Exercise 2(its not here because im a lazy butt)

Exercise 3

:import pygame

import time

#init

pygame.init()

#surface size

display\_width=600

display\_height=600

#color def

black= (0,0,0)

white=(255,255,255)

red=(255,0,0)

cyan=(65,255,243)

#more surface info

gameDisplay = pygame.display.set\_mode((display\_width,display\_height))

pygame.display.set\_caption('this is all you get')

#clock

clock = pygame.time.Clock()

#the refrence imiges

Card0= pygame.image.load('./c0.png')

Card1= pygame.image.load('./c1.png')

Card2= pygame.image.load('./c2.png')

Card3= pygame.image.load('./c3.png')

Card4= pygame.image.load('./c4.png')

#runing the elements

def crd0(x,y):

gameDisplay.blit(Card0,(x,y))

def crd1(x,y):

gameDisplay.blit(Card1,(x,y))

def crd2(x,y):

gameDisplay.blit(Card2,(x,y))

def crd3(x,y):

gameDisplay.blit(Card3,(x,y))

def crd4(x,y):

gameDisplay.blit(Card4,(x,y))

all\_y=200

i = True

while i:

for event in pygame.event.get():

if event.type==pygame.QUIT:

i=False

gameDisplay.fill(white)

crd0(0,all\_y)

crd1(120,all\_y)

crd2(240,all\_y)

crd3(360,all\_y)#360? 360 noscope!

crd4(480,all\_y)

pygame.display.update()

clock.tick(60)

pygame.quit()

quit()

(a playable version is in ch17\_ex3)

Exercise 4

: #!/usr/bin/env python

import random, os.path

#import basic pygame modules

import pygame

from pygame.locals import \*

#see if we can load more than standard BMP

if not pygame.image.get\_extended():

raise SystemExit("Sorry, extended image module required")

#game constants

MAX\_SHOTS = 5 #most player bullets onscreen

ALIEN\_ODDS = 22 #chances a new alien appears

BOMB\_ODDS = 60 #chances a new bomb will drop

ALIEN\_RELOAD = 12 #frames between new aliens

SCREENRECT = Rect(0, 0, 640, 480)

SCORE = 0

main\_dir = os.path.split(os.path.abspath(\_\_file\_\_))[0]

def load\_image(file):

"loads an image, prepares it for play"

file = os.path.join(main\_dir, 'data', file)

try:

surface = pygame.image.load(file)

except pygame.error:

raise SystemExit('Could not load image "%s" %s'%(file, pygame.get\_error()))

return surface.convert()

def load\_images(\*files):

imgs = []

for file in files:

imgs.append(load\_image(file))

return imgs

class dummysound:

def play(self): pass

def load\_sound(file):

if not pygame.mixer: return dummysound()

file = os.path.join(main\_dir, 'data', file)

try:

sound = pygame.mixer.Sound(file)

return sound

except pygame.error:

print ('Warning, unable to load, %s' % file)

return dummysound()

# each type of game object gets an init and an

# update function. the update function is called

# once per frame, and it is when each object should

# change it's current position and state. the Player

# object actually gets a "move" function instead of

# update, since it is passed extra information about

# the keyboard

class Player(pygame.sprite.Sprite):

speed = 10

bounce = 24

gun\_offset = -11

images = []

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image = self.images[0]

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

self.reloading = 0

self.origtop = self.rect.top

self.facing = -1

def move(self, direction):

if direction: self.facing = direction

self.rect.move\_ip(direction\*self.speed, 0)

self.rect = self.rect.clamp(SCREENRECT)

if direction < 0:

self.image = self.images[0]

elif direction > 0:

self.image = self.images[1]

self.rect.top = self.origtop - (self.rect.left//self.bounce%2)

def gunpos(self):

pos = self.facing\*self.gun\_offset + self.rect.centerx

return pos, self.rect.top

class Alien(pygame.sprite.Sprite):

speed = 13

animcycle = 12

images = []

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image = self.images[0]

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

self.facing = random.choice((-1,1)) \* Alien.speed

self.frame = 0

if self.facing < 0:

self.rect.right = SCREENRECT.right

def update(self):

self.rect.move\_ip(self.facing, 0)

if not SCREENRECT.contains(self.rect):

self.facing = -self.facing;

self.rect.top = self.rect.bottom + 1

self.rect = self.rect.clamp(SCREENRECT)

self.frame = self.frame + 1

self.image = self.images[self.frame//self.animcycle%3]

class Explosion(pygame.sprite.Sprite):

defaultlife = 12

animcycle = 3

images = []

def \_\_init\_\_(self, actor):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image = self.images[0]

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

self.life = self.defaultlife

def update(self):

self.life = self.life - 1

self.image = self.images[self.life//self.animcycle%2]

if self.life <= 0: self.kill()

class Shot(pygame.sprite.Sprite):

speed = -11

images = []

def \_\_init\_\_(self, pos):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image = self.images[0]

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

def update(self):

self.rect.move\_ip(0, self.speed)

if self.rect.top <= 0:

self.speed=11

elif self.rect.top >= 420:

self.kill()

class Bomb(pygame.sprite.Sprite):

speed = 9

images = []

def \_\_init\_\_(self, alien):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image = self.images[0]

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

alien.rect.move(0,5).midbottom)

def update(self):

self.rect.move\_ip(0, self.speed)

if self.rect.bottom >= 470:

Explosion(self)

self.kill()

class Score(pygame.sprite.Sprite):

def \_\_init\_\_(self):

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

self.font = pygame.font.Font(None, 20)

self.font.set\_italic(1)

self.color = Color('white')

self.lastscore = -1

self.update()

self.rect = self.image.get\_rect().move(10, 450)

def update(self):

if SCORE != self.lastscore:

self.lastscore = SCORE

msg = "Score: %d" % SCORE

self.image = self.font.render(msg, 0, self.color)

def main(winstyle = 0):

# Initialize pygame

pygame.init()

if pygame.mixer and not pygame.mixer.get\_init():

print ('Warning, no sound')

pygame.mixer = None

# Set the display mode

winstyle = 0 # |FULLSCREEN

bestdepth = pygame.display.mode\_ok(SCREENRECT.size, winstyle, 32)

screen = pygame.display.set\_mode(SCREENRECT.size, winstyle, bestdepth)

#Load images, assign to sprite classes

#(do this before the classes are used, after screen setup)

img = load\_image('player1.gif')

Player.images = [img, pygame.transform.flip(img, 1, 0)]

img = load\_image('explosion1.gif')

Explosion.images = [img, pygame.transform.flip(img, 1, 1)]

Alien.images = load\_images('alien1.gif', 'alien2.gif', 'alien3.gif')

Bomb.images = [load\_image('bomb.gif')]

Shot.images = [load\_image('shot.gif')]

#decorate the game window

icon = pygame.transform.scale(Alien.images[0], (32, 32))

pygame.display.set\_icon(icon)

pygame.display.set\_caption('Pygame Aliens')

pygame.mouse.set\_visible(0)

#create the background, tile the bgd image

bgdtile = load\_image('background.gif')

background = pygame.Surface(SCREENRECT.size)

for x in range(0, SCREENRECT.width, bgdtile.get\_width()):

background.blit(bgdtile, (x, 0))

screen.blit(background, (0,0))

pygame.display.flip()

#load the sound effects

boom\_sound = load\_sound('boom.wav')

shoot\_sound = load\_sound('car\_door.wav')

if pygame.mixer:

music = os.path.join(main\_dir, 'data', 'house\_lo.wav')

pygame.mixer.music.load(music)

pygame.mixer.music.play(-1)

# Initialize Game Groups

aliens = pygame.sprite.Group()

shots = pygame.sprite.Group()

bombs = pygame.sprite.Group()

all = pygame.sprite.RenderUpdates()

lastalien = pygame.sprite.GroupSingle()

#assign default groups to each sprite class

Player.containers = all

Alien.containers = aliens, all, lastalien

Shot.containers = shots, all

Bomb.containers = bombs, all

Explosion.containers = all

Score.containers = all

#Create Some Starting Values

global score

alienreload = ALIEN\_RELOAD

kills = 0

clock = pygame.time.Clock()

#initialize our starting sprites

global SCORE

player = Player()

Alien() #note, this 'lives' because it goes into a sprite group

if pygame.font:

all.add(Score())

while player.alive():

#get input

for event in pygame.event.get():

if event.type == QUIT or \

(event.type == KEYDOWN and event.key == K\_ESCAPE):

return

keystate = pygame.key.get\_pressed()

# clear/erase the last drawn sprites

all.clear(screen, background)

#update all the sprites

all.update()

#handle player input

direction = keystate[K\_RIGHT] - keystate[K\_LEFT]

player.move(direction)

firing = keystate[K\_SPACE]

if not player.reloading and firing and len(shots) < MAX\_SHOTS:

Shot(player.gunpos())

shoot\_sound.play()

player.reloading = firing

# Create new alien

if alienreload:

alienreload = alienreload - 1

elif not int(random.random() \* ALIEN\_ODDS):

Alien()

alienreload = ALIEN\_RELOAD

# Drop bombs

if lastalien and not int(random.random() \* BOMB\_ODDS):

Bomb(lastalien.sprite)

# Detect collisions

for alien in pygame.sprite.spritecollide(player, aliens, 1):

boom\_sound.play()

Explosion(alien)

Explosion(player)

SCORE = SCORE + 1

player.kill()

for alien in pygame.sprite.groupcollide(shots, aliens, 1, 1).keys():

boom\_sound.play()

Explosion(alien)

SCORE = SCORE + 1

for bomb in pygame.sprite.spritecollide(player, bombs, 1):

boom\_sound.play()

Explosion(player)

Explosion(bomb)

player.kill()

#draw the scene

dirty = all.draw(screen)

pygame.display.update(dirty)

#cap the framerate

clock.tick(40)

if pygame.mixer:

pygame.mixer.music.fadeout(1000)

pygame.time.wait(1000)

pygame.quit()

#call the "main" function if running this script

if \_\_name\_\_ == '\_\_main\_\_': main()

(a playable version is in the folder: jacobledbetter\_10\_30\_15\_ch17\_exercise 4\_bouncy aliens)

Exserzice 5(its also not here but because we weren’t supposed to do it (I think))