1. Pygame is a mo dule that designed for writing games, depends on the creator if they give the source code people could modify it (I think)
2. Rectangle is a 2D shape that have 4 edges
3. Yes,

pygame.mixer.music.play("Something.wav")  
pygame.mixer.music.play(0)

1. Import time > time.sleep(40) < seconds

If using pygame, pygame.time.wait (to pause amount of time), pygame.time.delay (to pause amount of time).

1. Group is a bunch of sprites and sprites are simple game object that give a value so you can use blitme to visualize it

class Alien(Sprite):  
 def \_\_init\_\_(self, ai\_settings, screen):  
 super(Alien, self).\_\_init\_\_()  
 self.screen = screen  
 self.ai\_settings = ai\_settings  
  
 self.image = pygame.image.load("E.bmp")  
 self.rect = self.image.get\_rect()

--snip—

#add aliens = Group() on the main.py (for example def run\_game():)

1. Collision detection is detecting if two object hit each other.

def check\_bullet\_alien\_collisions(ai\_settings, screen, ship, aliens, bullets):  
 collisions = pygame.sprite.groupcollide(bullets, aliens, True, True)

1. –snip—

self.image = pygame.image.load("E.bmp")  
self.rect = self.image.get\_rect()

--snip—

def blitme(self):  
 self.screen.blit(self.image, self.rect)

#add this to the main,

def update\_screen(self):

self.something.draw(self.screen)

pygame.display.flip()

if I remember blitme is to show the image

1. self.coins = pygame.sprite.Group()

for i in range(0, 11):  
 self.coins.add(Coin(self))

def update\_screen(self):

self.coins.draw(self.screen)

1. there is no number nine because I sometime check the number from the last first
2. Game physics introduce real life physics into game so it seems kind off realistic, depends on what kind of game is that it become important, like for maybe just text game they don’t need any physics but if its like something that could be visualize they are important so it feels kinda real.
3. font = pygame.font.Font('freesansbold.ttf', 32)

text = font.render('GeeksForGeeks', True, green, blue)

textRect = text.get\_rect()

textRect.center = (X // 2, Y // 2)

for event in pygame.event.get():

if event.type == QUIT:

pygame.display.update



def check\_keydown\_events(event, ai\_settings, screen, ship, bullets):  
 *"""Respond to keypresses."""* if event.key == pygame.K\_RIGHT:  
 ship.moving\_right = True  
 elif event.key == pygame.K\_LEFT:  
 ship.moving\_left = True  
 elif event.key == pygame.K\_UP:  
 ship.moving\_up = True  
 elif event.key == pygame.K\_DOWN:  
 ship.moving\_down = True  
 elif event.key == pygame.K\_SPACE:  
 fire\_bullet(ai\_settings, screen, ship, bullets)  
 elif event.key == pygame.K\_q:  
 sys.exit()

def check\_keyup\_events(event, ship):  
 *"""Respond to key releases."""* if event.key == pygame.K\_RIGHT:  
 ship.moving\_right = False  
 elif event.key == pygame.K\_LEFT:  
 ship.moving\_left = False  
 elif event.key == pygame.K\_UP:  
 ship.moving\_up = False  
 elif event.key == pygame.K\_DOWN:  
 ship.moving\_down = False

--snip—

#add this to the one that want to move (for the example it’s a ship)

self.moving\_right = False  
self.moving\_left = False  
self.moving\_up = False  
self.moving\_down = False



self.screen.fill((0, 0, 0))

or

self.bg\_color = (255, 255, 255)



self.rect.x = randint(0, self.screen\_rect.right - self.rect.width)  
self.rect.y = randint(0, self.screen\_rect.bottom - self.rect.height)

on the class for example on the class ship

1. No and Yes? I by sometimes check the last number number first, so I check number nine and yeah.