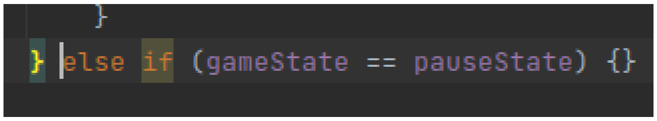
CMPT 276 Group 1 HW3

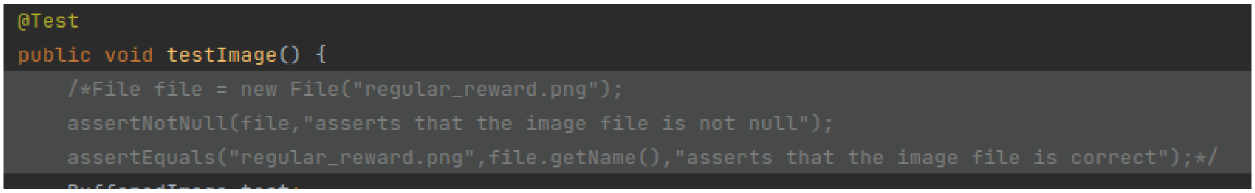
Brendan Murray
Jusung Park

- **Dead code**
  - In the GamePanel class there was a section of dead code that was doing nothing. I removed that code and tested the functionality with it removed. Everything worked well still.



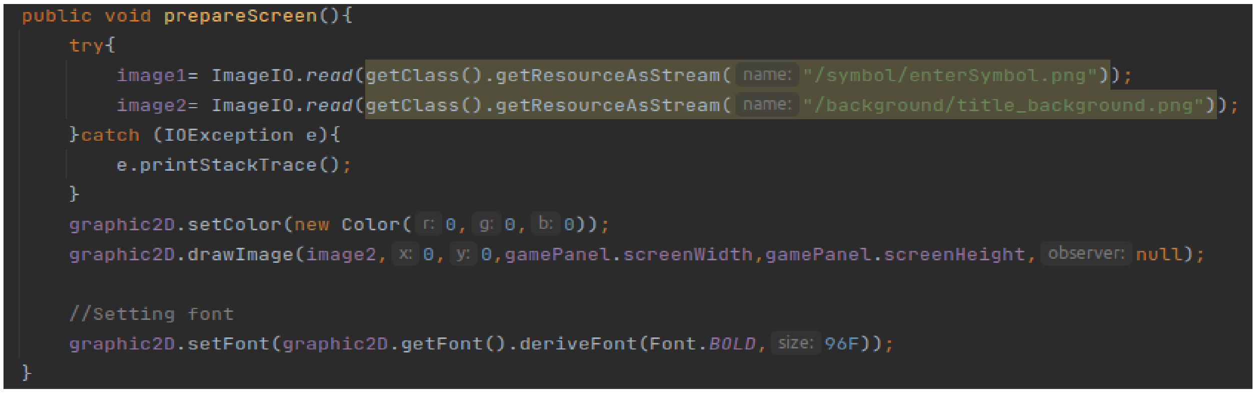
```
    }
} else if (gameState == pauseState) {}
```

  - Found a few imports in many of our classes that were never actually used and were left as remnants from the development phase. Those were all removed.
  - Another instance of dead code occurred in our testing classes, for the regular rewards. There was a block of code that was fully commented out from the development stage of the test. I deleted that block from all 3 tests.



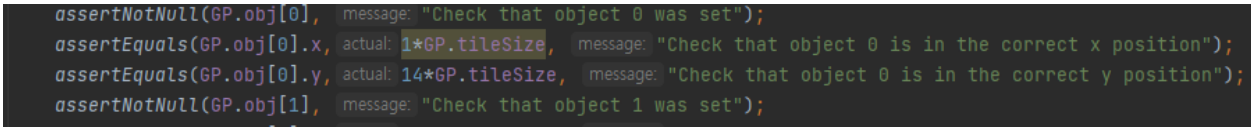
```
@Test
public void testImage() {
    /*File file = new File("regular_reward.png");
    assertNotNull(file,"asserts that the image file is not null");
    assertEquals("regular_reward.png",file.getName(),"asserts that the image file is correct");*/
```

- **Code duplication**
  - In UI class there was duplicated code in two methods that I turned into its own method, the prepareScreen method. This method now would handle loading the pngs for the UI and preparing the background. It would also setup the font that would be used to write to the screen.



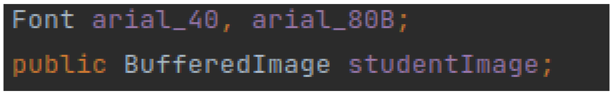
```
public void prepareScreen(){
    try{
        image1= ImageIO.read(getClass().getResourceAsStream( name: "/symbol/enterSymbol.png"));
        image2= ImageIO.read(getClass().getResourceAsStream( name: "/background/title_background.png"));
    }catch (IOException e){
        e.printStackTrace();
    }
    graphic2D.setColor(new Color( r: 0, g: 0, b: 0));
    graphic2D.drawImage(image2, x: 0, y: 0,gamePanel.screenWidth,gamePanel.screenHeight, observer: null);

    //Setting font
    graphic2D.setFont(graphic2D.getFont().deriveFont(Font.BOLD, size: 96F));
}
```

  - In AssetSetterTest, 1*GP.tileSize is a pointless arithmetic expression, and I simplified it to GP.tileSize.



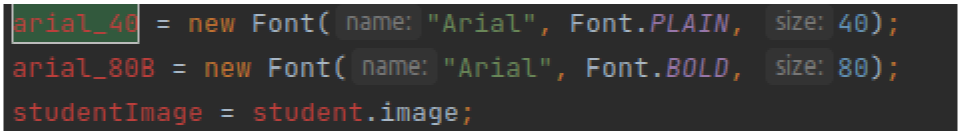
```
    assertNotNull(GP.obj[0], message: "Check that object 0 was set");
    assertEquals(GP.obj[0].x, actual: 1*GP.tileSize, message: "Check that object 0 is in the correct x position");
    assertEquals(GP.obj[0].y, actual: 14*GP.tileSize, message: "Check that object 0 is in the correct y position");
    assertNotNull(GP.obj[1], message: "Check that object 1 was set");
```

- **Unused or useless variables**
    - More dead code was found in the UI class, which had three fields that were never accessed.



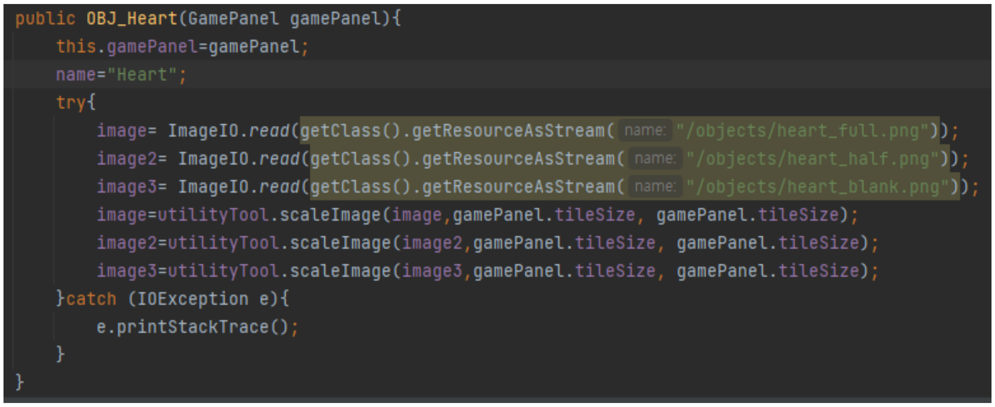
```
Font arial_40, arial_80B;
public BufferedImage studentImage;
```

These fields also had to be removed from the constructor.



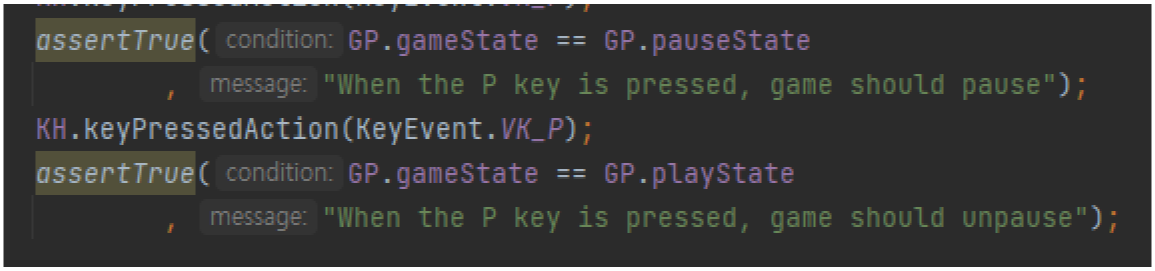
```
arial_40 = new Font( name: "Arial", Font.PLAIN,  size: 40);
arial_80B = new Font( name: "Arial", Font.BOLD,  size: 80);
studentImage = student.image;
```

- **Unnecessary use of unsafe or unsound constructs**
    - In every class that has its own image, the png arguments for the images may be null, which can result in an error.



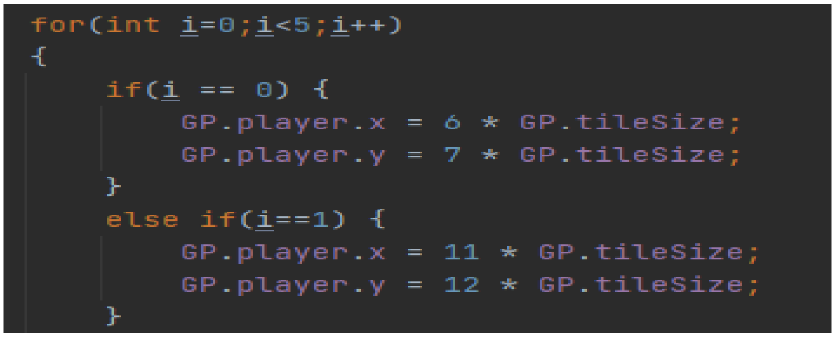
```
public OBJ_Heart(GamePanel gamePanel){
    this.gamePanel=gamePanel;
    name="Heart";
    try{
        image= ImageIO.read(getClass().getResourceAsStream( name: "/objects/heart_full.png"));
        image2= ImageIO.read(getClass().getResourceAsStream( name: "/objects/heart_half.png"));
        image3= ImageIO.read(getClass().getResourceAsStream( name: "/objects/heart_blank.png"));
        image=utilityTool.scaleImage(image,gamePanel.tileSize, gamePanel.tileSize);
        image2=utilityTool.scaleImage(image2,gamePanel.tileSize, gamePanel.tileSize);
        image3=utilityTool.scaleImage(image3,gamePanel.tileSize, gamePanel.tileSize);
    }catch (IOException e){
        e.printStackTrace();
    }
}
```

- In keyHandlerTest, assertTrue() functions can be simplified to assertEquals().



```
assertTrue( condition: GP.gameState == GP.pauseState
        , message: "When the P key is pressed, game should pause");
KH.keyPressedAction(KeyEvent.VK_P);
assertTrue( condition: GP.gameState == GP.playState
        , message: "When the P key is pressed, game should unpause");
```

- **Unnecessary if/else or switch/case statements**
    - TrapTest has so many if statements to test all the traps in the game.



```
for(int i=0;i<5;i++)
{
    if(i == 0) {
        GP.player.x = 6 * GP.tileSize;
        GP.player.y = 7 * GP.tileSize;
    }
    else if(i==1) {
        GP.player.x = 11 * GP.tileSize;
        GP.player.y = 12 * GP.tileSize;
    }
```