Lecture 4/17

Things focused on this week:

1. Adding pygame.quit()
2. Closing the game by clicking escape key
3. Use different size images for the meteors
4. Drawing Score on our game

**Adding pygame.quit()**

At the end of the program to have a catch all, you can add pygame.quit() at the end of the program

**Escape Key**

I wanted to a quicker way to end the game, instead of clicking the close window button, clicking the escape button can also close the game.

Find the event.type == pygame.KEYDOWN line and add another if statement

*if* event.key == pygame.K\_ESCAPE:  
 running = *False*

**Different size images for the meteors**

# create an empty list of meteor images

meteor\_images = []

# these are the images from the PNG folder that we got from free game art folder  
meteor\_list = ['meteorBrown\_big1.png', 'meteorBrown\_big2.png', 'meteorBrown\_big3.png', 'meteorBrown\_big4.png',  
 'meteorBrown\_med1.png', 'meteorBrown\_med3.png', 'meteorBrown\_small1.png', 'meteorBrown\_small2.png']

# Create a for loop of the images list  
*for* img *in* meteor\_list:

# adding the images to the meteor\_images list so that we can use the list  
meteor\_images.append(pygame.image.load(path.join(img\_dir, img)).convert())

**NOTE: DON’T FORGET TO COMMENT OUT THE METEOR\_IMG. WE WON’T BE USING THIS SINGLE IMAGE ANYMORE, RATHER MULTIPLE IMAGES**

In your Mob class, change the following code

self.image\_orig = random.choice(meteor\_images)

**Drawing Score on our game**

Pygame does not offer a great way to draw a UI, but rather we can create one and append it to the screen

# create a variable called font name and assign a #family that pygame can match with the inbuilt in #font classes in your computer

font\_name = pygame.font.match\_font('arial')

# Create the draw text method that will take #parameters as surface for the text to be drawn on, # the text to be displayed, the size of the text, the x and y coordinates of where to show the text  
*def* draw\_text(*surf*, *text*, *size*, *x*, *y*):

# Create a variable called font with the font name to be used and the size of the text to be displayed  
 font = pygame.font.Font(font\_name, *size*)

# Where the should the text be rendered, the second parameter is set to TRUE to symbolize anti-aliasing(to blend better with the background)  
 text\_surface = font.render(*text*, *True*, WHITE)

# if we make a surface, we need to create the rectangle representing the surface. This is the same logic we used for the mob, player and bullets  
 text\_rect = text\_surface.get\_rect()

# where does the text need to be shown, based on the x and y coordinates  
 text\_rect.midtop = (*x*, *y*)

# Where should the surfaces converge. The text\_surface and the text\_rect will combine to create the text we want to show  
 *surf*.blit(text\_surface, text\_rect)

In the draw section towards the bottom of the program, under the all\_sprites.draw(screen), we will use the draw\_text function, we just created.

draw\_text(screen, "Score: {}".format(score), 24, WIDTH/2, 10)

Lastly, in your for loops for hits where we check if the bullets collided with the mobs, add the following line, highlighted in green

*for* hit *in* hits:  
 score += 50 - hit.radius  
 m = Mob()  
 all\_sprites.add(m)  
 mobs.add(m)