Circle Shooter

```
import random

from OpenGL.GL import *

from OpenGL.GLUT import *

from OpenGL.GLU import *


w_width,w_height=600,900

shooter_center=w_width//2

y_from_below=20

bullet_speed=5

shooter_radius=15

health=3

miss_fire=0

pausestate=0

game_pause=False

game_over=False

game_restarted=False

fired=False

target_storer=[]

bullet_tracker=[]

game_points=0

targetspeed=0.75


############################ Logic ############################

def draw_points(x,y):

   glPointSize(3)

   glBegin(GL_POINTS)

   glVertex2f(x,y)

   glEnd()
```

```
def draw_points_allzones(xc,yc,a,b):
    draw_points(xc+a,yc+b)
    draw_points(xc-a,yc+b)
    draw_points(xc+a,yc-b)
    draw_points(xc-a,yc-b)
    draw_points(xc+b,yc+a)
    draw_points(xc-b,yc+a)
    draw_points(xc+b,yc-a)
    draw_points(xc-b,yc-a)
def midpoint_circle(xc, yc, radius):
    center = 0
    y = radius
    d = 1 - radius
    draw_points_allzones(xc, yc, center, y)

    while center < y:
        center += 1

        if d < 0:
            d += (2 * center) + 3
        else:
            y -= 1
            d += (2 * center) - (2 * radius) + 5

        draw_points_allzones(xc, yc, center, y)


def zone_finder(a,b,x,y): #a,b is the initial and x,y are the final points
```

```
    dx=x-a
    dy=y-b
    zone=0
    if abs(dx)>=abs(dy):
        if dx>0 and dy>0:
            zone=0
        elif dx>0 and dy<0:
            zone=7
        elif dx<0 and dy>0:
            zone=3
        elif dx<0 and dy<0:
            zone=4
    else:
        if dx>0 and dy>0:
            zone=1
        elif dx<0 and dy>0:
            zone=2
        elif dx<0 and dy<0:
            zone=5
        else:
            zone=6
    return zone

def original_to_convert(o_z,x,y):
    if o_z==0:
        return x,y
    elif o_z==1:
        return y,x
    elif o_z==2:
        return y,-x
    elif o_z==3:
        return -x,y
    elif o_z==4:
        return -x,-y
    elif o_z==5:
        return -y,-x
    elif o_z==6:
        return -y,x
    elif o_z==7:
        return x,-y

def from_convert_to_original(o_z,x,y):
    if o_z==0:
        return x,y
    elif o_z==1:
        return y,x
    elif o_z==2:
        return -y,x
    elif o_z==3:
        return -x,y
    elif o_z==4:
        return -x,-y
    elif o_z==5:
        return -y,-x
    elif o_z==6:
        return y,-x
    elif o_z==7:
        return x,-y
```

```
def midpoint_line(zone,a,b,x,y):
    dx=x-a
    dy=y-b
    d=(2*dy)-dx

    east=2*dy
    n_east=2*(dy-dx)

    while a<x:
        temp_x,temp_y=from_convert_to_original(zone,a,b)
        draw_points(temp_x,temp_y)

        if d<=0:
            d+=east
            a+=1
        else:
            d+=n_east
            a+=1
            b+=1

def eight_way(a,b,x,y):
    zone=zone_finder(a,b,x,y)
    temp_a,temp_b=original_to_convert(zone,a,b)
    temp_x,temp_y=original_to_convert(zone,x,y)
    midpoint_line(zone,temp_a,temp_b,temp_x,temp_y)

def shooter():
    global shooter_center,shooter_radius,y_from_below
    glColor3f(1.0,1.0,0.0)
    midpoint_circle(shooter_center,y_from_below,shooter_radius)

def bullets():
    global bullet_tracker
    for i in bullet_tracker:
        midpoint_circle(i["x"],i["y"],3)      #[{'x': 300 value of x  coordinate, 'y': 520 values of y coordinate}]

def target_generator():
    global target_storer

    while len(target_storer)<5:
        collison=False
        x=random.randint(30,570)
        y=random.randint(800,900)
        r=random.randint(10,30)
        for i in target_storer:
            x_dis=i["x"]-x
            y_dis=i["y"]-y
            total_radius=(x_dis**2 + y_dis**2)**0.5 #r**2=x**2+y**2
            if total_radius<=(i["r"]+r):
                collison=True
                break
        if collison==False:
            target_storer.append({'x': x, 'y': y, 'r': r})

def target_position_updater(): #makes the target move down gradually
    global health,bullet,game_over,game_pause,targetspeed
    for i in target_storer:
        i["y"]-=targetspeed

        if i["y"]<=0:
            target_storer.remove(i)
            health-=1
            print(f"Life lost! Remaining lives: {health}")

            if health<=0:
                game_over=True
                game_pause= True
                print(f"You missed 3 Targets || Game Over || Total Points: {game_points}")

def collision_detector():
    global
game_points,target_storer,bullet_tracker,shooter_center,shooter_radius,game_over,game_pause

    for i in target_storer:
        total_d=((shooter_center-i["x"])**2 + (shooter_radius-i["y"])**2)**0.5
        if total_d<=(shooter_radius+i["r"]):
            target_storer.clear()
            game_pause=True
            game_over=True
            print(f"You lost the bubbles colided with the shooter! \n|| Game Over || Total Points:
{game_points}")
            return

    #bullet collison check
    for j in bullet_tracker:
        for k in target_storer:
            total_d=((j["x"]-k["x"])**2 + (j["y"]-k["y"])**2)**0.5
            if total_d<= k["r"]:
                bullet_tracker.remove(j)
                target_storer.remove(k)
                game_points+=1
                print(f"Target Down ! || Current score: {game_points}")
                break


def bullet_position_updater():
    global bullet_tracker,miss_fire,bullet_speed,game_pause,game_over


    for i in bullet_tracker:
        i["y"]+=bullet_speed


        if i["y"]>=900:
            miss_fire+=1
            bullet_tracker.remove(i)
            print(f"Missed Fire! Remaining Missed Fires: {3-miss_fire}")
        if miss_fire>=3:
            game_over=True
            game_pause=True
            print(f"Missed 3 Fires || Game Over || Total Points: {game_points}")
            break


############################ Hardware Section ############################
def back_arrow():
    glColor3f(0.0,1.0,1.0)
    eight_way(25,855,105,855)
    eight_way(45,860,25,855)
    eight_way(45,850,25,855)

def cross():
    glColor3f(1.0,0.0,0.0)
    eight_way(500,845,580,885)
    eight_way(500,885,580,845)

def pause():
    glColor3f(1.0,1.0,0.0)
    eight_way(298,875,298,825)
    eight_way(298,875,315,850)
    eight_way(298,825,315,850)

def play():
    glColor3f(1.0,1.0,0.0)
    eight_way(298,875,298,825)
    eight_way(305,875,305,825)

def mouseListener(button,state,x,y):
    global game_pause,game_restarted,game_over,pausestate
    y_point=w_height-y
    if (button==GLUT_LEFT_BUTTON) and (state==GLUT_DOWN):

        if (21<=x<=105) and (850<=y_point<=860):
            game_restarted=True
            print("Restarting Game...")
        if(295<=x<=315) and (815<=y_point<=875):
            if game_over==False:
                pausestate+=1
                if pausestate%2 !=0:
                    game_pause = True
                    print(f"Game Paused || Current points: {game_points}")
                elif pausestate%2 ==0:
                    game_pause = False
                    print('Game Resumed --|>')

        if (495<=x<=580) and (845<=y_point<=885):
            print(f"Goodbye!\nYour Total Points are: {game_points}")
            glutLeaveMainLoop()

def keyboardlistner(key,x,y):
    global shooter_center,shooter_radius,fired,shooter_center,game_over,shooter_y
    if game_over== False:
        xl=shooter_center-shooter_radius
        xr=shooter_center+shooter_radius

        if game_pause==False:
            if key == b'a':
                if (xl>0):
                    shooter_center-=bullet_speed
            if key == b'd':
                if (xr<600):
                    shooter_center+=bullet_speed

            if key == b' ':
                fired=True
                shooter_center = shooter_center
                shooter_y = y_from_below+shooter_radius+5
                bullet_tracker.append({'x': shooter_center, 'y': shooter_y})

    glutPostRedisplay()

############################ Driver code + The Engine ############################
def init():
    glViewport(0,0,w_width,w_height)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    glOrtho(0.0,600.0,0.0,900.0,0.0,1.0)
    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()

def display():
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glLoadIdentity()
    init()
    glColor3f(1.0, 1.0, 1.0)
    back_arrow()
    cross()
    shooter()
    if  game_over==False:
        bullets()
        for i in target_storer:
            glColor3f(1.0,1.0,0.0)
            midpoint_circle(i['x'], i['y'], i['r'])
    if game_pause==True:
        pause()
    if game_pause==False:
        play()
    glutSwapBuffers()



def game_engine(value):
    global
health,miss_fire,pausestate,game_pause,game_over,game_restarted,fired,game_points,shooter_center,
shooter_radius


    if game_restarted==True:
        game_points=0
        health=3
        pausestate=0
        miss_fire=0
        shooter_center=w_width//2
        y_from_below=20
        shooter_y= 20
        shooter_radius=20
        game_over=False
        game_pause=False
        game_restarted=False
        fired=False
        target_storer.clear()
        bullet_tracker.clear()
        bullets()
    if game_over==True or game_pause==True:
        glutPostRedisplay()
        glutTimerFunc(10, game_engine, 0)
        return

    if game_over==False:
        target_position_updater()
        bullet_position_updater()
        collision_detector()

        if len(target_storer)<5:
            target_generator()

    glutPostRedisplay()
    glutTimerFunc(10, game_engine, 0)

print("Welcome to Bubble shooter Enjoy!!")
glutInit()
glutInitDisplayMode(GLUT_RGBA)
glutInitWindowSize(w_width,w_height)
glutInitWindowPosition(700,0)
glutCreateWindow(b"Circle Shooters!")
glutDisplayFunc(display)
glutMouseFunc(mouseListener)
glutKeyboardFunc(keyboardlistner)
glutTimerFunc(10, game_engine, 0)
glutMainLoop()
```