = np.sin(radian)
            vertices.append((x,y))

        vertices = np.array(vertices)
        return vertices

def getline(canvas,x0,y0,x1,y1,color):
    # |기울기|<1 --> y =    (x-x0) * (y1-y0) / (x1-x0) +y0
    # |기울기|>1 --> x = (y-y0) * (x1-x0)/(y1-y0) +x0
    if(abs(x1-x0)<abs(y1-y0)): #|기울기|>1 인  경우
        if(y1==y0):
            y = y0
            if(x0<x1):
                for x in range(x0,x1+1):
                    canvas[int(y),int(x)] = color
            else:
                for x in range(x0,x1-1,-1):
                    canvas[int(y),int(x)] = color
        else:
            if(y0<y1):
                for y in range(y0,y1+1):
                    x = (y-y0) * (x1-x0)/(y1-y0) +x0
                    canvas[int(y),int(x)] = color
            else:
                for y in range(y0,y1-1,-1):
                    x = (y-y0) * (x1-x0)/(y1-y0) +x0
                    canvas[int(y),int(x)] = color

    else:#|기울기|<=1 인  경우
        if(x1==x0):
            x = x0
            if(y0<y1):
                for y in range(y0,y1+1):
                    canvas[int(y),int(x)] = color
            else:
                for y in range(y0,y1-1,-1):
                    canvas[int(y),int(x)] = color

        else:
            if(x0<x1):
                for x in range(x0,x1+1):
                    y =    (x-x0) * (y1-y0) / (x1-x0) +y0
                    canvas[int(y),int(x)] = color

            else:
                for x in range(x0,x1-1,-1):
                    y =    (x-x0) * (y1-y0) / (x1-x0) +y0
                    canvas[int(y),int(x)] = color

def getCenter(pts):
    center = np.array([[0.0,0.0]])
    for p in pts:
        center+=p

    center = center / pts.shape[0]
    center = center.astype('int')

    return center


def drawPolygon(canvas,pts,color):
    num = pts.shape[0]
    for i in range(num-1):
        getline(canvas,pts[i,0],pts[i,1],pts[i+1,0],pts[i+1,1],color)
    getline(canvas,pts[0,0],pts[0,1],pts[-1,0],pts[-1,1],color)
```

```
def makeTmat(a,b,ellipse=None):
        if ellipse is None:
                T = np.eye(3,3)
                T[0,2] = a
                T[1,2] = b
                return T
        else:
                T = np.eye(3,3)
                r = (0.25*np.cos(2*deg2rad(ellipse)))+0.75
                if ellipse%90==0:
                        print("@@@@@@@@@@@@@@@@@@@@@@@@@@@@@")
                        print(1-r)
                        print(ellipse)
                        print("@@@@@@@@@@@@@@@@@@@@@@@@@@@@@")
                T[0,2] = a-a*(1-r)
                T[1,2] = b
                return T



def makeRmat(deg,ellipse=False):
        if ellipse==False:
                R = np.eye(3,3)
                radian = deg2rad(deg)
                c = np.cos(radian)
                s = np.sin(radian)
                R[0,0] = c
                R[0,1] = -s
                R[1,0] = s
                R[1,1] = c
                return R
        else:
                R = np.eye(3,3)
                radian = deg2rad(deg)
                c = np.cos(radian)
                s = np.sin(radian)
                R[0,0] = c
                R[0,1] = -s/2
                R[1,0] = s
                R[1,1] = c/2
                return R


def ctoc(canvas,SUN,p2):
        c1 = getCenter(SUN)
        c2 = getCenter(p2)

        getline(canvas, c1[0,0],c1[0,1], c2[0,0],c2[0,1], color=(255, 128, 128))

def draw_diag(canvas,polygon,axis=False):
```

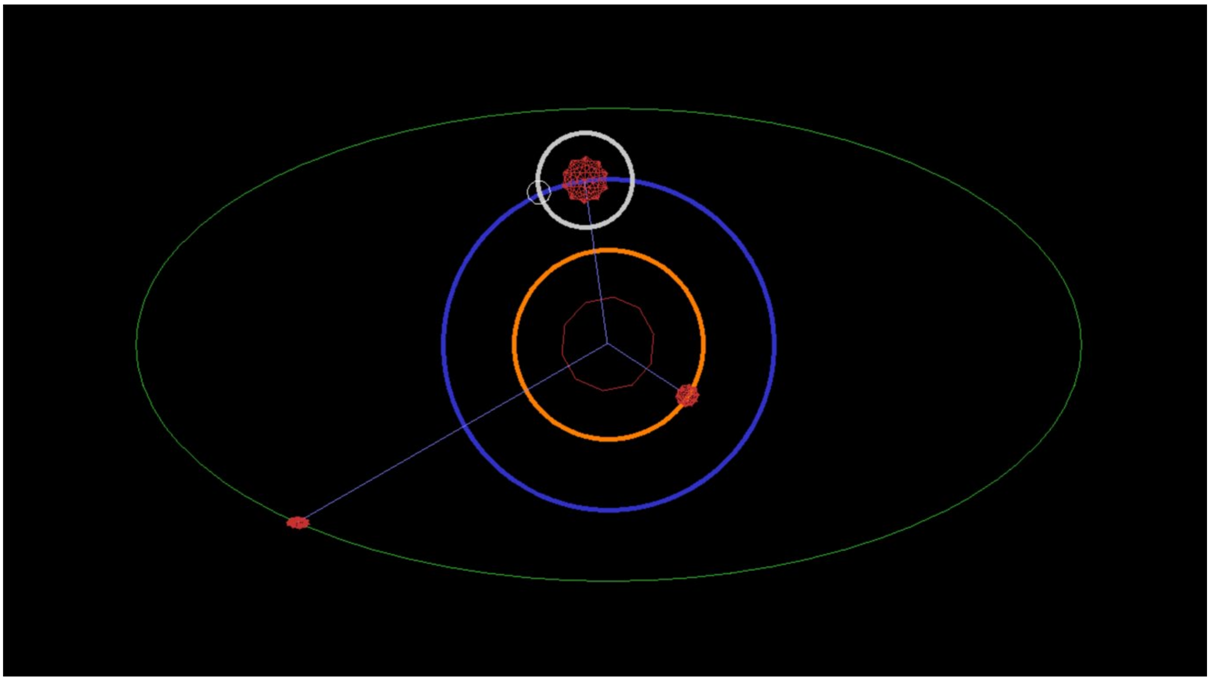


```
        num = polygon.shape[0] #몇각형인지  //5 각형이면  5
        for i in range(num): #0-4
                for count in range(num):
                        if(2<=abs(i-count)<num-1):

getline(canvas,polygon[count,0],polygon[count,1],polygon[i,0],polygon[i,1],color=(50,50,200))
        if axis==True:
                Center = getCenter(polygon)
                getline(canvas, Center[0,0],Center[0,1], polygon[0,0],polygon[0,1], color=(255, 128, 128))

def main():
        width,height = 1280,720
        color = (255,255,255)
        theta=1
        theta_earth = 0#각 행성들의 각도 측정
        theta_venus = 0
        theta_moon = 0
        theta_rocket = 0
        while True:
                canvas = np.zeros((height,width,3),dtype='uint8')
                ngon=10
                points = getRegularNgon(ngon)
                SUN = points.copy()
                SUN*=50
                SUN = SUN.T #태양을 기준으로 각 행성들 그리기
                VENUS = SUN.copy()
                EARTH = SUN.copy()
                MOON = SUN.copy()
                ROCKET = SUN.copy()
                VENUS /=4
                EARTH /=2
                MOON/=4
                ROCKET/=4

                I = np.ones(ngon) #1 로만 이루어진 행렬 만들기
                SUN = np.append(SUN,[I],axis=0) #axis=0 --> 행 추가 // if axis=1 -->열 추가
                VENUS = np.append(VENUS,[I],axis=0)
                EARTH = np.append(EARTH,[I],axis=0)#3 by 3 행렬로 만들기
                MOON = np.append(MOON,[I],axis=0)
                ROCKET = np.append(ROCKET,[I],axis=0)

                points = makeTmat(width//2,height//2) @ makeRmat(theta) @ SUN
                # points = makeTmat(600,400) @ makeRmat(theta) @ P1
                points = np.delete(points,2,axis=0)
                points = points.T
                points = points.astype('int')
                drawPolygon(canvas,points,(50,50,200))
                ############################################################태양
                venus = makeTmat(width//2,height//2)@ makeRmat(theta_venus) @makeTmat(100,0) @
makeRmat(theta_venus) @VENUS
```



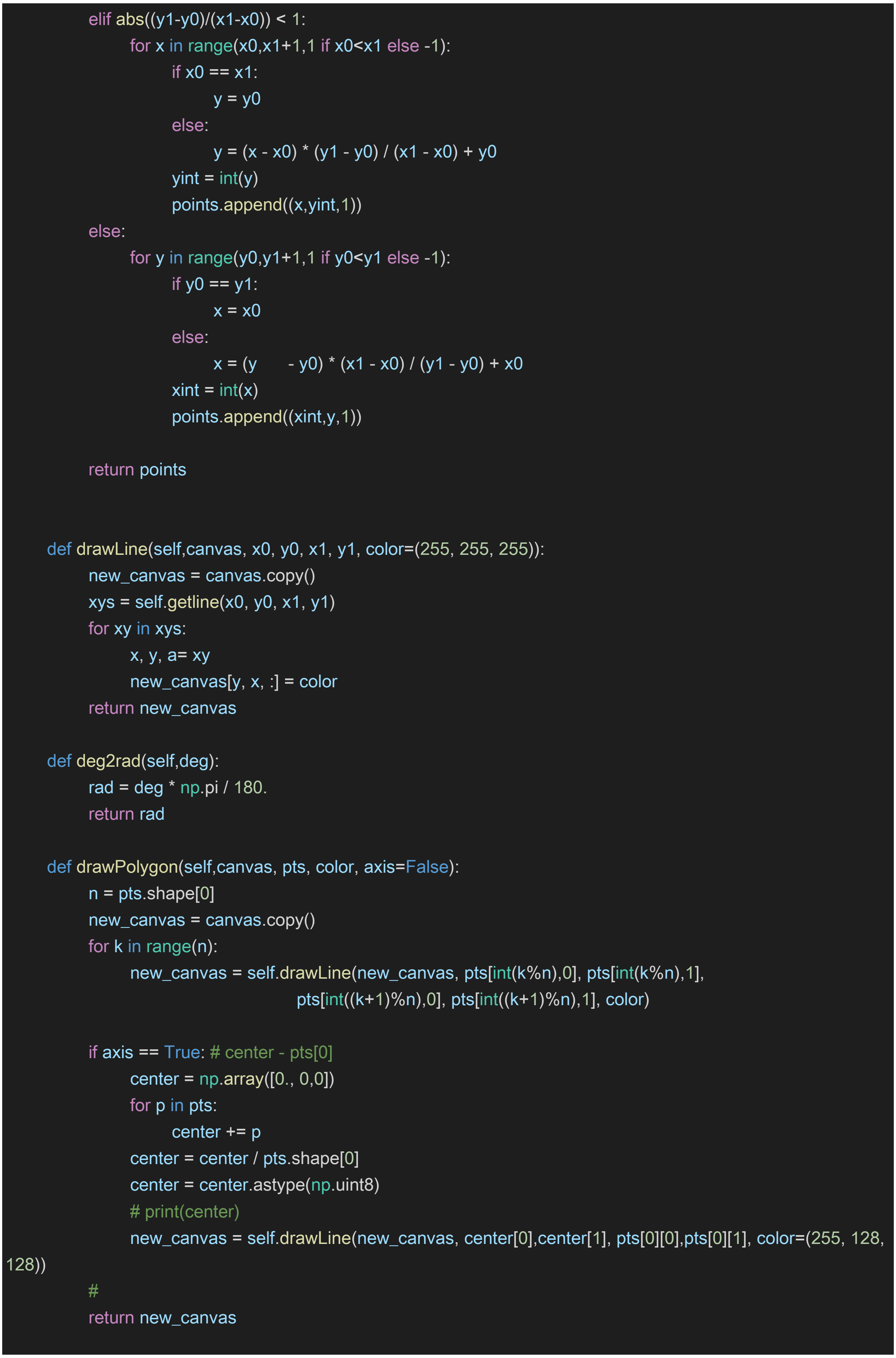
```
        cv2.circle(canvas,(width//2,height//2),100,(0,127,255),3)
        venus = np.delete(venus,2,axis=0)
        venus = venus.T
        venus =venus.astype('int')
        ctoc(canvas,points,venus)
        draw_diag(canvas,venus,(150,233,233))
        ##############################################################금성
        earth = makeTmat(width//2,height//2)@ makeRmat(theta_earth) @makeTmat(175,0) @
makeRmat(theta_earth) @EARTH
        cv2.circle(canvas,(width//2,height//2),175,(200,50,50),3)
        earth = np.delete(earth,2,axis=0)
        earth = earth.T
        earth =earth.astype('int')
        ctoc(canvas,points,earth)
        draw_diag(canvas,earth,(200,50,50))
        ###############################################################지구
        earth_x,earth_y = getCenter(earth)[0,:]
        moon = makeTmat(earth_x,earth_y)@ makeRmat(theta_moon) @makeTmat(50,0) @
makeRmat(theta_moon) @MOON
        cv2.circle(canvas,(earth_x,earth_y),50,(200,200,200),3)
        moon = moon.T
        moon =moon.astype('int')
        drawPolygon(canvas,moon,color)
        ################################################################ 달
        # rocket = makeTmat(500,0,ellipse=theta_rocket) @ROCKET
        rocket = makeTmat(width//2,height//2)@ makeRmat(theta_rocket) @makeTmat(500,0) @
makeRmat(theta_rocket) @ROCKET
        ys = rocket[1,:]
        y_dif = ys-(height//2) #로켓은 기존 원 궤도의 y 값을 2/1 로 줄여 타원의 궤도를 만들었다
        rocket[1,:] = rocket[1,:]-y_dif//2
        rocket = np.delete(rocket,2,axis=0)
        cv2.ellipse(canvas,(width//2,height//2),(500,250),0,0,360,(50,200,50))
        rocket = rocket.T
        rocket =rocket.astype('int')
        ctoc(canvas,points,rocket)
        draw_diag(canvas,rocket)
        ################################################################ 로켓
        cv2.imshow("myWindow",canvas)
        if cv2.waitKey(20) == 27:
            break

        theta +=1
        theta_earth+=2
        theta_venus+=3
        theta_moon+=4
        theta_rocket+=1
        theta%=360
        theta_earth%=360
        theta_venus%=360
        theta_moon%=360
        theta_rocket%=360
```

```
if __name__ =="__main__":
    main()
```



5. Robot Arms

```
import numpy as np
import cv2


class RobotArm(object):
    def __init__(self , canvas_width = 1500 , canvas_height=1000 , rect_w = 100 ,rect_h = 100): #가로  세로
        self.canvas_width = canvas_width
        self.canvas_height = canvas_height
        self.rect_width = rect_w
        self.rect_height = rect_h
        self.canvas = np.zeros((self.canvas_height,self.canvas_width,3),dtype=np.uint8)
        self.delta_theta = 15

    def getline(self,x0,y0,x1,y1):
        points = []
        if (x0==x1):
            x = x0
            if y0>y1:
                for y in range(y0,y1+1,-1):
                    points.append((x,y,1))
            else:
                for y in range(y0,y1+1):
                    points.append((x,y,1))
            # print(points)
```



```
        elif abs((y1-y0)/(x1-x0)) < 1:
            for x in range(x0,x1+1,1 if x0<x1 else -1):
                if x0 == x1:
                    y = y0
                else:
                    y = (x - x0) * (y1 - y0) / (x1 - x0) + y0
                yint = int(y)
                points.append((x,yint,1))
        else:
            for y in range(y0,y1+1,1 if y0<y1 else -1):
                if y0 == y1:
                    x = x0
                else:
                    x = (y      - y0) * (x1 - x0) / (y1 - y0) + x0
                xint = int(x)
                points.append((xint,y,1))

        return points


    def drawLine(self,canvas, x0, y0, x1, y1, color=(255, 255, 255)):
        new_canvas = canvas.copy()
        xys = self.getline(x0, y0, x1, y1)
        for xy in xys:
            x, y, a= xy
            new_canvas[y, x, :] = color
        return new_canvas

    def deg2rad(self,deg):
        rad = deg * np.pi / 180.
        return rad

    def drawPolygon(self,canvas, pts, color, axis=False):
        n = pts.shape[0]
        new_canvas = canvas.copy()
        for k in range(n):
            new_canvas = self.drawLine(new_canvas, pts[int(k%n),0], pts[int(k%n),1],
                                        pts[int((k+1)%n),0], pts[int((k+1)%n),1], color)

        if axis == True: # center - pts[0]
            center = np.array([0., 0,0])
            for p in pts:
                center += p
            center = center / pts.shape[0]
            center = center.astype(np.uint8)
            # print(center)
            new_canvas = self.drawLine(new_canvas, center[0],center[1], pts[0][0],pts[0][1], color=(255, 128,
128))
        #
        return new_canvas
    def makeTmat(self,a,b):
        T = np.eye(3,3)
        T[0,2] = a
        T[1,2] = b
        return T

    def makeRmat(self,deg):
        rad = self.deg2rad(deg)
        c = np.cos(rad)
        s = np.sin(rad)
        R =    np.eye(3,3)
        R[0,0] = c
        R[0,1] = -s
        R[1,0] = s
        R[1,1] = c
        return R

    def getrectangle(self,width,height): #robot arm 의 뼈대를 만드는 함수
        points = []
        points.append((0,0,1))
        points.append((width,0,1))
        points.append((width,height,1))
        points.append((0,height,1))

        points = np.array(points)
        return points


    def main(self):
        theta1 = -90
        theta2 = (theta1+self.delta_theta)%360
        theta3 = (theta2+self.delta_theta)%360
        theta4 = (theta3+self.delta_theta)%360
        theta5 = (theta4+self.delta_theta)%360


        while True:
            new_canvas = self.canvas.copy() #4 개의 robotarm 을 만든다
            rect1 = self.getrectangle(self.rect_width,self.rect_height)
            rect2 = self.getrectangle(self.rect_width,self.rect_height)
            rect3 = self.getrectangle(self.rect_width,self.rect_height)
            rect4 = self.getrectangle(self.rect_width,self.rect_height)
            rect5 = self.getrectangle(self.rect_width,self.rect_height)

            H1 = self.makeTmat(self.canvas_width//3,self.canvas_height*2//3) @ self.makeRmat(theta1) @
self.makeTmat(0,-self.rect_height//2)#각 팔의 높이 조절
            H2 = self.makeTmat(self.rect_width,0) @ self.makeTmat(0,self.rect_height//2) @
self.makeRmat(theta2) @ self.makeTmat(0,-self.rect_height//2)
            H3 = self.makeTmat(self.rect_width,0) @ self.makeTmat(0,self.rect_height//2) @
self.makeRmat(theta3) @ self.makeTmat(0,-self.rect_height//2)
```

```python
            H4 = self.makeTmat(self.rect_width,0) @ self.makeTmat(0,self.rect_height//2) @ self.makeRmat(theta4) @ self.makeTmat(0,-self.rect_height//2)
            H5 = self.makeTmat(self.rect_width,0) @ self.makeTmat(0,self.rect_height//2) @ self.makeRmat(theta5) @ self.makeTmat(0,-self.rect_height//2)

            rect1 = H1 @ rect1.T
            rect1 = rect1.T
            rect1 = rect1.astype("int")
            new_canvas = self.drawPolygon(new_canvas,rect1,(255,255,255),axis = False)


            rect2 = H1 @ H2 @ rect2.T
            rect2 = rect2.T
            rect2 = rect2.astype("int")
            new_canvas = self.drawPolygon(new_canvas,rect2,(255,255,255),axis = False)


            rect3 = H1 @ H2 @ H3 @ rect3.T
            rect3 = rect3.T
            rect3 = rect3.astype("int")
            new_canvas = self.drawPolygon(new_canvas,rect3,(255,255,255),axis = False)


            rect4 = H1 @ H2 @ H3 @ H4 @ rect4.T
            rect4 = rect4.T
            rect4 = rect4.astype("int")
            new_canvas = self.drawPolygon(new_canvas,rect4,(255,255,255),axis = False)


            rect5 = H1 @ H2 @ H3 @ H4 @ H5 @ rect5.T
            rect5 = rect5.T
            rect5 = rect5.astype("int")
            new_canvas = self.drawPolygon(new_canvas,rect5,(255,255,255),axis = False)




            theta2 += 1
            theta3 += 1
            theta4 += 1
            cv2.imshow("canvas",new_canvas)
            cv2.waitKeyEx(30)





if __name__=="__main__":
    robot_arm = RobotArm(1000,1000,150,80) #가로길이
    robot_arm.main()
```