Space Invaders - Iterative Enhancement Plan (IEP)

For teams-of-two, Person 1 and Person 3 are merged. All team members help teammates as needed. All stages delete temporary code from previous stages. **Team members coordinate whenever they touch the shared Controller, Game and View files. Beware of Git conflicts!**

1. [**Entire team**] [*Most of the code for this stage is included in the starting code.*] A blank screen appears. A message appears on the Console that a Model (Game), View and Controller have been constructed. Events are printed (for this stage only) on the Console. The background, title and size are set and appear. Files contain team member names. [After this stage, the code is a bare-bones Model-View-Controller architecture.]

2. [**Entire team**, with your instructor, dividing up the work] The Game object constructs a Fighter, Enemy, Enemies, Missile, and Missiles object, using stubs for those 5 classes, and with each constructor having the ***screen*** as an argument. The constructor method for each of these prints a simple message, temporarily.

3. [**Person 1**] The Fighter appears, centered horizontally and near the bottom of the screen. The Fighter is an image in the ***Assets*** folder. Sub-stages:
   a. The Game's ***draw_game*** method calls the Fighter's ***draw*** method, which PRINTS "draw fighter", temporarily.
   b. The Fighter appears on the screen at a hard-coded place.
   c. The Fighter shows up on the screen as specified.

4. [**Person 1**] The Fighter moves left/right a fixed number of pixels (try 5) per game loop cycle whenever the left/right arrow key is in the PRESSED state, but restricted so that the Fighter does not go more than ½ of the Fighter's width off the left or right edges of the screen. Sub-stages:
   a. Left/right keys PRINT "left", "right".
   b. Left/right keys make the Fighter move left/right.
   c. Fighter does not go off the screen, per specification.

5. [**Person 1**] When the Space bar is pressed, the Fighter fires a Missile, which causes a "pew" sound. (*Holding* the Space bar down does NOT generate additional Missile objects.) For now, no Missile actually appears; just the “pew” sound, which is via a sound file in the ***Assets*** folder.

6. [**Person 2**] A single Enemy appears, somewhere near the top of the screen. Enemies are images in the ***Assets*** folder. Sub-stages:
   a. The Game's ***draw_game*** method calls the Enemy's ***draw*** method, which PRINTS "draw enemy", temporarily.
   b. The Enemy itself shows up on the screen as specified.

7. [**Person 2**] At each cycle of the game loop, the Enemy moves sideways a specified number of pixels (try 5), and when it gets 100 units from where it starts, it reverses direction and moves down a specified number of pixels (try 10).

8. [**Person 2**] A two-dimensional array of Enemies appear, centered horizontally, near the top of the screen, in 5 rows of 8 Enemy objects per row. [Implement this in the Enemies class, with the Enemies object constructing and storing the Enemy objects.] The temporary Enemy is removed.

9. [**Person 2**] Enemies move. Sub-stages:
   a. At each cycle of the game loop, each Enemy moves sideways a specified number of pixels (try 5), and when it gets 100 units from where it starts, it reverses direction and moves down a specified number of pixels (try 10).
   b. An Enemy is removed from the Enemies object’s list of Enemy objects when the Enemy goes below the bottom of the screen. [Test this by printing the length of the Enemies list and being

sure that it reduces as Enemies go below the bottom of the screen.]

10. [**Person 3**] A single Missile appears, somewhere near the bottom of the screen. Missiles are filled red lines, 4 pixels wide and 8 pixels tall. Sub-stages:
    a. The Game's ***draw_game*** method calls the Missile's ***draw*** method, which PRINTS "draw missile", temporarily.
    b. The Missile itself shows up on the screen as specified.

11. [**Person 3**] At each cycle of the game loop, the Missile moves up a specified number of pixels (try 5).

12. [**Person 3**] Two Missiles appear, at different places on the screen. [Implement this in the Missiles class, with the Missiles object constructing and storing the Missile objects.] The temporary Missile is removed.

13. [**Person 3**] Missiles move. Sub-stages:
    a. At each cycle of the game loop, each Missile moves up a specified number of pixels (try 5).
    b. A Missile is removed from the Missile object's list of Missile objects when the Missile goes above the top of the screen. [Test this by printing the length of the list of Missile objects and being sure that it reduces as Missile objects go above the top of the screen.]

14. [**Person 1 and 3, working together**] When the Space bar is pressed, the Fighter fires a Missile, which causes a "pew" sound (per a previous stage). As before, HOLDING the Space bar down does NOT generate additional Missile objects.

    At this stage, each Missile appears on the screen centered at the Fighter horizontally, with the top of the Missile at the same y-position as the Fighter, in response to pressing the space bar. [To test multiple Missiles, move the Fighter while firing.]

15. [**Person 2 and 3, working together**] Missiles touching enemies cause the missiles and enemies to explode (i.e. be removed from their lists) and an "explosion" sound occurs. If multiple missiles are touching the same enemy, the explosion sound happens only once. Sub-stages:
    a. At each cycle of the game loop, each Missile touching an Enemy is removed from the list of Missile objects in the Missiles object.
    b. When a Missile is touching an Enemy, the Enemy is marked as "has_exploded". Any Enemy that changes its "has_exploded" from False to True causes an "explosion" sound, per a sound in the ***Assets*** folder.

16. [**Person 1 and 2, working together**] If an Enemy touches the Fighter, the Fighter explodes (i.e., disappears, with an "explosion" sound), the main game loop stops, the gameover.png image appears at (170, 200), and a "you lose" sound is played.

17. If the Fighter destroys all the Enemies in the enemy fleet, the game starts again from the beginning.

Reminders:

1. Image / blit:

```
self.image = pygame.image.load("../assets/fighter.png")
self.screen.blit(self.image, (self.x, self.y))
```

2. Draw line:

```
pygame.draw.line(self.screen, self.color, (self.x, self.y),
    (self.x, self.y + self.height), self.width)
```

3. Play sound:

```
self.fire_sound = pygame.mixer.Sound("../assets/pew.wav")
self.fire_sound.play()
```

4. Key interaction:

```
if pressed_keys[pygame.K_LEFT]:  or  if key_was_pressed_...
```