```
# pong.py

import pygame, random, time
from pygame.locals import *
from pygame.font import *

# some colors
BLACK = (   0,   0,   0)
WHITE = ( 255, 255, 255)
RED   = ( 255,   0,   0)
GREEN = (   0, 255,   0)
BLUE  = ( 0,   0,   255)

WALL_SIZE = 10
STEP = 0.5
PADDLE_STEP = 22
LEFT  = 0
RIGHT = 1

WINNING_SCORE = 8 # (1) set a new winning score


class BlockSprite(pygame.sprite.Sprite):

    def __init__(self, x, y, width, height, color=BLACK):
        super().__init__()
        self.image = pygame.Surface((width, height))
        self.image.fill(color)
        self.rect = self.image.get_rect()
        self.rect.topleft = (x, y)

# -----------------------------------------------------------

class Paddle(BlockSprite):
    def __init__(self, x, y):
        super().__init__(x, 270, 10, 135, BLUE)  # paddle width & height

    def update(self, x, y):
        self.image = pygame.Surface((10, y))
        self.image.fill(BLUE)
        super().__init__(x, y, 10, y, BLUE)
```

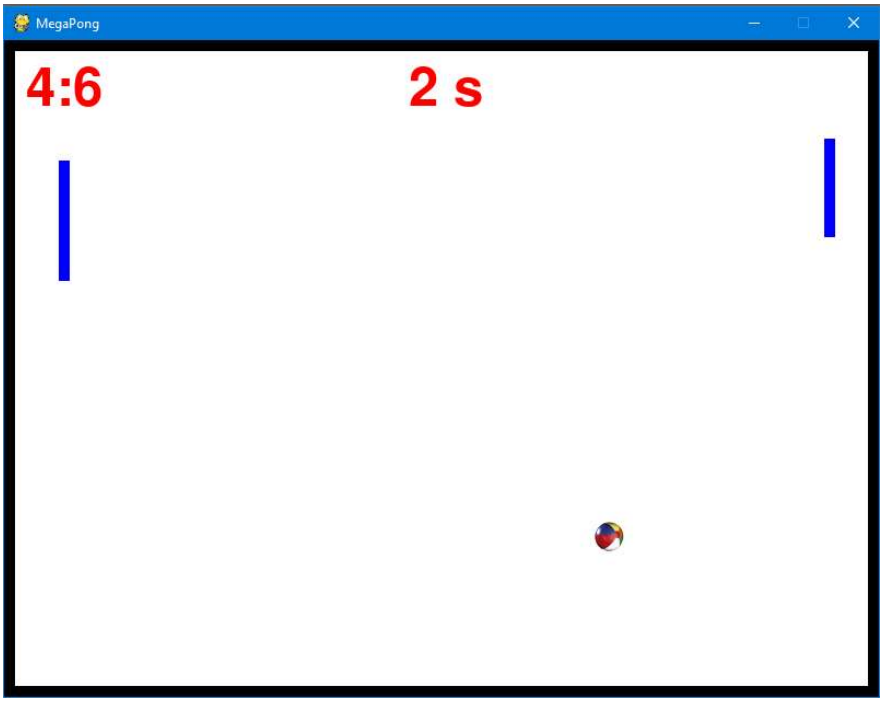
```
    def move(self, step):
        if pygame.sprite.collide_rect(self, top) and (step < 0):  # at top &
going up
            step = 0
        elif pygame.sprite.collide_rect(self, bottom) and (step > 0):
            # at bottom and going down
            step = 0
        self.rect.y += step

# ------------------------------------------------------

class BallSprite(pygame.sprite.Sprite):
    def __init__(self, fnm):
        super().__init__()
        self.image = pygame.image.load(fnm).convert_alpha()
        self.rect = self.image.get_rect()
        self.rect.center = [scrWidth/2, scrHeight/2]
            # start position of the ball in center of window
        self.speed = 0.5
        self.xStep, self.yStep = self.randomSteps()
            # step size and direction along each axis

    def update(self):
        global scoreLeft, scoreRight

        if pygame.sprite.collide_rect(self, leftPaddle) and (self.xStep < 0):
            # hit left paddle and going left
            self.xStep = -self.xStep    # change direction

        elif pygame.sprite.collide_rect(self, rightPaddle) and (self.xStep >
0):
            # hit right paddle and going right
            self.xStep = -self.xStep    # change direction

        if pygame.sprite.spritecollideany(self, horizWalls):
            # change y-step direction at top and bottom sides
            self.yStep = -self.yStep

        if pygame.sprite.spritecollideany(self, vertWalls):
            # ball has reached left or right sides
            if pygame.sprite.collide_rect(self, right):
                scoreLeft += 1
            else:   # left side
                scoreRight += 1

            # reset the ball
            self.rect.center = (scrWidth/2, scrHeight/2)
            self.xStep, self.yStep = self.randomSteps()
```

```
        # (2) set a new ball speed

        self.rect.x += self.speed * self.xStep
        self.rect.y += self.speed * self.yStep

    def randomSteps(self):
        # create a random +/- STEP pair
        x = STEP
        if random.random() > 0.5:
            x = -x
        y = STEP
        if random.random() > 0.5:
            y = -y
        return [x,y]

# -----------------------------------

def centerImage(screen, im):
    x = (scrWidth - im.get_width())/2
    y = (scrHeight - im.get_height())/2
    screen.blit(im, (x,y))

# ---------- main -------------

pygame.init()
screen = pygame.display.set_mode([800,600]) # (1) set a new display size
screen.fill(WHITE)
pygame.display.set_caption("MegaPong")

scrWidth, scrHeight = screen.get_size()

#create time variable
clock = pygame.time.Clock()
playTime = 0.0

# create wall sprites
top    = BlockSprite(0, 0, scrWidth, WALL_SIZE)
bottom = BlockSprite(0, scrHeight-WALL_SIZE, scrWidth, WALL_SIZE)
left   = BlockSprite(0, 0, WALL_SIZE, scrHeight)
right  = BlockSprite(scrWidth-WALL_SIZE, 0, WALL_SIZE, scrHeight)

horizWalls = pygame.sprite.Group(top, bottom)
vertWalls = pygame.sprite.Group(left, right)

# create two paddles
leftPaddle = Paddle(50, scrHeight/2)
rightPaddle = Paddle(scrWidth-50, scrHeight/2)
```

```
ball = BallSprite('smallBall.png')
ball2 = BallSprite('smallBall.png')
ball3 = BallSprite('smallBall.png')

sprites = pygame.sprite.OrderedUpdates(top, bottom, left, right,
                                leftPaddle, rightPaddle, ball)
sprites2 = pygame.sprite.OrderedUpdates(top, bottom, left, right,
                                leftPaddle, rightPaddle, ball2)
sprites3 = pygame.sprite.OrderedUpdates(top, bottom, left, right,
                                leftPaddle, rightPaddle, ball3)


# game vars
leftStep = 0; rightStep = 0
  # move step in pixels for paddles
scoreLeft = 0; scoreRight = 0
winMsg = ""
gameOver = False

#decreases paddles
oldLeft = 0; oldRight = 0
hight_L = 150; hight_R = 150

# font = pygame.font.Font(None, 30)
font = pygame.font.Font(None, 72)

running = True
while running:
    ms = clock.tick(30)
    playTime += ms/1000.0

    # ---------------- Speed ball ------------------- # (2)
    if playTime <= 14:
        ball.speed += 0.08
        ball2.speed += 0.08
        ball3.speed += 0.08
    elif playTime > 14:
        ball.speed += 0.08
        ball2.speed += 0.08
        ball3.speed += 0.08
    #set a speed ball with define time
    # handle events
    for event in pygame.event.get():
        if event.type == QUIT:
            running = False
        if (event.type == KEYUP and event.key == K_ESCAPE):
            running = False
```

```
    if event.type == KEYDOWN:
        if event.key == K_q:   # left paddle
            leftStep = -PADDLE_STEP    # up
        elif event.key == K_s:
            leftStep = PADDLE_STEP     # down

        if event.key == K_p:   # right paddle
            rightStep = -PADDLE_STEP   # up
        elif event.key == K_l:
            rightStep = PADDLE_STEP    # down

    elif event.type == KEYUP:
        if event.key == K_q or event.key == K_s:   # left paddle
            leftStep = 0
        if event.key == K_p or event.key == K_l:   # right paddle
            rightStep = 0


# update game
if not gameOver:
    leftPaddle.move(leftStep)
    rightPaddle.move(rightStep)
    ball.update()

    if scoreLeft > oldLeft:
        playTime = 0;
        hight_L -= 10 # (3) reduce a left paddle size
        leftPaddle.update(50,hight_L)
        oldLeft += 1
        ball.speed = 1
        ball2.speed = 1
        ball3.speed = 1
        ball.rect.center = [scrWidth/2, scrHeight/2]
        ball.update();

    elif scoreRight > oldRight:
        playTime = 0;
        hight_R -= 10 # (3) reduce a right paddle size
        rightPaddle.update(scrWidth-50,hight_R)
        oldRight += 1
        ball.speed = 1
        ball2.speed = 1
        ball3.speed = 1
        ball.rect.center = [scrWidth/2, scrHeight/2]
        ball.update();
# -----------------------------------
```

```
        if scoreLeft >= WINNING_SCORE:
            winMsg = "Left Wins!"
            gameOver = True
        elif scoreRight >= WINNING_SCORE:
            winMsg = "Right Wins!"
            gameOver = True


    # redraw
    screen.fill(WHITE)
    sprites.draw(screen);
    ball.update();
    if playTime >= 4:
        sprites2.draw(screen); # (4) display a second ball
        ball2.update();
    if playTime >= 9:
        sprites3.draw(screen); # (4) display a third ball
        ball3.update();

    screen.blit( font.render(str(scoreLeft) + ":" +
                             str(scoreRight), True, RED), [20, 20])
    screen.blit( font.render(str(int(playTime))+ " s", True, RED), [370, 20])

    if gameOver:
        centerImage(screen, font.render(winMsg, True, RED))
        ball.speed = 0
        ball2.speed = 0
        ball3.speed = 0
        ball.rect.center = [scrWidth/2, scrHeight/2]
        ball2.rect.center = [scrWidth/2, scrHeight/2]
        ball3.rect.center = [scrWidth/2, scrHeight/2]
        playTime =0.0
        screen.blit( font.render(str(int(playTime))+ " s", True, RED), [370,
20])


    pygame.display.update()

pygame.quit()
```


MegaPong
4:6
2 s


MegaPong
8:6
0 s
Left Wins!