**Adding Baddie Hit Points**

Today I’m going to make baddies have hit points, so some baddies needs to be hit with multiple bullets to destroy them.

By default, all baddies will have one hit point, and only need 1 bullet to kill them. Some baddies will have more hit points. The number of hit points will be controlled by the SpaceshipData class. The tracking of each baddie’s hit points will be controlled by the Baddie class.

**Adding Hit Points to the Baddie Class**

In baddie.py, we need to add a hit_points variable to each of the baddies to track its health. If the hit_points is 0 or less, then the baddie will be dead.

In the __init__ method, near the end, add a line like this to create the hit_points variable and make the default 1 for all baddies.

```python
self.hit_points = 1
```

**Allowing the Hit Points to be Controlled**

Still in baddie.py, create a new method to allow the SpaceshipData to set the strength of each baddie. Add these lines of code between or after the other methods. Make sure you get the indentation the same as other methods in the class.

```python
def setHitPoints(self, hit_points):
    self.hit_points = hit_points
```

**Allowing the Baddie to Take a Damage**

Still in baddie.py, create a new method to allow the baddie to take damage. Add these lines of code between or after the other methods. Make sure you get the indentation the same as other methods in the class.

```python
def decreaseHitPoints(self, damage):
    self.hit_points = self.hit_points - damage
    if self.hit_points <= 0:
        self.setAlive(False)
```

Notice this method will cause the baddie to die if the hit_points become 0 or less.

**Causing Baddie Damage**

Now, in SpaceshipData.py, we want to use the hit point system. In the evolve method, find the location where baddies are killed with the line:

```python
baddie.setAlive(False)
```

Replace this line with

```python
baddie.decreaseHitPoints(1)
```

This will cause 1 damage point every time a bullet hits the baddie. The baddie will be killed when the hit points reaches 0.

**Making Stronger Baddies**

Still in SpaceshipData.py, add a new method that will allow you to create stronger baddies. Be sure to get the indentation correct:

```python
def addStrongBaddie(self):
    new_baddie = Baddie( self.baddie_width, self.baddie_height, self.width, 
                        random.randint(0,(self.height-self.baddie_height)), self.baddie_color )
    new_baddie.setHitPoints(2)
```
Notice this function will create baddies with 2 hit points.

This method will allow you to create the strong baddies, but you need to call it to actually create them. In the evolve function, find where addBaddie is called. Add these two lines after that:

```python
elif random.randint(1, self.frame_rate) == 1:
    self.addStrongBaddie()
```

This will cause a strong baddie to be created about once per second, but only if a normal baddie was not created in the same frame.

**Optional Fun**

If you want to be able to visually see the baddies with more than 1 hit point, you can draw them differently. For example, replacing the draw method in baddie.py with these lines of code will cause strong baddies to be drawn as white rectangles, and weak (1 hit point) baddies to be drawn with the normal baddie color.

```python
def draw(self, surface):
    rect = pygame.Rect( self.x, self.y, self.width, self.height )
    if self.hit_points > 1:
        color = (255, 255, 255)
    else:
        color = self.color
    pygame.draw.rect(surface, color, rect)
return
```

**Finer Points**

You could use similar code to create baddies with different attributes, and have them behave differently.