Copter

- Install PyGame
  - `pip install pygame`
- Download
  - https://github.com/calaldees/TeachProgramming/blob/master/teachprogramming/static/projects/game/animation_base_pygame.py
- Image Size
  - Background (3000, 360)? - Copter (48, 24)?

```
import pygame
from animation_base_pygame import PygameBase

class CopterGame(PygameBase):
    def __init__(self):
        self.background_color = (0, 0, 0, 0)
        self.reset()
        super().__init__(resolution=(640,360))
    def reset(self):
        pass
    def loop(self, screen, frame):
        pygame.draw.rect(screen, self.background_color, (0,0)+screen.get_size())

if __name__ == '__main__':
    CopterGame().run()
```

Background

```
 class CopterGame(PygameBase):
     def __init__(self):
+        self.background_image = pygame.image.load("images/CopterLevel1.png")
         self.background_color = (0, 0, 0, 0)
...
         pygame.draw.rect(screen, self.background_color, (0,0)+screen.get_size())
+        screen.blit(self.background_image, (0, 0))
```

Background Move

```
     def reset(self):
-        pass
+        self.background_x_pos = 0
     def loop(self, screen, frame):
+        self.background_x_pos += 1
...
-        screen.blit(self.background_image, (0, 0))
+        screen.blit(self.background_image, (-self.background_x_pos, 0))
```

Copter

```
 class CopterGame(PygameBase):
     def __init__(self):
         self.background_image = pygame.image.load("images/CopterLevel1.png")
         self.background_color = (0, 0, 0, 0)
+        self.copter_image     = pygame.image.load("images/ship.png")
         self.reset()
         super().__init__(resolution=(640,360))
     def reset(self):
         self.background_x_pos = 0
+        self.copter_x_pos = 50
+        self.copter_y_pos = 100
...

         screen.blit(self.background_image, (-self.background_x_pos, 0))
+        screen.blit(self.copter_image, (self.copter_x_pos, self.copter_y_pos))
+
```

Copter Move

```
     def loop(self, screen, frame):
         self.background_x_pos += 1

+        if self.keys[pygame.K_SPACE]: self.copter_y_pos += -2
+        else                        : self.copter_y_pos +=  1
+
```

Collision Single

```
         else                        : self.copter_y_pos +=  1

+        def safe_get_pixel(p):
+            try   : return self.background_image.get_at(p)
+            except: return (0,0,0,0)
+        point = (self.background_x_pos + int(self.copter_x_pos), int(self.copter_y_pos))
+        pixel = safe_get_pixel(point)
+        r,g,b,a = pixel
+        if a < 10:
+            pass
+        else:
+            self.reset()
+
         pygame.draw.rect(screen, self.background_color, (0,0)+screen.get_size())
```

Optional Advanced Bits

These can be attempted in any order.
(although there is some overlap - so look out for the detail)

Level

```
-       self.background_image = pygame.image.load("images/CopterLevel1.png")
...
+       self.level_color = (255, 255, 0, 255)
+       self.level_number = 1
+       self.load_level()
-       self.reset()
        super().__init__(resolution=(640,360))
+   def load_level(self):
+       self.background_image = pygame.image.load(f"images/CopterLevel{self.level_number}.png")
+       self.reset()
...
            if a < 10:
                pass
+           elif pixel == self.level_color:
+               self.level_number += 1
+               self.load_level()
            else:
                self.reset()
```

Physics

```
    def reset(self):
        self.background_x_pos = 0
        self.copter_x_pos = 50
        self.copter_y_pos = 100
+       self.copter_x_vel = 0
+       self.copter_y_vel = 0
    def loop(self, screen, frame):
        self.background_x_pos += 1

-       if self.keys[pygame.K_SPACE]: self.copter_y_pos += -2
-       else                        : self.copter_y_pos +=  1
+       if self.keys[pygame.K_UP   ]: self.copter_y_vel += -0.1
+       if self.keys[pygame.K_DOWN ]: self.copter_y_vel +=  0.1
+       if self.keys[pygame.K_LEFT ]: self.copter_x_vel += -0.1
+       if self.keys[pygame.K_RIGHT]: self.copter_x_vel +=  0.1

+       self.copter_x_vel  = self.copter_x_vel * 0.99
+       self.copter_y_vel  = self.copter_y_vel * 0.99
+       self.copter_y_vel += float(0.025)
+       self.copter_x_pos += self.copter_x_vel
+       self.copter_y_pos += self.copter_y_vel
+
        def safe_get_pixel(p):
```

Parallax

```
    def load_level(self):
-       self.background_image = pygame.image.load(f"images/CopterLevel{self.level_number}.png")
+       self.background_images = [
+           pygame.image.load(file)
+           for file in sorted(pathlib.Path('images').glob(f"*CopterLevel{self.level_number}*"))
+       ]
        self.reset()
...
        pygame.draw.rect(screen, self.background_color, (0,0)+screen.get_size())
-       screen.blit(self.background_image, (-self.background_x_pos, 0))
+       for parallax_number, background_image in sorted(enumerate(self.background_images), reverse
+           screen.blit(background_image, (-self.background_x_pos/2**parallax_number, 0))
```

```
        def safe_get_pixel(p):
-           try   : return self.background_image.get_at(p)
+           try   : return self.background_images[0].get_at(p)
            except: return None
```

Collision Multi

```
        self.copter_image     = pygame.image.load("images/ship.png")
+       self.copter_collision_points = ((0,0),(32,9),(17,2),(22,12),(2,12))
        self.reset()
...
        def safe_get_pixel(p):
            try   : return self.background_image.get_at(p)
            except: return (0,0,0,0)
-       point = (self.background_x_pos + int(self.copter_x_pos), int(self.copter_y_pos))
+       for x, y in self.copter_collision_points:
+           point = (self.background_x_pos + int(self.copter_x_pos) + x, int(self.copter_y_pos) +
            pixel = safe_get_pixel(point)
```

(all bits need indenting from collision_single)