from pygame import \*

from random import randint

from time import time as timer #импортируем функцию для засекания времени, чтобы интерпретатор не искал эту функцию в pygame модуле time, даём ей другое название сами

#подгружаем отдельно функции для работы со шрифтом

font.init()

font1 = font.Font(None, 80)

win = font1.render('YOU WIN!', True, (255, 255, 255))

lose = font1.render('YOU LOSE!', True, (180, 0, 0))

font2 = font.Font(None, 36)

#фоновая музыка

mixer.init()

mixer.music.load('space.ogg')

mixer.music.play()

fire\_sound = mixer.Sound('fire.ogg')

#нам нужны такие картинки:

img\_back = "galaxy.jpg" #фон игры

img\_bullet = "bullet.png" #пуля

img\_hero = "rocket.png" #герой

img\_enemy = "ufo.png" #враг

img\_ast = "asteroid.png" #астероид

score = 0 #сбито кораблей

goal = 20 #столько кораблей нужно сбить для победы

lost = 0 #пропущено кораблей

max\_lost = 10 #проиграли, если пропустили столько кораблей

life = 3  #очки жизни

#класс-родитель для других спрайтов

class GameSprite(sprite.Sprite):

#конструктор класса

  def \_\_init\_\_(self, player\_image, player\_x, player\_y, size\_x, size\_y, player\_speed):

      #вызываем конструктор класса (Sprite):

      sprite.Sprite.\_\_init\_\_(self)

      #каждый спрайт должен хранить свойство image - изображение

      self.image = transform.scale(image.load(player\_image), (size\_x, size\_y))

      self.speed = player\_speed

      #каждый спрайт должен хранить свойство rect - прямоугольник, в который он вписан

      self.rect = self.image.get\_rect()

      self.rect.x = player\_x

      self.rect.y = player\_y

#метод, отрисовывающий героя на окне

  def reset(self):

      window.blit(self.image, (self.rect.x, self.rect.y))

#класс главного игрока

class Player(GameSprite):

  #метод для управления спрайтом стрелками клавиатуры

   def update(self):

       keys = key.get\_pressed()

       if keys[K\_LEFT] and self.rect.x > 5:

           self.rect.x -= self.speed

       if keys[K\_RIGHT] and self.rect.x < win\_width - 80:

           self.rect.x += self.speed

   #метод "выстрел" (используем место игрока, чтобы создать там пулю)

   def fire(self):

       bullet = Bullet(img\_bullet, self.rect.centerx, self.rect.top, 15, 20, -15)

       bullets.add(bullet)

#класс спрайта-врага

class Enemy(GameSprite):

  #движение врага

  def update(self):

      self.rect.y += self.speed

      global lost

      #исчезает, если дойдёт до края экрана

      if self.rect.y > win\_height:

          self.rect.x = randint(80, win\_width - 80)

          self.rect.y = 0

          lost = lost + 1

#класс спрайта-пули

class Bullet(GameSprite):

  #движение врага

  def update(self):

      self.rect.y += self.speed

      #исчезает, если дойдёт до края экрана

      if self.rect.y < 0:

          self.kill()

#создаём окошко

win\_width = 700

win\_height = 500

display.set\_caption("Shooter")

window = display.set\_mode((win\_width, win\_height))

background = transform.scale(image.load(img\_back), (win\_width, win\_height))

#создаём спрайты

ship = Player(img\_hero, 5, win\_height - 100, 80, 100, 10)

#создание группы спрайтов-врагов

monsters = sprite.Group()

for i in range(1, 6):

  monster = Enemy(img\_enemy, randint(80, win\_width - 80), -40, 80, 50, randint(1, 5))

  monsters.add(monster)

#создание группы спрайтов-астероидов ()

asteroids = sprite.Group()

for i in range(1, 3):

   asteroid = Enemy(img\_ast, randint(30, win\_width - 30), -40, 80, 50, randint(1, 7))

   asteroids.add(asteroid)

 bullets = sprite.Group()

#переменная "игра закончилась": как только там True, в основном цикле перестают работать спрайты

finish = False

#основной цикл игры:

run = True #флаг сбрасывается кнопкой закрытия окна

rel\_time = False #флаг, отвечающий за перезарядку

num\_fire = 0  #переменная для подсчёта выстрелов

while run:

   #событие нажатия на кнопку “Закрыть”

   for e in event.get():

       if e.type == QUIT:

           run = False

       #событие нажатия на пробел - спрайт стреляет

       elif e.type == KEYDOWN:

           if e.key == K\_SPACE:

                   #проверяем, сколько выстрелов сделано и не происходит ли перезарядка

                   if num\_fire < 5 and rel\_time == False:

                       num\_fire = num\_fire + 1

                       fire\_sound.play()

                       ship.fire()

                   if num\_fire  >= 5 and rel\_time == False : #если игрок сделал 5 выстрелов

                       last\_time = timer() #засекаем время, когда это произошло

                       rel\_time = True #ставим флаг перезарядки

   #сама игра: действия спрайтов, проверка правил игры, перерисовка

   if not finish:

       #обновляем фон

       window.blit(background,(0,0))

       #производим движения спрайтов

       ship.update()

       monsters.update()

       asteroids.update()

       bullets.update()

       #обновляем их в новом местоположении при каждой итерации цикла

       ship.reset()

       monsters.draw(window)

       asteroids.draw(window)

       bullets.draw(window)

       #перезарядка

       if rel\_time == True:

           now\_time = timer() #считываем время

           if now\_time - last\_time < 3: #пока не прошло 3 секунды выводим информацию о перезарядке

               reload = font2.render('Wait, reload...', 1, (150, 0, 0))

               window.blit(reload, (260, 460))

           else:

               num\_fire = 0   #обнуляем счётчик пуль

               rel\_time = False #сбрасываем флаг перезарядки

       #проверка столкновения пули и монстров (и монстр, и пуля при касании исчезают)

       collides = sprite.groupcollide(monsters, bullets, True, True)

       for c in collides:

           #этот цикл повторится столько раз, сколько монстров подбито

           score = score + 1

           monster = Enemy(img\_enemy, randint(80, win\_width - 80), -40, 80, 50, randint(1, 5))

           monsters.add(monster)

       #если спрайт коснулся врага, уменьшает жизнь

       if sprite.spritecollide(ship, monsters, False) or sprite.spritecollide(ship, asteroids, False):

           sprite.spritecollide(ship, monsters, True)

           sprite.spritecollide(ship, asteroids, True)

           life = life -1

       #проигрыш

       if life == 0 or lost >= max\_lost:

           finish = True #проиграли, ставим фон и больше не управляем спрайтами.

           window.blit(lose, (200, 200))

       #проверка выигрыша: сколько очков набрали?

       if score >= goal:

           finish = True

           window.blit(win, (200, 200))

       #пишем текст на экране

       text = font2.render("Счет: " + str(score), 1, (255, 255, 255))

       window.blit(text, (10, 20))

       text\_lose = font2.render("Пропущено: " + str(lost), 1, (255, 255, 255))

       window.blit(text\_lose, (10, 50))

       #задаём разный цвет в зависимости от количества жизней

       if life == 3:

           life\_color = (0, 150, 0)

       if life == 2:

           life\_color = (150, 150, 0)

       if life == 1: