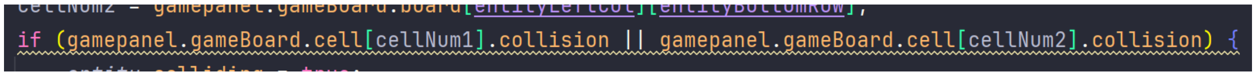
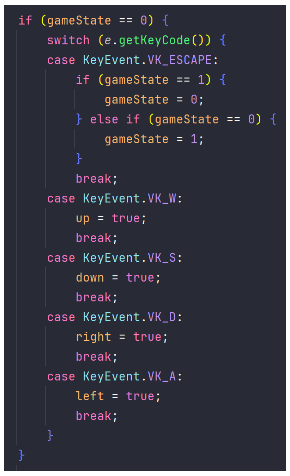
1. **Lack of Documentation:** Entity class documentation was lacking. Added javadoc comments to clarify the purpose of the variables. Previously, there was no documentation or explanation for variable names in the file.
2. **Dead Code:** The BufferedImages up0, down0, left0, and right0 were not being used. They have now been removed.
3. **Unnecessary if/else or switch/case statements**: Reduced redundancy by simplifying the if statement.



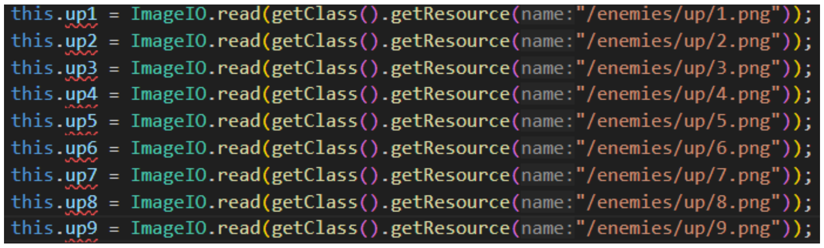
```
if (gamepanel.gameBoard.cell[cellNum1].collision || gamepanel.gameBoard.cell[cellNum2].collision) {
```

4. **Lack of Documentation**: Added in proper documentation for the numerous variables and instances initialized in the Game class.
5. **Lack of Documentation:** Provided better documentation detailing which game states it goes to the update method.
6. **Unnecessary if/else or switch/case statements**: Switched if/else statements to switch cases to better test equality of values, increase efficiency, and increase readability when handling various key presses.



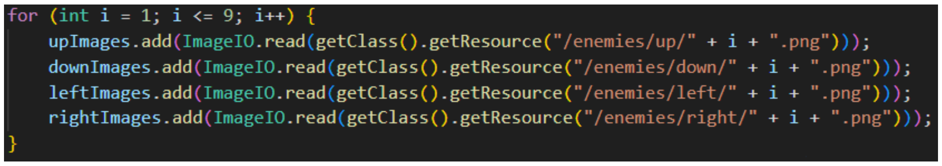
```
if (gameState == 0) {
    switch (e.getKeyCode()) {
    case KeyEvent.VK_ESCAPE:
        if (gameState == 1) {
            gameState = 0;
        } else if (gameState == 0) {
            gameState = 1;
        }
        break;
    case KeyEvent.VK_W:
        up = true;
        break;
    case KeyEvent.VK_S:
        down = true;
        break;
    case KeyEvent.VK_D:
        right = true;
        break;
    case KeyEvent.VK_A:
        left = true;
        break;
    }
}
```

7. **Poorly Structured Code:** When reading sprite files in the raccoon and player classes, each one was read on a separate line. This was further compounded by the fact that each entity sprite had 4 possible directions. This has been converted into a loop to make the code more readable. Before (raccoon):



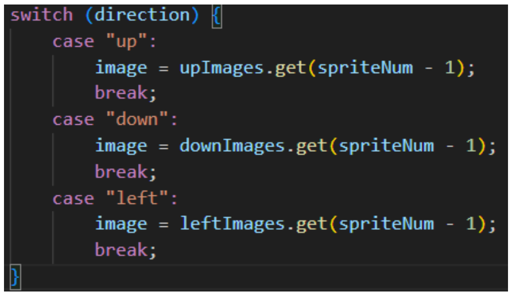
```
this.up1 = ImageIO.read(getClass().getResource(name:"/enemies/up/1.png"));
this.up2 = ImageIO.read(getClass().getResource(name:"/enemies/up/2.png"));
this.up3 = ImageIO.read(getClass().getResource(name:"/enemies/up/3.png"));
this.up4 = ImageIO.read(getClass().getResource(name:"/enemies/up/4.png"));
this.up5 = ImageIO.read(getClass().getResource(name:"/enemies/up/5.png"));
this.up6 = ImageIO.read(getClass().getResource(name:"/enemies/up/6.png"));
this.up7 = ImageIO.read(getClass().getResource(name:"/enemies/up/7.png"));
this.up8 = ImageIO.read(getClass().getResource(name:"/enemies/up/8.png"));
this.up9 = ImageIO.read(getClass().getResource(name:"/enemies/up/9.png"));
```

After:



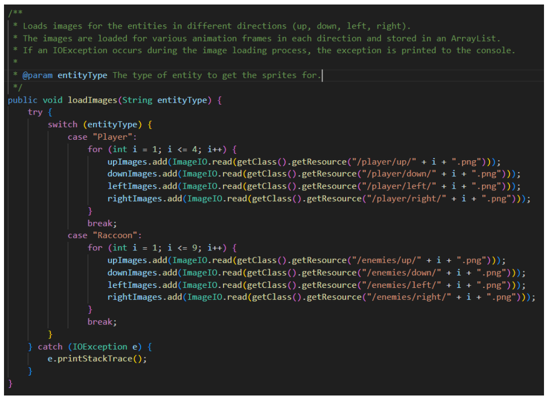
```
for (int i = 1; i <= 9; i++) {
    upImages.add(ImageIO.read(getClass().getResource("/enemies/up/" + i + ".png")));
    downImages.add(ImageIO.read(getClass().getResource("/enemies/down/" + i + ".png")));
    leftImages.add(ImageIO.read(getClass().getResource("/enemies/left/" + i + ".png")));
    rightImages.add(ImageIO.read(getClass().getResource("/enemies/right/" + i + ".png")));
}
```

8. **Unnecessary if/else or switch/case statements:** After changing how sprites were loaded by adding them into an arraylist, the way sprites were displayed also had to be changed. Previously, we were using if/else statements to check the value of spriteNum. The new implementation is:

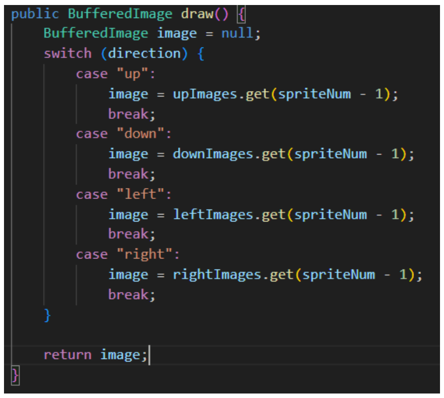


```java
switch (direction) {
    case "up":
        image = upImages.get(spriteNum - 1);
        break;
    case "down":
        image = downImages.get(spriteNum - 1);
        break;
    case "left":
        image = leftImages.get(spriteNum - 1);
        break;
}
```

9. **Code Duplication:** From the above changes, we can now pull the methods to the entity class, since they are almost identical between the player and raccoon classes.



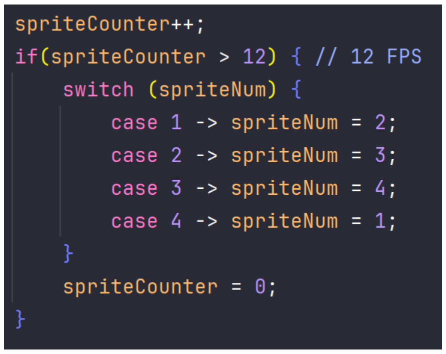
```java
/**
 * Loads images for the entities in different directions (up, down, left, right).
 * The images are loaded for various animation frames in each direction and stored in an ArrayList.
 * If an IOException occurs during the image loading process, the exception is printed to the console.
 *
 * @param entityType The type of entity to get the sprites for.
 */
public void loadImages(String entityType) {
    try {
        switch (entityType) {
            case "Player":
                for (int i = 1; i <= 4; i++) {
                    upImages.add(ImageIO.read(getClass().getResource("/player/up/" + i + ".png")));
                    downImages.add(ImageIO.read(getClass().getResource("/player/down/" + i + ".png")));
                    leftImages.add(ImageIO.read(getClass().getResource("/player/left/" + i + ".png")));
                    rightImages.add(ImageIO.read(getClass().getResource("/player/right/" + i + ".png")));
                }
                break;
            case "Raccoon":
                for (int i = 1; i <= 9; i++) {
                    upImages.add(ImageIO.read(getClass().getResource("/enemies/up/" + i + ".png")));
                    downImages.add(ImageIO.read(getClass().getResource("/enemies/down/" + i + ".png")));
                    leftImages.add(ImageIO.read(getClass().getResource("/enemies/left/" + i + ".png")));
                    rightImages.add(ImageIO.read(getClass().getResource("/enemies/right/" + i + ".png")));
                }
                break;
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



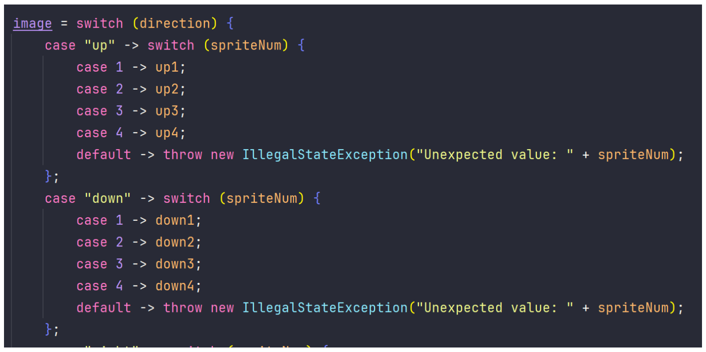
```java
public BufferedImage draw() {
    BufferedImage image = null;
    switch (direction) {
        case "up":
            image = upImages.get(spriteNum - 1);
            break;
        case "down":
            image = downImages.get(spriteNum - 1);
            break;
        case "left":
            image = leftImages.get(spriteNum - 1);
            break;
        case "right":
            image = rightImages.get(spriteNum - 1);
            break;
    }

    return image;
}
```

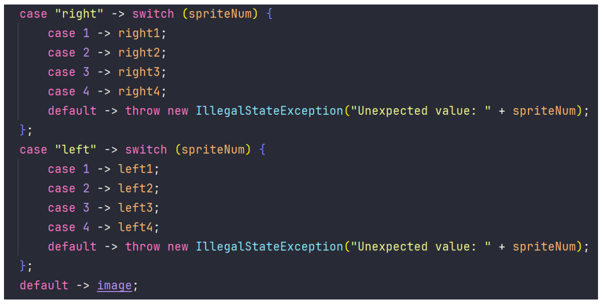
10. **Unnecessary if/else or switch/case statements**: Implemented switch cases to better test equality of the sprite images as they're going through each image at a certain frame rate. This change enhances the readability of our code.



```java
spriteCounter++;
if(spriteCounter > 12) { // 12 FPS
    switch (spriteNum) {
        case 1 -> spriteNum = 2;
        case 2 -> spriteNum = 3;
        case 3 -> spriteNum = 4;
        case 4 -> spriteNum = 1;
    }
    spriteCounter = 0;
}
```



```java
image = switch (direction) {
    case "up" -> switch (spriteNum) {
        case 1 -> up1;
        case 2 -> up2;
        case 3 -> up3;
        case 4 -> up4;
        default -> throw new IllegalStateException("Unexpected value: " + spriteNum);
    };
    case "down" -> switch (spriteNum) {
        case 1 -> down1;
        case 2 -> down2;
        case 3 -> down3;
        case 4 -> down4;
        default -> throw new IllegalStateException("Unexpected value: " + spriteNum);
    };
```



```java
case "right" -> switch (spriteNum) {
    case 1 -> right1;
    case 2 -> right2;
    case 3 -> right3;
    case 4 -> right4;
    default -> throw new IllegalStateException("Unexpected value: " + spriteNum);
};
case "left" -> switch (spriteNum) {
    case 1 -> left1;
    case 2 -> left2;
    case 3 -> left3;
    case 4 -> left4;
    default -> throw new IllegalStateException("Unexpected value: " + spriteNum);
};
default -> image;
```